TRABZON ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ
BİLGİSAYAR VE ÖĞRETİM TEKNOLOJİLERİ EĞİTİMİ
ANABİLİM DALI

PROGRAMLAMA SÜRECİNDE İÇSEL BİLİŞSEL YÜK OLUŞTURAN KAYNAKLARIN BELİRLENMESİ

YÜKSEK LİSANS TEZİ

Şeval BİLGİ

TRABZON
Ocak, 2020

TRABZON ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ
BİLGİSAYAR VE ÖĞRETİM TEKNOLOJİLERİ EĞİTİMİ
ANABİLİM DALI

PROGRAMLAMA SÜRECİNDE İÇSEL BİLİŞSEL YÜK OLUŞTURAN KAYNAKLARIN BELİRLENMESİ

Şeval BİLGİ

Trabzon Üniversitesi Lisansüstü Eğitim Enstitüsü'nce Yüksek Lisans Unvanı Verilmesi İçin Kabul Edilen Tezdir.

Tezin Danışmanı
Prof. Dr. Ünal ÇAKIROĞLU

TRABZON
Ocak, 2020

Trabzon Üniversitesi Lisansüstü Eğitim Enstitüsü Müdürlüğü'ne

Bu çalışma jürimiz tarafından Bilgisayar ve Öğretim Teknolojileri Eğitimi Anabilim Dalında YÜKSEK LİSANS tezi olarak kabul edilmiştir. 23 / 01 / 2020

Tez Danışmanı	: Prof. Dr. Ünal ÇAKIROĞLU	.....
Üye	: Doç. Dr. Emine ŞENDURUR	.....
Üye	: Dr. Öğr. Üyesi Yasemin KARAL	.....

Onay

Yukarıdaki imzaların adı geçen öğretim üyelerine ait olduğunu onaylarım.

Prof. Dr. Bülent GÜVEN

Enstitü Müdürü

ETİK İLKE VE KURALLARA UYGUNLUK BEYANNAMESİ

Tezimin içerdiği yenilik ve sonuçları başka bir yerden almadığımı; çalışmamın hazırlık, veri toplama, analiz ve bilgilerin sunumu olmak üzere tüm aşamalardan bilimsel etik ilke ve kurallara uygun davrandığımı, tez yazım kurallarına uygun olarak hazırlanan bu çalışmada kullanılan her türlü kaynağa eksiksiz atıf yaptığımı ve bu kaynaklara kaynakçada yer verdiğimi, ayrıca bu çalışmanın Trabzon Üniversitesi tarafından kullanılan "bilimsel intihal tespit programı"yla tarandığını ve hiçbir şekilde "intihal içermediğini" beyan ederim. Herhangi bir zamanda aksinin ortaya çıkması durumunda her türlü yasal sonuca razı olduğumu bildiririm.

Şeval BİLGİ
23 / 01 / 2020

ÖN SÖZ

Programlama sürecinde içsel bilişsel yük oluşturan kaynakların belirlenmesini inceleyen bu çalışma, Trabzon Üniversitesi Eğitim Bilimleri Enstitüsü Bilgisayar ve Öğretim Teknolojileri Anabilim Dalında Yüksek Lisans Tezi olarak hazırlanmıştır.

Bu çalışma süresince danışmanım olan ve hiçbir zaman yardımını esirgemeyen çalışmanın tüm detaylarında bana destek veren, gerek konunun belirlenmesinde gerekse çalışmanın yürütülmesi sırasında engin bilgi ve deneyimlerinden sürekli yararlandığım değerli hocam, Prof. Dr. Ünal ÇAKIROĞLU'na sonsuz teşekkürlerimi sunarım.

Seminerlerim esnasında yönlendirmeleriyle bana yardımcı olan tüm hocalarıma teşekkürlerimi sunarım.

Yüksek lisans yapmam konusunda beni her zaman teşvik eden ve yanımda olan, tüm eğitim hayatım boyunca ilerlemem için maddi manevi bana destek olan anneme ve babama sonsuz teşekkürlerimi sunarım.

Ocak, 2020
Şeval BİLGİ

İÇİNDEKİLER

ÖZET

Programlama Sürecinde İçsel Bilişsel Yük Oluşturan Kaynakların Belirlenmesi

Bilgi işlemsel düşünme becerisini geliştirmek için problem tabanlı programlamanın önemli bir araç olduğu yapılan birçok araştırma ile ortaya konulmuştur. Programlama öğretimine yönelik uygulanan farklı yöntemler zaman zaman başarılı sonuçlar üretse de programlama dersi alan öğrencilerin ders başarılarının genel olarak düşük olduğu, programlamaya yönelik birçok öğrencinin olumsuz tutumlar geliştirdiği belirtilmektedir. Bu zorluklar arasında programlamanın kendi doğası gereği programlama öğrenme ve program geliştirme sürecinde karşılaşılan bilişsel yüklerin önemli bir yer kapladığı görülmektedir. Dolayısıyla bu çalışma ile programlama sürecinde etkili bilişsel yüke yönelik çalışmalar yapabilmek ve bu noktada bazı tedbirleri işe koşabilmek için programlamada yaşanan zihinsel süreçler içerisinde içsel bilişsel yük kaynaklarının belirlenmesi üzerine odaklanılmaktadır. Bu çerçevede bu çalışma ile programlamada yaşanan bilişsel süreçlerdeki bilişsel yük oluşturan kaynakların belirlenerek sınıflandırılması amaçlanmaktadır. Bu doğrultuda ilk olarak uygulama sürecinde kullanılan problemlerin zorluk durumları iki boyutlu olarak belirlenmiş ve süreçte öğrencilerin algılamış olduğu içsel bilişsel yük ile ilişkilendirilmiştir. Bu çalışmada yöntem olarak açıklayıcı durum çalışması kullanılmıştır. Araştırmanın çalışma grubu Bilgisayar Bilimi dersini alan lise öğrencilerinden oluşmaktadır. Araştırmadan elde edilen bulgular programlama bilgi türü boyutunda analiz edilmiştir. Sonuç olarak, programlama sürecinde yapılan hatalar ve karşılaşılan zorluklar problemin zorluk durumu arttıkça sözdizimsel, anlamsal ve stratejik anlamda karşılaşılan hatalar farklılaşmaktadır. Bu çalışmanın, öğrenciler tarafından zor ve karmaşık olarak görülen programlama öğretiminde öğrenme sürecini kolaylaştırıcı çalışmalar yapabilme noktasında katkı sağlayacağı düşünülmektedir.

Anahtar Kelimeler: Bilgi İşlemsel Düşünme, Programlama, Problem Tabanlı Programlama Öğretimi, Bilişsel Yük, İçsel Bilişsel Yük.

ABSTRACT

Determination the Sources of Intrinsic Cognitive Load in the Programming Process

Many researches have shown that problem-based programming is an important tool for improving computational thinking skills. Although different methods applied to programming teaching produce successful results from time to time, it is stated that students who take programming courses generally have low course success and many students develop negative attitudes towards programming. Among these difficulties, due to the nature of programming itself, it is seen that the cognitive loads encountered in the process of programming learning and program development processes an important. Therefore, this study focuses on determining the sources of intrinsic cognitive load within the mental processes experienced in programming in order to be able to work towards effective cognitive load in the programming process and to run some measures at this point. In this context, this study aims to determine and classify the sources that create cognitive load in the cognitive processes experienced in programming. In this direction, firstly, the difficulty situations of the problems used in the application process were determined in two dimensions and they were associated with the intrinsic cognitive load perceived by the students. In this study, descriptive case study was used. The study group of the research consists of high school students taking Computer Science course. The findings obtained from the research were analyzed in terms of programming knowledge type. As a result, errors and difficulties encountered during the programming process, as the difficulty of the problem increases, syntactic, semantic and strategic errors differ. It is thought that this study will contribute to the process of facilitating the learning process in programming teaching which is seen as difficult and complex by the students.

Keywords: Computational Thinking, Programming, Problem-Based Programming Teaching, Cognitive Load, Intrinsic Cognitive Load.

TABLOLAR LİSTESİ

ŞEKİLLER LİSTESİ

KISALTMALAR LİSTESİ

BİD : Bilgi İşlemsel Düşünme

BYT : Bilişsel Yük Teorisi

ISTE : International Society for Technology in Educational [Uluslararası Eğitim Teknolojileri Topluluğu]

1. GİRİŞ

Uluslararası Eğitim Teknolojileri Topluluğu'nun (ISTE) belirlediği 21. yüzyılda öğrencilerin sahip olması gereken beceriler arasında bilgi işlemsel düşünme becerileri önemli bir yere sahiptir (ISTE, 2016). Bilgi işlemsel düşünme, bir problemin formüle edilmesinde bir bilgisayarın (insanın veya makinenin) etkili bir şekilde gerçekleştirebileceği düşünce süreçleri olarak tanımlanmıştır (Wing, 2006). Birçok farklı tanımlama söz konusu olsa da bilgi işlemsel düşünme becerisi çoğunlukla problem çözme süreçleri ile birlikte ele alınmıştır (AERA, 2012).

Bilgi işlemsel düşünme becerilerinin geliştirilmesi için bilgisayarsız etkinlikler, disiplinlerarası etkinlikler, robotik uygulamalar ve programlama ile problem çözme gibi yollar önerilmektedir. Bu yollar içerisinde farklı eğitim kademesindeki öğrencilere uygulanabilirliği nedeniyle programlama sıklıkla tercih edilmektedir.

Programlama, programlama dili ve yapılarının anlaşılmasından genel problem çözme stratejileri ve programın kullanılacağı alanın bilgisine kadar birçok bilgi ve beceri gerektirdiğinden zor olarak görülür (McGill, Volet ve Hobbs, 1997'den akt. Yurdagül ve Aşkar, 2013, s. 605). Bu zorluk programlama öğrenme ve program yazma sırasında çoğu zaman öğrencilerin algılamış olduğu bilişsel yük olarak karşımıza çıkmaktadır.

Farklı alanlarda olduğu gibi programlama alanında da öğretim materyallerinin veya öğrenme ortamının tasarımından kaynaklanan dışsal bilişsel yükler birçok çalışmada ortaya konulmaktadır. Dışsal bilişsel yükün yanında, programlamanın kendi doğasından kaynaklı içsel bilişsel yük de programlama derslerinde öğrenmeyi ve problem çözmeyi ayrıca zorlaştırmaktadır. Dolayısıyla öğrencilerin programlama sırasında karşılaştıkları durumları ve deneyimledikleri zihinsel süreçleri derinlemesine incelemek programlama alanı için önemli görülebilir. Bu düşünceden hareketle, bu çalışmada programlama sürecinde içsel bilişsel yük oluşturan kaynaklar belirlenmeye çalışılmıştır.

1. 1. Araştırmanın Amacı

Programlama yapmak süreçte öğrencilerin işe koşması gereken birçok bilgi ile karmaşık ve zor bir süreç olarak değerlendirilir. Bu çerçevede bu zorlukları farklı açılardan ele alıp sınıflandıran bazı çalışmalara rastlanmaktadır. Bu zorluk programlama öğrenme ve program yazma sırasında çoğu zaman öğrencilerin algılamış olduğu bilişsel yük olarak karşımıza çıkmaktadır. Bu noktada birçok farklı alandaki öğrenmelerde olduğu gibi öğretim materyallerinin veya ortamının tasarımından kaynaklanan dışsal etkenlerin

oluşturduğu bilişsel yüklerin yanında, programlamanın kendi doğasından kaynaklı içsel bilişsel yük de öğrenmeyi ve problem çözmeyi zorlaştırmaktadır. Bu durum öğrencinin çalışan bellek kapasitesini verimli bir şekilde kullanamamasına yol açar. Dolayısıyla öğrencilerin programlama sırasında karşılaştıkları durumları ve deneyimledikleri zihinsel süreçleri derinlemesine incelemek öğretmenler ve öğretim tasarımcılarına önemli ipuçları sağlayabilir. Bu çerçevede dışsal etkilerin rolünü ortaya koyan çalışmalara kısmen rastlanılsa da doğrudan öğrencilerin deneyimledikleri zorluğu ve algıladıkları yükü ortaya koyan çalışmalara ihtiyaç söz konusudur. Bu durum bu yükün oluşum durumunun gözlenmesi, yükü oluşturan kaynakların belirlenmesi ve yükün ölçülmesi için farklı yöntemlerin önerilmesini gerekli kılar.

Bu düşünceden hareketle çalışmanın amacı programlama sürecinde içsel bilişsel yük oluşturan kaynakların belirlenmesidir. Bu kapsamda araştırma aşağıda bulunan problemler çerçevesinde yürütülmüştür.

1. Programlama öğretiminde problemlerin zorluğunun belirlenmesi için öğe etkileşimi temelli önerilecek yöntemin özellikleri nelerdir?
2. Programlama sürecinde öğrencilerin deneyimledikleri içsel bilişsel yük kaynakları nelerdir?
3. Önerilen yöntem doğrultusunda belirlenen problemlerin zorluk durumu ile öğrencilerin deneyimlediği içsel bilişsel yük arasında nasıl bir ilişkisi söz konusudur?

1. 2. Araştırmanın Gerekçesi ve Önemi

Literatür incelendiğinde programlama öğreniminde karşılaşılan zorluklar ve bu zorlukların bilişsel yüke yol açtığını ifade eden birçok çalışma vardır. Bu süreçte bu zorlukları ortadan kaldırmak amacıyla çeşitli yöntemler önerilmiştir (Bonar ve Soloway, 1983; Coull ve Duncan, 2011; Kazımoğlu, Kiernan, Bacon ve Mackinnon, 2012; Lahtinen, Mutka ve Jarvinen, 2005).

İçsel bilişsel yükün öğrenciye verilen belirli bir göreve yönelik öğe etkileşimi seviyesi ile belirlenebildiği varsayılmaktadır (Van Merrienboer ve Sweller, 2005). Öğrencinin verilen görev karşısında gösterdiği davranışlar incelenerek içsel bilişsel yük durumu tahmin edilebilir. İçsel bilişsel yükün programlama sürecinde öğrenci davranışlarının gözlenmesi yoluyla ortaya konulması içsel bilişsel yükü oluşturan ilişkilerin belirlenmesinde özgün bir yol olarak değerlendirilebilir. Bununla birlikte programlama sürecinde işe koşulan programlama yapıları ve kullanımının oluşturduğu bilişsel yükün bilgi türü bağlamında sınıflandırılması, programlama sürecinde problem çözümünde içsel bilişsel yük hakkında

alana bilgi sunacaktır. Bu çerçevede bu çalışma ile her ne kadar doğrudan içsel bilişsel yükün derecelendirilmesine yönelik bir yöntem ortaya konulmayacak olsa da gözlemler üzerinden elde edilecek veriler ile programlamada içsel bilişsel yük kaynaklarını oluşturan verilerin ortaya konulacak olması çalışmayı benzerlerinden farklılaştıracaktır.

Bilişsel yük çalışmalarında çoğunlukla yoğunlaşılan dışsal bilişsel yük kaynaklarının belirlenmesi ve giderilmesi yerine bu çalışmanın içsel bilişsel yüke odaklanması, bilişsel yük çalışmaları için yol gösterici özellikler içermektedir. Bilişsel yükün belirlenmesi için kullanılacak ekran kayıtları, beden dili, sesli düşünmeler ve gözlem formundan elde edilecek veriler programlama süresince karşılaşılan zorluklar ve bu zorlukların giderilmesi noktasındaki çalışmalarda kullanılabilecek olması yönüyle önemlidir.

Programlama öğretiminde içsel bilişsel yük kaynaklarını belirlemek, bu kaynakların insan bilişinin çözmekte zorlanmayacağı şekilde sunulması açısından önemlidir(Yousuf, Sapiyan ve Kamaluddin, 2006). Bu süreci kolaylaştırmak özellikle şema inşası gibi doğrudan öğrenmedeki zihinsel faaliyetler ile ilgili süreçlere ekstra çaba harcanmamasını sağlayarak hem öğretmen hem de öğrenci açısından yararlı olacaktır. Bu kaynakların belirlenip programlama öğretiminde kullanılan uygulamalara ve problemlere entegre edilmesi programlama öğretimi alanına katkı sağlayabilir. Ayrıca bu çalışma ile programlama için oluşturulmuş problemlerin bilişsel yük potansiyelini ortaya koyabilmek adına, üzerinde çalışılacak problemlerin hangi özellikleri taşıması gerektiğine yönelik fikir vermesi açısından çalışmanın alana katkıları olacağı değerlendirilebilir.

Diğer taraftan programlama öğreniminde yaşanan bilişsel süreçlerdeki bilişsel yük oluşturan durumların belirlenmesine yönelik çalışmaların oldukça sınırlı olduğu görülmektedir. Bu yönüyle çalışma programlama sürecinde temel programlama yapılarından kaynaklı içsel bilişsel yükün belirlenmesi, programlama öğretimi alanı için önem arz etmektedir.

1. 3. Araştırmanın Sınırlılıkları

Araştırma;

1. Lise 1 ve lise 2. Sınıf 5 öğrenci ile
2. 3 ay süren eğitim ve 5 haftalık uygulama ile
3. Python programı ile sınırlıdır.

1. 4. Araştırmanın Varsayımları

Araştırma kapsamında bulunan öğrenciler üzerinde, uygulama koşulları dışındaki etkilerin aynı olduğu ve önemli özel bir etkilenmenin olmadığı varsayılmıştır.

1. 5. Tanımlar

Bilgi İşlemsel Düşünme: Bilgi işlemsel düşünme, örüntü tanıma, soyutlama, ayrıştırma, planlama, varsayımsal akıl yürütme, veriyi mantıksal şekilde düzenleme ve analiz etme gibi farklı becerileri bünyesinde barındıran çoğunlukla problem çözme becerileriyle ilişkilendirilen bir düşünme biçimi olarak karşımıza çıkmaktadır.

Bilişsel Yük: Bilişsel sisteme belirli bir görevi yerine getiren yükü temsil eden bir yapı olarak kabul edilir.

İçsel Bilişsel Yük: Konunun zorluğuyla ve öğrenilecek içeriğin karakteristik özellikleriyle ilgili olan yük olarak tanımlanır.

2. LİTERATÜR TARAMASI

2. 1. Araştırmanın Kuramsal Çerçevesi

Bu çalışma programlama öğrenimi sürecinde programlama problemlerinin oluşturduğu içsel bilişsel yükü ortaya çıkarabilecek birtakım yollar üzerine odaklanmaktadır. Dolayısıyla bu araştırma programlama öğretimi çalışmaları arasında değerlendirilebilir. Diğer yandan programlama öğretiminin genel hedeflerinden birisi olan bilgi işlemsel düşünme becerisi gelişimi araştırmanın ilişkili olduğu temel beceriler arasında sayılabilir. Bu bağlamda araştırmanın bilgi işlemsel düşünme becerisi, programlama öğretimi ve bilişsel yük gibi kavram ve yöntemler ile ilişkili olduğu düşünülebilir.

2. 1. 1. Bilgi İşlemsel Düşünme ve Programlama

Bilgi işlemsel düşünme; problemlerin çözümünde örüntü tanıma, soyutlama, ayrıştırma, planlama, varsayımsal akıl yürütme, veriyi mantıksal şekilde düzenleme ve analiz etme gibi farklı becerileri bünyesinde barındıran ve çoğunlukla problem çözme becerileriyle ilişkilendirilen bir düşünme biçimi olarak karşımıza çıkmaktadır (Bundy, 2007; ISTE ve CSTA, 2011'den akt. Barut, Tuğtekin ve Kuzu, 2016, s. 211). Bilgi işlemsel düşünme becerilerinin geliştirilmesi için bilgisayarsız etkinlikler, disiplinlerarası etkinlikler, robotik uygulamalar ve programlama ile problem çözme gibi yollar önerilmektedir. Bu yollar içerisinde farklı eğitim kademesindeki öğrencilere uygulanabilirliği nedeniyle programlama sıklıkla tercih edilmektedir.

Birçok araştırmacı programlama ile problem çözen öğrencilerin bilgi işlemsel düşünme becerilerinin gelişebileceğine yönelik değerlendirmeler ve kanıtlar öne sürmektedir (Barut vd., 2017; Djambong ve Freiman, 2013). Programlama sürecinde prosedürel ve koşullu akıl yürütme, planlama ve ilişkilendirme gibi karmaşık bilişsel beceriler işe koşulduğundan bu becerilerin bilgi işlemsel düşünme alt becerilerinin gelişmesinde rol oynayabileceği değerlendirilmektedir (Barut vd., 2017; Moons ve Backer, 2012).

Bu çerçevede bilgi işlemsel düşünme becerilerinin geliştirilmesi için dünyada ve Türkiye'de Bilişim Teknolojileri ile ilgili derslere programlamaya ilişkin kazanımlar entegre edilmeye çalışılmaktadır. Bununla birlikte programlama ile problem çözme öğretimine yönelik birçok yöntem önerilmektedir (Berland ve Wilensky, 2015; Chao, 2016; Djambong ve Freiman, 2013; Gülbahar, 2018; Lye ve Koh, 2014).

2. 1. 1. 1. Programlama Öğretim Yöntemleri

Araştırmalar, öğrencilerin programlama derslerinde çeşitli zorluklarla karşılaştıklarını ortaya koymaktadırlar (Yousuf vd., 2006). Bu duruma çözümler üretme için programlama öğretim yöntemleri çerçevesinde programlamanın en iyi şekilde nasıl öğretileceğine yönelik birçok araştırma ortaya konulmuştur (Bonar ve Soloway, 1983; Coull ve Duncan, 2011; Kazımoğlu vd., 2012; Lahtinen vd., 2005).

Tamamlama stratejisi, çalışılmış örnek etkisi, amaçsız problemler bu yöntemler arasındadır (Kirschner, 2010). Çalışılmış örnekler fizik, programlama ve matematik gibi alanlar için beceri edinme sırasında en önemli yöntemlerdendir (Van-Lehn, 1996'dan akt. Moreno, 2006, s. 170). Bu yöntem, "bir sorunun nasıl çözüldüğünü ya da verilen görevin nasıl gerçekleştirildiğini öğretebilmek amacıyla çözüm yolunun öğrenene adım adım verildiği bir öğretim yöntemidir" (Clark, Nyugen, Sweller ve Baddeley, 2006'dan akt. Tepgeç, 2017, s.19). Bir diğer yöntem olan tamamlama stratejisi öğrencilere kısmi çözümleri tamamlamaları gereken problemler sunan bir yöntemdir (Van Merrienboer ve Paas, 1990). Araştırmalar incelendiğinde dijital oyunlar(Berland ve Lee, 2011) ve görsel programlama araçlarının(Cevahir ve Özdemir, 2017; Çakıroğlu vd., 2018) programlama öğretiminde kullanıldığı görülmektedir. Görsel programlama ortamlarında öğrenciler, öğrenmede zorluk yaşadıkları soyut kod ve komutları, adım adım somutlaştırarak öğrenebilmektedirler (Saygıner ve Tüzün, 2017).

Yapılan araştırmalar incelendiğinde programlama öğretilirken programlama ile çözülebilen problemlerin sıklıkla kullanıldığı görülmektedir. Bu problemler içerisinde verilen öğrenme görevleri genellikle karmaşık bilişsel süreçler gerektirmektedir. Bu karmaşıklık problemlerin ve programlama sürecinin içerdiği doğal zorluk olarak ifade edilebilir. Bu çalışmalarda karşılaşılan zorluklarda özellikle çalışan bellek kapasitesini verimli bir şekilde kullanan yenilikçi öğretim yöntemlerinin tasarımı önerilmektedir (Sweller, Van Meerienboer ve Paas, 1998). Bu noktada karmaşık görevlerde öğrencilerin yüksek derecede karşılaşılabilecek zorlukların belirlenmesi ve bunların azaltılması için yeni öğretim yöntemleri geliştirmesi gerektiği ifade edilmektedir (Van Merrienboer ve Sweller, 2005). Bu çerçevede programlama sürecinde karşılaşılan bazı zorluk durumları aşağıda tartışılmaktadır.

2. 1. 1. 2. Programlama Öğrenme Sürecinde Karşılaşılan Zorluklar

Programlama becerisi, problem çözerek bireyin üst düzey düşünme becerilerinin geliştirilmesinde önemli bir araç olarak tanımlanmaktadır (Chao, 2016'dan akt. Djambong

ve Freiman, 2013, s. 42; Fessakis, Gouli ve Mavrodi, 2013'ten akt. Özmen ve Altun, 2014, s. 10; Papert, 1991). Programlama ile problem çözme karmaşık bir süreç olarak görülür.

Programlama, programlama dili ve yapılarının anlaşılmasından genel problem çözme stratejileri ve programın kullanılacağı alanın bilgisine kadar birçok bilgi ve beceri gerektirdiğinden karmaşık görülür (McGill, Volet ve Hobbs, 1997'den akt. Yurdagül ve Aşkar, 2013, s. 605). Bu karmaşıklıkta programlamanın yanında problem çözmenin de özgün yapıları, ve bu yapıların ilişkilerinin ortaya konulması önemli bir neden olarak ortaya çıkmaktadır. Genel olarak programlama sürecinde öğrenciler; sözdizimsel, anlamsal(yapısal), stratejik bilgi türleriyle karşılaşır ve bunların anlamlı ilişkilerini oluşturmaya çalışırlar. Bu bilgi türlerinin programlama sürecinde kullanımı bir yandan bu bilgilerin kendi doğaları, diğer taraftan da birbirleriyle olan ilişkileri nedeniyle kolay değildir. Örneğin, programlama dillerinde sözdizimleri bazı öğrenenlere çok karmaşık gelebilmektedir (Gomes ve Mendes, 2007).

Karmaşık bilişsel görevlerde, ilgili bileşen becerilerinin sayısı ve doğası zihinsel çabayı etkiler. Diğer bir ifadeyle, sergilenmesi gereken beceriler için çözülmesi gereken sorunların hedef hiyerarşileri ne kadar karmaşık olursa o zihinsel becerileri sergilemek o kadar zorlaşır (Paas ve Van Merrienboer, 1994). Bu çerçevede programlama öğrenim sürecinde ortaya çıkan sorunların kaynaklarının belirlenmesi ile ilgili çalışmalarda bu süreçteki bilişsel gereksinimler öne çıkar. Bu gereksinimler çoğunlukla programlamadaki sözdizimsel, anlamsal ve diğer beceriler ile ilişkili görülen zorluk alanlarında ele alınır (Mow, 2008). Örneğin; programlama öğrenen öğrenciler, başlangıçta sadece kod sözdizimini öğrenmekle kalmayıp, herhangi bir derleme başarısızlığından veya basit bir koşul yapısının(if) uygun olmayan kullanımından veya tipik bir veri tipinde saklanamayan verinin yol açtığı hatalar ile de karşılaşabilirler (Coull ve Duncan, 2011).

Bu anlamda programlama sürecindeki bilişsel süreçlerde öğrencilerin belleklerinin kapasitelerini kullanabilmeleri önem arz eder. Bu süreçte farklı bilgilerin saklanması, kullanılması ve ilişkilendirilmesi belleğin farklı yapıları tarafından ele alınır (Sweller, 1994). Bu nedenle programlama sırasında gerçekleşen bilişsel süreçlerin tanımlanması programlama sürecinin kolaylaştırılması için yapılacak çalışmalar için önemlidir.

2. 1. 2. Programlama Öğreniminde Gerçekleşen Bilişsel Süreçler

Programlama; yüksek düzeyde bir soyutlama, genelleme, transfer etme, eleştirel düşünme(Gomes ve Mendes, 2007), prosedürel ve koşullu akıl yürütme, planlama ve analojik akıl yürütme gibi karmaşık bilişsel becerileri gerektirir (Kurland, Pea, Clement ve Mawby, 1986'dan akt. Moons ve Backer, 2012, s. 369). Programlamada, öncelikle programlama dili yazım kurallarının bilinmesi, bireyin öğrendiklerini anlamlandırma ve bir

problemle karşılaştığında çözüm yollarını yordayabilmesi gerekir (Renumol, Jayaprakash ve Janakiram, 2009). Bu çerçevede programlama bilgisinin sözdizimsel, anlamsal ve stratejik olmak üzere temel olarak üç bilgi türünden oluştuğu ifade edilir (Coull ve Duncan, 2011).

Sözdizimsel (Syntactic) Bilgi: Derleyicilerin programlama dilini makine diline çevirebilmesi için yazılan programların belli kurallara uygun şekilde yazılması gerekmektedir. Bir değişkenin tanımının nasıl yapılacağı, fonksiyonların nasıl tanımlanacağı, satır sonlarına noktalı virgül konulup konulmayacağı gibi bilgiler sözdizimsel bilgilerdir (Bayman ve Mayer, 1983). Programlama dilindeki sözdizimsel ifadeler, bir program oluşturmak için kullanılan temel eylemleri tanımlar ve bir operatörden veya değişkenlerden oluşan bir plan yapısı ve bu öğeleri yazmak için bir şema yapısı vardır. Program yazma sırasında öğrenciler çoğunlukla bu ifadeleri eksiksiz hatırlamak, kullanım yer ve sıralarını bilmek durumunda kalırlar.

Anlamsal/Kavramsal (Semantic-yapısal) Bilgi: Programlama dilinde kullanılan komutların ve yapıların yaptığı işlemlere ilişkin bilgi sahibi olmak olarak tanımlanabilir. Programlama dilleri genel olarak bazı ortak kavram ve yapıları kullanırlar. Bu yapılar içerisinde, değişken kullanma, fonksiyon oluşturma, döngü ve karar yapıları bütün programlama dillerinde problem çözmek için kullanılmaktadır. Programlama sırasında öğrencilerin bu yapıların sadece ne anlama geldiğini değil, nasıl kullanıldığını ve diğer yapılarla ne gibi ilişkileri olduğunu bilmesi gereklidir.

Stratejik (Strategic) Bilgi: Sözdizimsel bilgi ve kavramsal bilgiyi bilmek bir program yazmak için yeterli değildir. Bu bilgilerin doğru bir strateji ile ne şekilde kullanılacağı bilgisine de sahip olmak gerekmektedir. Bu noktada programlamada stratejik bilgi; bir taraftan bir problemin çözümü için gerekli algoritmayı oluşturmayı içerirken, diğer taraftan gerekli kavramsal ve sözdizimsel bilgileri programlama dili içerisinde nasıl işe koşulacağının da bilinmesini de içermektedir (Bayman ve Mayer, 1983).

Programlama dilleri temelde bu üç bilgi türünün ilişkilerini içerir. Programlama öğretimi ile ilgili önerilen yöntemler temelde bu üç bilginin ilişkilerinin öğretimini kolaylaştırmaya çalışmaktadır. Örneğin, blok tabanlı ortamlarla sözdizimsel bilgi için gereken komutları hatırlamayı kolaylaştırır(Weintrop ve Wilensky, 2015). Ayrıca blokların bir araya getirilmesiyle anlamsal bilginin de bu süreçteki kullanımı kolaylaştırılır. Benzer biçimde görselleştirilmiş ortamlar, sözdizimini kolaylaştıran görsel programlama ortamları öğrencilerin bir araya getirmek durumunda oldukları bilgileri kullanabilmelerini kolaylaştırmaktadır. Bu noktada programlama sürecini kolaylaştırmak için gerek programlama dillerinde gerekse öğretim yöntemlerinde yapılan iyileştirmelerin öğrencilerin

yaşadıkları zihinsel süreçlerde kısa ve uzun süreli belleklerinde gerçekleşen işlemler için katkı sağladığı düşünülebilir.

Programlama sürecinde temelde problem çözüldüğünden problem çözme sürecindeki bilişsel süreçler programlamada da yaşanmaktadır. Diğer taraftan programlamanın kendi doğası da bilişsel aktiviteler içermektedir (Yousuf vd., 2006). Örneğin; öğrencilerin etkin program yazmaları için, program yazarken noktalama işaretleri ve uygun olan karakterlerin kullanımı, dile özgü kısa yazılan ifadeler, komutların hiyerarşisi gibi sözdizimsel özellikleri bilmeleri önemlidir. Benzer biçimde temel programlama yapıları arasında yer alan koşul yapıları, döngüsel ifadeler, dizi yapıları vb. temel bilgileri uygun yerlerde kullanabilmeleri beklenir (Moons ve Backer, 2012). Bu yapıların görevsel özellikleri ve sözdizimsel yapılarını doğru bir biçimde kullanmak, herhangi bir stratejik düşünme yapılamasa bile oldukça zordur. Bununla birlikte stratejik olarak bir problemin çözümü için gereken programlama yapıları ve bu yapıların hangi sırada kullanılacağı konusunda öğrenciler sıklıkla zorluk yaşarlar. Diğer ifadeyle öğrencilerin algoritma geliştirme ve sözdizimsel kurallara eş zamanlı olarak konsantre olmaları zordur (Gomes ve Mendes, 2007).

Sözdizimsel, anlamsal veya stratejik bilgilerin birlikte kullanılmasının oluşturduğu zorluklar, programlamanın doğasındaki içsel karmaşıklığın yüksek seviyede olduğuna işaret etmektedir (Çakıroğlu vd., 2018; Coull ve Duncan, 2011; Malan ve Halland, 2004; Moons ve Backer, 2012; Yousuf vd., 2006). Bu zorluklar, süreçte öğrencilerin çözüm için uğraşılarını şekillendirmekte ve performanslarını etkilemektedir. Bu sebeple bu zorlukların problemlerde karşılaşılan bilişsel yüklere işaret ettikleri düşünülebilir. Bu çerçevede öğrenciler programlama sırasında birbiriyle etkileşimli birçok bilgiyi belleklerinde değerlendirerek oluşturdukları bilgiyi problem çözmede kullanmakta ya da problemi çözerek yeni programlama bilgileri oluşturmaktadır. Bu durum programlama sırasında öğrencilerin çalışan belleklerinde anlık işlemleri ve ilişkileri işlemelerini, mevcut bilgilerini uzun süreli belleklerinden çağırarak uygun ilişkileri oluşturmalarını gerektirmektedir. Bu ilişkiler bilişsel yük teorisi çerçevesinde içsel bilişsel yük oluşturan durumlar arasında ele alınırlar. Bu noktada birçok farklı alandaki öğrenmelerde olduğu gibi öğretim materyallerinin veya ortamının tasarımından kaynaklanan dışsal etkenlerin oluşturduğu bilişsel yüklerin yanında, programlamanın kendi doğasından kaynaklı içsel bilişsel yük de öğrenmeyi ve problem çözmeyi zorlaştırmaktadır.

2. 1. 2. 1. Programlama Öğretiminde Bilişsel Yük Unsurları

Programlama sürecinin karmaşıklığı, program yazan kişinin algılamış olduğu bilişsel yükün önemli bir kaynağı durumundadır(Yousuf vd., 2006). Bilişsel Yük Teorisi (BYT)

bilişsel yük kavramını, karmaşık bilişsel görevlerin yerine getirilmesinde önemli bir faktör olarak kabul eder (Paas, Tuovinen, Tabbers ve Gerven, 2003).

2. 1. 2. 1. 1. Bilişsel Yük

Bilişsel yük, "genellikle belirli bir görevi yerine getiren yükü temsil eden bir yapı" olarak kabul edilir (Sweller, Van Merrienboer ve Paas, 1998, s. 258). Bilişsel Yük Teorisi bilgi yapıları ve insan bilişi bilgisi arasındaki etkileşimleri kullanan ve bağlamdan bağımsız olarak öğrenme sürecindeki belleğe olan yüklemeyi tanımlamaya çalışan bir teori olarak ilgili çalışmalara yön vermektedir (Van Merrienbour ve Sweller, 2005). BYT'nin odak noktası olan karmaşık bilişsel görevleri öğrenme ve gerçekleştirme başarısızlıkları, mevcut bilişsel kapasiteyi aşan görev taleplerine, bilişsel kaynakların yetersiz tahsisine veya her ikisinin birden gerçekleşmesine bağlı olarak ortaya çıkmaktadır (Paas vd., 2003).

BYT, çalışma belleğinin öğrenme sürecindeki rolüne odaklanır (Cooper, 1998; Paas vd., 2003). Bilişsel Yük Teorisi *içsel bilişsel yük(intrinsic cognitive load), dışsal bilişsel yük (extraneous cognitive load)* ve *etkili (ilgili) bilişsel yük(germane or effective cognitive load) olmak üzere* üç tür bilişsel yük olduğunu varsayar (Paas vd., 2003). Belirli bir zamanda çalışma belleğinde uygulanan zihinsel aktivitenin oluşturduğu bu yüklerin toplanabilir olduğu varsayılır (Cooper, 1998).

Dışsal bilişsel yük, temel olarak yetersiz tasarlanmış öğrenme materyallerinden kaynaklanan yüktür (Paas vd., 2003). Bu tür bilişsel yük, öğrenme için gerekli olmayan ve öğretici müdahalelerle değiştirilebilen yük olarak da değerlendirilebilir (Van Merrienboer ve Sweller, 2005; Van Merrienboer, Kester ve Paas, 2006). Bazı durumlarda öğretim materyalinin formatından da kaynaklanabilir (Sweller vd., 1998). Dışsal bilişsel yükün gerektirdiği bilişsel kaynaklar, çalışma belleği kapasitesinin sınırlarını aştığında bilişsel yüke neden olduğu düşünülür (Gerjets, Scheiter ve Catrambone, 2004).

Etkili bilişsel yük, bellekte şemaların oluşturulmasını kolaylaştırarak öğrenme sürecinde karşılaşılan yükün kısmen az algılanmasını sağlar (Paas vd., 2003; Sweller vd., 1998). Örneğin önceden çalışılan örnekler ile öğrenciler problemlere alıştırılabilir. Bilişsel olarak zorlayıcı olan bilgiler, içerikler veya problemlerde problemi oluşturan parçaları belirleme ve bu parçalar arasında ilişkiler oluşturma ile etkili bilişsel yük meydana gelebilir (Gerjets vd., 2004). Bir başka ifade ile etkili bilişsel yük, öğrenenin bilişsel yapılarını ve performansını artıran yük olarak da ifade edilebilir (Van Merrienboer vd., 2006). Bu noktada öğrenmeye doğrudan etki edebildiği değerlendirilebilir. Bilgi uzun süreli bellekte saklanmadan önce, çalışma belleğinde ayıklanmalı ve manipüle edilmelidir (Paas vd., 2003). Bu noktada etkili bilişsel yük, öğrenme süreçlerine şema kullanımı ve otomasyonu

çalışan bellek yükünü önemli ölçüde azaltmaya yardımcı olur (Sweller, 1994). Şema, belirli nesnelerin, olayların veya faaliyetlerin genel kategorilere atanmasına izin veren bilişsel yapılar olarak ifade edilebilir (Merrienboer ve Paas, 1990). Şemaları kullanmak, bilginin kolayca işlenmesine veya işlenme sürecinin otomatikleştirilmesine izin vererek öğrenme sürecini kolaylaştırabilir (Sweller, 1990). Bir şema, bireyin ilgili konunun bileşenlerine ilişkin oluşturduğu, bilginin öğelerini kategorik olarak düzenleyen ve uzun süreli bellekte saklayan bilişsel bir yapıdır (Pollock, Chandler ve Sweller, 2002; Yousuf vd., 2006). Sweller, (1990) anlamlı şemalar oluşturup, gerektiğinde kullanabilmenin bilginin otomatik olarak işlenmesine izin vereceğini bu şekilde bilişsel yükü azaltabileceğini ifade etmektedir. Şemalar; kişiden kişiye farklılık göstermekle birlikte, uygun şemalar oluşturmaya yönelik müdahaleler bireyin çalışma belleğini uygun biçimde kullanabilme potansiyelini geliştirebilir. Bu durum etkili bilişsel yük çerçevesinde ele alınır.

İçsel bilişsel yük, görev talepleri için yeterli bilişsel kaynakların ayrılamaması durumu olarak değerlendirilebilir. Bu noktada öğrenciler şema oluşturma ve şema inşası gibi doğrudan öğrenmeyle ilgili süreçler için fazladan çaba harcayabilir(Gerjets vd., 2004). İçsel bilişsel yük, konunun zorluğuyla (Cooper, 1998) ve öğrenilecek içeriğin karakteristik özellikleriyle ilgili olan yük olarak tanımlanır (Jong, 2010). İçsel bilişsel yük değiştirilemediği için öğretimin dışsal bilişsel yükü azaltacak şekilde tasarlanması zorunlu olabilir (Seufert, Janen ve Brunken, 2007).

İçsel bilişsel yükte, bir içeriği oluşturan alt bilgiler, gerekli ön kazanımlar ve bunların etkileşimleri ön plandadır. Bu çerçevede öğelerin etkileşimi aracılığıyla, öğrenilen içeriğin doğası ve öğrencilerin uzmanlığı arasındaki etkileşim ile belirlenir. Etkileşimde ne kadar fazla öğe olursa, çalışma belleği bilişsel olarak o kadar çok yüklenir (Paas vd., 2003). Bilginin doğası gereği içerdiği karmaşıklık ve içsel bilişsel yük ile başa çıkmak için çalışan bellek yapısını bilmek önemlidir (Sweller, 2010). Yüksek içsel ve yüksek dışsal bilişsel yükün bir kombinasyonu öğrenmeyi önemli ölçüde zorlaştırabilir, çünkü çalışma belleği belirgin ölçüde aşılabilir.

Programlama öğrenme süreci göz önüne alındığında acemi programcıların şemaları uzmanınki kadar zenginleşmiş olmadığından bir problemi çözmek için daha fazla zaman harcaması ve daha çok bilişsel yük deneyimlemesi beklenebilir (Yousuf vd., 2006).

2. 1. 2. 2. Öğrenme Sürecinin Etkililiğinde Bilişsel Yükün Rolü

Bilişsel yük ile ilgili araştırmalarda bilişsel yükün nasıl ölçüleceği konusu bazı kesin olmayan durumları barındırır (Paas vd., 2010). Bilişsel yükün nasıl belirleneceği sorusu, araştırmacılar için yükün çok boyutlu karakteri ve performans, zihinsel yük ve zihinsel çaba arasındaki karmaşık ilişkiler nedeniyle zordur (Sweller, 1998).

Bilişsel yük miktarına ilişkin kabul edilebilecek bir yük seviyesi için kesin değerler bulunmamakla birlikte, bu anlamda oluşan toplam yük üzerinden değerlendirmeler yapılmaktadır. Bu doğrultuda öğretim yöntemlerinin düzenlenmesi için birtakım düzenlemeler yapılabilmektedir. Bu çerçevede belirli bir performans seviyesiyle ilişkili bilişsel çabaların ne kadar olduğuna ilişkin bilgiler için, öğrenenin sergilediği performansı artırıcı önlemlerin azlığı veya çokluğu üzerinden çıkarımlar yapılması mümkündür. Nitekim Paas vd. (2003), zihinsel çaba ve performans ölçütlerinin birleşimi, bilişsel yük hakkında önemli bilgiler ortaya çıkarabileceğini ve özellikle performansa dayalı ölçümler içerisinde bilişsel yükün rolü bağlamında öğretimin verimliliğine ilişkin yorumlamalar yapılabileceğini ortaya koymaktadırlar.

Bir öğretim sürecinin verimliliği, performans sırasında Paas ve Van Merrienboer (1993) tarafından matematiksel olarak test veya performansın sonucu üzerinde performans ve zihinsel çaba olarak tanımlanmış olan performans ve zihinsel çabanın birleşik bir hesaplamasıdır (Vogel-Walcutt, Gebrim, Bowers, Carper ve Nicholson, 2011). Diğer bir ifadeyle, öğrenenlerin sergiledikleri performans için ne kadar zihinsel çaba harcadıkları veya ne kadar bilişsel yük ile mücadele ettiklerine yönelik ölçümler üzerinden öğretimin verimliliği hesaplanabilir. Bu hesaplama içerisine düşük çaba ile ilişkili yüksek görev performansı, yüksek öğretim verimliliği olarak değerlendirilirken yüksek çaba gerektiren düşük görev performansı, düşük öğretim verimliliği olarak düşünülür (Paas vd., 2003). Bu yönüyle öğrenme sürecinin verimliliği de acemilerin uzmanlardan ayırt edilemeyecek bir performans göstermeye başlamalarını sağlayacak bilgi ve beceri kazanma durumu olarak ele alınabilir (Kolfschoten, Lukosch, Verbraeck, Valentin ve Vreede, 2010). Sweller (1998), bu konuda performansa dayalı ölçümler çerçevesinde, öğrenciler tarafından harcanan zihinsel çabanın yoğunluğu ve öğrenciler tarafından elde edilen performansın bir kombinasyonu, öğretim verimliliğinin en iyi tahmincisini oluşturduğunu öne sürer. Öğretim verimliliği hesaplamasında bir göreve ilişkin performans sırasında tahsis edilen çalışma belleği kapasitesiyle ilgili öğrenme sonuçları (test performansı) ele alınır (Paas vd., 2003; Tuovinen ve Paas, 2004'ten akt. Kester, Kirschner, Merrienboer; 2005, s.171). Öğretim verimliliği, koordinasyon sistemindeki belirli bir noktadan, zihinsel çabanın her birimine bir performans biriminin karşılık geleceği varsayımından hareketle "temel" duruma dik bir mesafe olarak hesaplanmaktadır (Van Merrienboer vd., 2002).

2. 1. 2. 2. 1. Bilişsel Yükün Ölçülmesi

Bilişsel yük ölçümü; öznel, fizyolojik ve görev ve performansa dayalı olmak üzere üç ana ölçüm tekniği ile yapılır (Paas vd., 2003; Sweller vd., 1998). Öznel ölçümler,

insanların bilişsel süreçlerini gözden geçirebildikleri ve harcanan zihinsel çabaların miktarını bildirdikleri varsayımına dayanır (Sweller vd., 1998). Bu ölçümler, bireyin kendi bilişsel süreçlerini dikkate alarak bir öğrenme etkinliğini yürütürken ne kadar çaba harcadığını bildirmesiyle yapılmaktadır (Sezgin, 2009). Bu ölçümde bilişsel yükün, öğrencinin zihninde gerçekleşen bir süreç sırasında ortaya çıktığı düşünüldüğünden genellikle dolaylı yollarla ölçmek durumunda kalınır (Sweller vd.,1998). Bu amaçla Paas (1992) tarafından bilişsel yük çalışmalarında kullanılmak üzere geliştirilen ölçekler sıklıkla kullanılmaktadır. Bu ölçeklerde bilişsel yüke maruz kalanların çaba algıları üzerinden öznel derecelendirme ölçekleri geliştirmiştir. Çoğunlukla öğrencilerin öznel değerlendirmelerini içerdiğinden, zaman zaman doğru cevaplanamadığı düşüncesiyle bu ölçeklere eleştiriler getirilmektedir (Paas vd., 2003). *Fizyolojik ölçümler,* bireyde bilişsel olarak gerçekleşen farklılıkların bireyin fizyolojik reaksiyonlarına yaptığı etkiler üzerinden gerçekleştirilir. Bu ölçümler, kalpte, beyinde ve gözlerde meydana gelen reaksiyonların sayısal verilere dönüştürür (Sweller vd., 1998).

Görev-performans ölçümleri, öğrenenlerin davranışlarındaki değişiklikleri verilen farklı birkaç görevin aynı anda yapabilmeleri üzerinden belirler (Sweller, 1998). Bu ölçümlerde bilişsel yük, verilen bir göreve ilişkin sergilenen performanslar üzerinden ölçülebilmektedir. Belli bir performans seviyesine bağlı bilişsel aktivitelerin, görev ve performansa dayalı önlemlerden tutarlı bir şekilde çıkarılması zordur (Sweller vd., 1998). Bunun yerine; zihinsel çaba ölçüleri, performans ve zihinsel yük önlemlerine mutlaka yansıtılmayan bilişsel yük hakkında önemli bilgileri ortaya çıkarabilir (Sweller, 1998). *Zihinsel çaba*, görev taleplerini karşılamak için tahsis edilen bilişsel kapasite veya kaynak miktarını ifade eder. *Performans*, ilgili öğrencinin performansını ifade eder. *Zihinsel yük* ise görev (çevresel) talepleri tarafından yüklenen yükü ifade eder. Bu talepler, öğretimsel müdahalelere göreceli olarak bağışık olan ve etkileşimli tasarımla ilişkili görev dışı yönleri olan öğe etkileşimi gibi görev yönleriyle ilgili olabilir(Sweller, 1998).

2. 1. 2. 2. 2. İçsel Bilişsel Yükün Belirlenmesi

Bilişsel yük teorisinde tanımlanan içsel bilişsel yük, anlaşılması gereken bilginin doğal karmaşıklığı ile ilgilidir (Sweller, 1994; Sweller ve Chandler, 1994). Bu çerçevede bilginin sunum şeklinden öğretimsel müdahalelerin engelleyemediği, önemli ölçüde içeriğin doğası ile ilgili olan yüktür(Sweller, 2010). Bu tanımlama çerçevesinde içsel bilişsel yükün temel olarak öğrenme içeriğine bağlı olduğu, diğer bir ifadeyle öğrencinin belirli bir öğrenme ortamı içinde ele alması gereken görevle belirlendiği söylenebilir (Seufert vd., 2007). İçsel bilişsel yükün tanımlanmasına katkıda bulunan en önemli faktör, öğrenmeye katılması gereken öğelerinin sayısıdır(Cooper, 1998). İçsel bilişsel yük

çalışmalarında, çoğunlukla öz raporlama yollarıyla elde edilen veriler ile yük düzeyi belirlenmeye çalışılır. Bu çalışmalardaki içsel bilişsel yük, öğrenilmesi gereken bir şemaya entegre edilmesi gereken ve bu nedenle aynı anda çalışma belleğinde işlenmesi gereken öğelerin sayısına odaklanır(Cook, 2006; Leahy ve Sweller, 2008; Gerjets vd., 2004; Pollock vd. 2002). Öğrenme görevinin birbiriyle ilişkili unsurların miktarı, bu görevlerin potansiyel karmaşıklığı, bir öğrencinin bilgi birikimlerini birlikte bir öğeye gruplama yeteneğine göre değişir (Seufert vd., 2007). Bu noktada bu bilgi birimlerinin etkileşimi öğrenilecek içeriğin ilişkisel karmaşıklığını oluşturur (Seufert vd., 2006). Bu yükün öğrenci tarafından algılanma durumu öğrencinin öğrenecek olduğu bilgi ile ilgili önceki bilgilerine yönelik oluşturmuş olduğu şemalar ile ilişkili olduğu düşünülebilir (Gerjets vd., 2004).

İçsel bilişsel yük ile ilgili çalışmalar bazı içeriklerin neden diğerlerinden daha zor olduğunu ve bunun bellek üzerinde ne gibi süreçlerde ne şekilde ele alındığını açıklamaya çalışır (Jong, 2010). İçsel bilişsel yükün öğrenciler tarafından algılanmasında öğrenilecek bilgi için gerekli bilgi birimlerine ilişkin mevcut şemalarının önemli rolü vardır. Bir şema, bilginin kategorik olarak soyut düzeyde organize edildiği bilişsel bir yapıdır (Pollock vd., 2002; Seufert vd., 2007). Öğrencinin önceden oluşturmuş olduğu şemalar ne kadar güçlü ise öğrenilecek içerik ile ilgili etkileşimli öğeleri bir şemaya dahil edebilme ihtimali yükselir ve öğrencinin yeni bilgiyi tek bir birim olarak işleyebilmesi sağlanabilir(Cook, 2006). Cook (2006), herhangi bir içeriğin yapısındaki içsel bilişsel yük yüksek olduğunda, şema oluşumu ve var olan şemaların öğrenme sürecinde aktifleştirilmesi için daha fazla çaba gerektireceğini ifade etmektedir. Bu noktada birçok çalışmada içsel bilişsel yüke ilişkin algılanan yük miktarının uzmanlık faktörüne bağlı olduğuna ilişkin düşünceler aslında öğrencinin ilgili içeriğe yönelik mevcut doğru şemalarına işaret etmektedir(Seufert vd., 2007). Bu durumda potansiyel olarak bir içeriğe ilişkin içsel bilişsel yük yalnızca öğrenilen içeriğin niteliğini değiştirerek değiştirilebilir diye düşünülebilir. Ancak, içsel bilişsel yük çalışmalarında bu yükün öğrenci tarafından algılanma durumunu etkileyen faktörlerin öğrenciler arasında farklılık gösterebileceği düşünülür veya kendi kendine öğrenme eylemlerine odaklanılır (Sweller, 2010).

2. 1. 2. 2. 3. Bilişsel Yük Türünün Belirlenmesi

Bilişsel Yük Teorisi'nde hangi ölçme yolu kullanılırsa kullanılsın, ölçülen bilişsel yükün hangi türden yük olduğunun belirlenmesi oldukça güçtür. Nitekim öznel değerlendirmelerde öğrencinin kendisi için "zor" olarak algıladığı durumun materyalin tasarımı ile ilgili dışsal yükten mi, yoksa içeriği oluşturan etkileşimlerden mi kaynaklandığını belirlemek kolay değildir. Benzer biçimde görev tabanlı veya fizyolojik

ölçümlerde de genellikle toplam bilişsel yük ile ilgili ölçümler elde edildiği varsayılır (Paas vd., 2003). Bu çerçevede içsel, dışsal ve etkili yük ayrımını dikkate alarak, araştırmacıların toplam bilişsel yükü ölçtüğünü ve bu üç bilişsel yük bileşeni arasında ayrım yapmak için ölçüm tekniklerinden birini kullanamadıklarını dikkate almak önemlidir (Paas vd., 2003). Ancak bir görevin etkileşimli unsurları analitik olarak tanımlanabilirse, şema yapımını ve otomasyonunu engelleyen görevlerin yönleri ve bu süreçlere faydalı olan görev yönleri ve bunların bilişsel sonuçları deneysel olarak belirlenebildiğinde kesin olmasa da içsel, dışsal ve etkili yük arasında ayrımlar yapabilmek mümkün olacaktır (Paas vd., 2003).

Özetle bu çalışma, BİD Becerileri, Programlama Öğretim Yöntemleri, Programlama ile Problem Çözme, Bilişsel Yük, İçsel Bilişsel Yük Kaynaklarının Belirlenmesi gibi kavramlarla ilişkilidir. Bu temel unsurların çalışma çerçevesindeki ilişkisi Şekil 1' de özetlenmektedir.

Şekil 1. Çalışmanın teorik çerçevesi

2. 1. 3. İlgili Çalışmalar

2. 1. 3. 1. Programlama Öğrenmede Bilişsel Süreçlerin Değerlendirilmesine Yönelik Çalışmalar

Programlamadaki bilişsel süreçleri değerlendiren çalışmalar daha çok sürecin öğrenciler üzerinde bıraktığı zorluk algıları ve öğrencilerin bu zorlukları aşmak için çabaları üzerinden ele alınmaktadır. Bu çalışmalardan birisinde; Milne ve Rowe (2002), birinci sınıftaki öğrencilerin nesne yönelimli programlama dersinde karşılaştığı en yaygın zorlukları öğrencilere web tabanlı bir anket üzerinden belirlemiştir. Sonuç olarak, işaretçiler ve bellekle ilgili kavramların öğrencilerin anlamakta en çok zorlandıkları konular olduğu görülmüştür. Programlama sürecindeki bilişsel süreçlerin incelendiği çalışmaların önemli bir kısmı, dışsal bilişsel yük üzerine odaklanır. Bu çerçevede Moreno ve Valdez (2005), programlama öğrenmede iki dış gösterimin (görsel, sözlü), tek dış gösterime (görsel veya sözlü) göre öğrenmeyi kolaylaştırdığı ve bilişsel yükü azalttığını ifade etmişlerdir. Benzer biçimde Kolfschoten ve diğerleri, (2009) programlama sürecinde, tasarımın bilişsel yükü ve problem çözme becerisi üzerindeki etkisini incelemişlerdir. Sonuç olarak tasarım kalıpları, acemilerin problem çözme ve tasarım becerilerindeki anlayışı daha hızlı kazanmalarına yardımcı olurken, bilişsel yükü azalttığı görülmüştür. Diğer bir çalışmada Renumol ve diğerleri, (2009) programlama eğitimi ile ilgili problemleri araştırmak için, bir grup lisans öğrencisinden toplanan verileri analiz etmiştir. Sonuçlar öğrencilerin, duyuşsal ve psikomotor alanlara kıyasla bilişsel olarak daha fazla zorluklar yaşadığını göstermektedir. Özmen ve Altun (2014), lisans öğrencilerinin programlama derslerinde başarısızlığın nedenleri ve programlamada karşılaştıkları problemler hakkındaki görüşlerini incelemiştir. Öğrencilerin süreçte yaşadıkları zorluklarının temel olarak programlama bilgisi, programlama becerileri, programın anlamını anlama ve hata ayıklama ile ilgili olduğu görülmüştür. Çakıroğlu ve diğerleri, (2018) ise programlama sürecinde karşılaşılan zihinsel süreçleri ortaya koyarken problem çözmenin problemlerin görselleştirilmesinin ötesinde bir şey olduğunu ifade etmektedir. Zorlukların azaltılması için görselleştirme yanında problemlerin yapısının da incelenmesi ve programlanabilirliklerinin belirlenmesinin gerektiğini ifade etmektedirler.

Programlama öğretimi orta öğretimden yükseköğretime kadar birçok yaş grubunda gerçekleşmektedir. Bu çerçevede yükseköğretimde programlama öğretimi ile ilgili olarak programlama öğretiminde yazma ve programı okuma süreçlerinin zorluklarının belirlenmesi de programlama çalışmalarında bilişsel süreçler hakkında değerli ipuçları sunmaktadır. Bu çerçevede Özdinç ve Altun (2014), yaptıkları çalışmada bilişim teknolojileri öğretmen adaylarının program yazarken ortaya koydukları davranışları ve

performansılarını değiştiren etkenleri ve bu etkenler arasındaki ilişkiyi bilişsel görev analizi yöntemi kullanarak incelemişlerdir. Çalışmanın sonucunda öğretmen adaylarından gelişmiş düzeydeki katılımcılar program yazmadar daha başarılı olmuş, orta seviyedekiler ise daha çok program okumada daha çok başarı göstermişlerdir. Bu bulgu ile program yazma ve okuma sürecinde birbirinden farklı bilişsel süreçler gerektiği ve programlamanın farklı özelliklerinin işe koşulduğu sonucuna ulaşılmıştır.

2. 1. 3. 2. Bilişsel Yükün Belirlenmesi ve Yönetilmesine Yönelik Çalışmalar

Programlama sürecinde birçok farklı bilgi türü birlikte kullanılması gerektiğinden bilişsel yük oluşumu kaçınılmaz görülmektedir. Bu yük zaman zaman problemi anlama zaman zaman da çözümü gerçekleştirme sürecinde karşımıza çıkabilmektedir. Dolayısıyla bu alandaki çalışmalar için genel olarak farklı bilişsel yüklerin nasıl belirlendiğine ilişkin çalışmalar yol gösterici olabilir.

Literatür incelendiğinde genellikle bilişsel yükü kontrol etmek için dışsal bilişsel yükü azaltmanın yolları arandığı ve uygulandığı görülmektedir. Bu çalışmalardan birisinde materyali basit-karmaşık bir sıraya göre sıralamak, böylece öğrencilerin başlangıçtaki karmaşıklığı tam olarak deneyimleyememesi, içsel yükü kontrol etmenin bir yolu olduğunu ortaya konulmuştur (Van Merrienboer vd., 2003). Bir başka çalışmada ise dışsal bilişsel yükün tüm kaynaklarının kaldırılmasından sonra bile, karmaşık görevlerin öğe etkileşiminin, verimli öğrenmeye izin vermek için hala çok yüksek olduğu sonucuna ulaşılmıştır. Bu nedenle öğrenme görevlerinin öğe etkileşimini manipüle edilmesinin içsel bilişsel yükü azalttığına işaret etmektedir (Van Merrienboer vd., 2006).

İçsel bilişsel yükün belirlenmesine yönelik yöntemlerin önerildiği bir başka çalışmada Pollock ve diğerleri, (2002) yapay olarak öğe etkileşimini azaltmış ve öğeleri yalıtımlı olarak sunmuşlardır. Öğrenciler çalışma belleği sınırlamalarını önleyebilmiş ve sunulan bilgiler için kısmi şemalar oluşturabilmişlerdir. Sonuç olarak, belirli öğrenci grupları için bilginin yalıtılmış etkileşimli öğeler öğretim yöntemiyle daha iyi öğrenildiği görülmüştür.

Leahy ve Sweller (2008), içsel bilişsel yükün azaltılmasına yönelik olarak hayal gücü etkisinin, düşük öğe etkileşimi yerine yüksek öğe etkileşimi olduğu durumlarda daha güçlü olduğu hipotezini test etmişlerdir. Hayal gücü grubu ve çalışma grubundan sunulan materyalleri “hayal etmeleri” istenerek öğe etkileşimleri belirlenmeye çalışılmıştır. Öğrencilerin yüksek düşük sayıda öğe etkileşimi içeren bilgiyi işlemesi için göreceli olarak daha fazla sayıda etkileşimli öğe (daha yüksek bir içsel bilişsel yük) gerektiren bilgiyi kullanması nedeniyle içsel bilişsel yükte yüksek olan bilgileri kodlayıp kodlamadıkları test edilmiştir. Sonuç olarak, hayal gücü grubu yüksek öğe etkileşim

soruları üzerine çalışma grubundan daha iyi performans göstermiştir. Özellikle içsel bilişsel yükün belirlenmesinde öğe etkileşimleri belirlenirken; öğrencilerden formüllerin, kavramların ve tanımların zorluğunu veya karmaşıklığını derecelendirmelerinin istenildiği çalışmalara rastlanılmaktadır.

Leppink ve diğerleri (2013), çalışmalarında bu şekilde kavramlar arasındaki ilişkilerin karmaşıklığının öğrenci görüşleri üzerinden derecelendirmesinin doğru sonuçlar üretmeyeceğini ifade etmektedir. Bu çerçevede Ayres (2006), içsel bilişsel yükü azaltmanın bir yolunun, problem karmaşıklığını doğrudan azaltmak olabileceğini ifade etmiştir. Bu anlamda, matematiksel bir alanda öğrenmeyi kolaylaştırmak için içsel bilişsel yükü azaltma amacıyla yalıtılmış öğe stratejisini iki grup üzerinde test etmiştir. Bir gruptan problem başına dört hesaplama yapması istenirken, ikinci grubun (yalıtılmış) yalnızca tek bir hesaplama yapması istenmiştir. Bu hipotezi test etmek için, hata oranları ve bilişsel yükün kendi kendine derecelendirme ölçümleri her iki problem grubunda da doğrudan karşılaştırılmıştır. Sonuç olarak yalıtılmış öğe stratejisinin, hem hata oranını hem de bilişsel yükü azalttığı belirlenmiştir.

Hangi alandaki öğrenmeler olursa olsun, bilişsel yük ölçümleri uzun yıllardır tartışılan bir konu olmuştur. Bu noktada düzenlenen çalışmalardan birisinde Ayres (2004), bir başka çalışmasında öğrencilerin verilen görevlerde içsel bilişsel yükü belirlemek için öznel derecelendirme ölçeği kullanmıştır. Dışsal ve etkili bilişsel yükü sabit tutarak, içsel bilişsel yükteki (öğe etkileşimi) değişikliklerin ölçüldüğünü ifade etmiştir. Sonuçlar öznel ölçümlerin oldukça güvenilir olduğunu, problemler içinde önemli ölçüde değiştiğini ve hatalarla oldukça ilişkili olduğunu göstermiştir. Hsu, Chang ve Yu (2012) programlamada öğrenme etkilerini geliştirmek ve öğrencilerin bilişsel yüklerini azaltmak için çift ekranlı bir öğrenme ortamında öğrenenlerin işlem bilgileri arasındaki farklılıkları araştırmışlardır. Araştırma kapsamında bir gruba çift ekranda (ekranlardan biri metinsel açıklamaları gösterirken diğeri programlama sayfasını göstermektedir.) diğer gruba ise tek ekranda öğretim gerçekleştirilmiştir. Bilişsel yükü ölçmek için öğrencilere içerisinde netlik derecesi (dışsal bilişsel yükü ölçümü için) ve zorluk derecesi (içsel bilişsel yük ölçümü için) bulunan derecelendirme ölçeği uygulanmıştır. Sonuçlar, eğitim materyallerini öğrenirken iki öğrenme ortamındaki öğrencilerin içsel yüklerinde bir fark olmadığını ifade eder.

Bu noktada Garner (2002) programlamada kodlara renklendirme uygulamış ve öğrencilerin farklı programlama yapılarını daha kolay hatırlayarak kullanabilmelerini sağlamıştır. Uygulanan yöntemin öğrencilerin bilişsel yükünü azalttığına ilişkin sonuçlar, kodların dış görünüşüne yapılacak müdahalenin bilişsel yük ile ilgili olabileceği fikrini doğurmuştur.

Doğrudan bilişsel yükün belirlenmesine yönelik olmasa da yükün azaltılmasına yönelik programlama dersleri çerçevesinde farklı çalışmalara rastlanılmaktadır. Bu çalışmalardan birisinde Stachel vd. (2013), destekleme araçlarının bilişsel yük seviyelerine etkilerini incelemiştir. Uygulamalar için bir Visual Basic programlama dersinde laboratuvar ödevlerini tamamladıklarında katılımcılardan, bilişsel yüklerini değerlendirmeleri istenmiştir. Sonuçlar, desteklenen grubun destek almayan gruba göre daha yüksek puanlara ulaştığını göstermektedir. Anvari, Tran ve Kavaklı (2013) bilişsel yük ölçümü ve mekansal yetenek testi kullanarak üç boyutlu bilgisayar grafikleri programında yetenekli öğrencileri tanımlamışlardır. Bu çalışmada öğrencilere, görev ve performansa dayalı teknikleri ve algı anketlerini kullanarak mekansal yetenek testi, performans ve bilişsel yük testi kullanarak programlama sürecindeki bilişsel yük algıları üzerinden öğrencilerin programlama yeteneklerini sınıflamışlardır. Moreno'nun (2006) yapmış olduğu çalışmada ve geçmiş çalışmalarda bulunan molar çalışılmış örneklerden ziyade modüler kullanımı için faydalı öğrenme etkileri, içsel bilişsel yük seviyesinin öğretim yoluyla manipüle edilebileceği en güncel BYT gelişimini desteklemektedir.

2. 2. Literatür Taramasının Sonucu

Programlama için öğrencilerin problemleri anlamaları ve çözümler hakkında kolayca uygulanabilir bilgileri öğrenmeleri çok önemlidir (Kolfschoten vd., 2009). Programlama öğretimi için kullanılan programlama dillerinden bağımsız olarak, program yazma ve program anlama sırasında çeşitli düzeylerde bilişsel güçlükler olduğu görülmektedir (Renumol vd., 2009). Bu güçlükler karşımıza bilişsel yük olarak çıkmaktadır. Bilişsel yükü azaltmak veya yönetmek öğretim verimliliği için önemlidir. Bunun için öncelikle bilişsel yükün kaynağının belirlenmesi ve öğrenmeyi ne kadar etkilediğinin belirlenmesi gereklidir. Bu yönde birçok strateji önerilmiştir.

Öğrenilecek içerikte veya çözülecek bir problemde öğeler arasındaki etkileşim azalırsa, ayrı bilgi birimleri oluşturulması önerilmektedir. Bu noktada eşzamanlı olarak bilgi öğelerinin işlenmesi yerine seri işlemler gerçekleştirilebilir ve bu da çalışma belleği yükünü büyük ölçüde azaltabilir (Pollock vd., 2002)**.** Bu anlamda birçok çalışmada yalıtılmış öğe etkileşimi stratejisi ve derecelendirme ölçeklerinin kullanılabileceği ortaya koyulmuştur. Bazı çalışmalar doğrudan her adım veya görevden sonra derecelendirme ölçekleri uygularken ve bu derecelendirmelerin ortalamasını kullanırken, diğerleri bilişsel yükü yalnızca tüm öğrenme veya problem çözme aşamasından sonra değerlendirmektedir (Schmeck, Opfermann, Van Gog, Paas ve Leutner, 2015). Özdinç ve Altun (2014), programlama başarısını etkileyen faktörleri incelerken sadece süreç sonundaki testlerin kullanılmasının yeterli olmayacağını ve programlama sürecinin de incelenmesi gerektiğini

vurgulamışlardır. Bu çerçevede programlamada problemlerin çözülmesi sürecinde hangi öğeler olduğu, bu öğelerin nasıl etkileşimler halinde olduğu, bu etkileşimler bilişsel yük bağlamında nasıl ölçüleceğine ilişkin literatürdeki örneklerden yararlanılarak programlama özelinde bir ölçüm modeline ihtiyaç olduğu değerlendirilmektedir.

Yapılan çalışmalar incelendiğinde programlama becerisi, problem çözerek bireyin üst düzey düşünme becerilerinin geliştirilmesinde önemli bir araç olarak tanımlanmaktadır (Chao, 2016'dan akt. Djambong ve Freiman, 2013, s. 44; Fessakis, Gouli ve Mavrodi, 2013'ten akt. Özmen ve Altun, 2014, s. 10; Papert, 1991). Bu çalışmalarda programlama ile problem çözme karmaşık bir süreç olarak görülür. Bu süreçte programlamanın doğasından kaynaklı bazı zorluklar öğrencinin algılamış olduğu bilişsel yük olarak ortaya çıkar. İncelenen araştırmalarda genellikle bilişsel yükü azaltmaya yönelik çalışmalar yapıldığı, bu yükü oluşturan kaynakların belirlenmesine yönelik çalışmaların sınırlı olduğu görülmektedir. Bu durumdan hareketle bu çalışmanın farklı bölümlerine yön veren bazı çalışmalar Tablo 1'de özetlenmektedir.

Tablo 1. Araştırmanın Şekillendirilmesine Literatürün Katkısı

Çalışma Bölümü	Kaynaklar
Bilgi işlemsel düşünme ve Programlamaya ilişkin çerçeveler	AERA (2012); Barut vd. (2017), Djambong ve Freiman (2013), Gülbahar (2017), ISTE (2016), Kalelioğlu, Gülbahar ve Kukul (2014), Moons ve Backer (2012), Wing (2006)
Programlama ve Öğretim Yöntemleri	Bonar ve Soloway (1983), Coull ve Duncan (2011), Gomes ve Mendes (2007), Kazımoğlu vd. (2012), Kirschner (2010), Sweller vd. (1998), Van Merrienboer ve Paas (1990), Paas ve Van Merrienboer (1994), Renumol vd. (2009)
Bilişsel Yük	Cooper (1998), Gerjets vd. (2004), Seufert vd. (2007), Sweller, Van Merrienboer ve Paas (1998), Jong (2010), Paas vd. (2003), Paas vd. (2010)
Araştırma probleminin belirlenmesi	Ayres (2006), Cook (2006), Leahy ve Sweller (2008), Pollock vd. (2002)
Veri toplama araçlarının seçilmesi Araştırma yönteminin belirlenmesi	Aytaçlı (2012), Gökçek (2009), Leymun, Odabaşı, Yurdakul (2017), Patton (2014)
Uygulamanın yapılması	Van Merrienboer vd. (2003)

3. YÖNTEM

Bu araştırmada, programlama sürecinde içsel bilişsel yük oluşturan kaynakların belirlenmesi amaçlanmaktadır. Araştırma 3 aylık eğitim sürecini kapsamaktadır. Eğitim sonrası 1 hafta pilot uygulama yapılmıştır. Pilot çalışmanın ardından süreç düzenlenerek 4 haftalık uygulama süreci tamamlanmıştır. Bu çerçevede bu bölümde araştırmanın modeli, çalışma grubu, süreci, ölçme araçları ve uygulanması, verilerin toplanması ve verilerin analizi açıklanmıştır.

Araştırma 3 temel aşamada yürütülmüştür. İlk olarak, programlama ile çözülebilecek problemlerin zorluklarının belirlenmesi için öğe etkileşimi temelli bir yöntem önerilmiştir. Daha sonra öğrencilerin deneyimledikleri içsel bilişsel yük kaynakları belirlenmiş ve son olarak deneyimlenen bilişsel yük ile problemlerin zorlukları ilişkilendirilmiştir.

1. Aşama Yöntem Önerisi: Problemlerin Zorluklarının Belirlenmesine Yönelik Önerilen Yöntem:

Problemlerin Belirlenmesi: Bir öğrencinin belirli programlama kavramlarını, yapılarını (örneğin, değişkenler, diziler, döngüler veya şartlı ifadeler) veya stratejilerini (örneğin bir ortalama bulma) uygulayacağı bir durumu sağlamak için dikkatlice tasarlanmış olan hesaplama (veya algoritmik) problemleri oluşturulmalıdır (Chao, 2016). Bu düşünceden hareketle bu çalışmada uygulama esnasında katılımcılara problem tabanlı örnekler sunulmuştur. Öğretim programına bakılarak ilgili kazanımlar seçilmiştir. Bt öğretmeni ve uzmanlar ile tartışılmıştır. Temel öğeleri ve öğelerin etkileşimlerini içeren problemler olmasına dikkat edilmiştir. Bu problemler basitten karmaşığa doğru hiyerarşik bir yapı oluşturmasına bakılmıştır.

Zorluk Derecelerinin Belirlenmesi: Alan uzmanları ile birlikte problemlerin muhtemel çözümleri yazılmış, öğrencilerin bilmeleri gereken programlama yapı ve kavramları değerlendirilmiştir. Her bir problem için temel öğeler ve etkileşim öğelerinin zorluk dereceleri belirlenmiştir. Belirlenen bu zorluk durumları ile öğrencilerin problemlerin çözümü sırasında yaşayabilecekleri bilişsel yük olarak ortaya çıkan durumlar ilişkilendirilerek sunulmuştur. Bu şekilde bir yandan programlama sürecinde oluşan bilişsel yüklerin kestirilebildiği bir yol ile ortaya konulan içsel bilişsel yük değerleri doğrulanırken diğer yandan bu şekilde belirlenen bilişsel yüklerin kaynakları ortaya konulmuştur. Öğrencilere sunulan problemlerde yer alan temel öğeler ve öğe etkileşimleri Tablo 2'de sunulmuştur.

Tablo 2. Problemlerdeki Öğeler ve Öğe Etkileşimleri

	Temel Öğeler		Öğe Etkileşimi		Öğelerin Toplam Güçlüğü
Problem	Değişken atama	D-O-Y	Değişken atama -Koşul	D-O-Y	Düşük (D) Orta (O) Yüksek (Y)
	Operatörler	D-O-Y	Değişken atama - Döngü	D-O-Y	
	Koşul	D-O-Y	Döngü - Koşul	D-O-Y	
	Döngü	D-O-Y	Fonksiyon Döngü	D-O-Y	
	Fonksiyon	D-O-Y	Fonksiyon-Değişken atama	D-O-Y	
	Sonuç	D-O-Y	Sonuç	D-O-Y	

Değişken atama: Girilen veya programın getireceği değerlerin atandığı veri tutuculardır.

Operatörler: Tek başına herhangi bir anlamı olmayan, programın işleyişine katkı sağlayan karakterlerdir.

Koşul: Programın yapacağı işlemleri bir temele dayandırmasını sağlayan yapıdır.

Döngü: Sıralı bir kod bloğunun istenilen sayıda tekrarlanmasıdır.

Fonksiyon: Tekrar kullanılabilen kod parçacığıdır.

Tablo 2'de içerikte yer alan temel öğeler, öğe etkileşimleri ve öğelerin toplam güçlüğü olmak üzere üç kategori bulunmaktadır. "Temel öğeler" kategorisi uygulama için ele alınan programlama yapılarını, "öğe etkileşimi" kategorisi bu yapıların birbiri ile etkileşim durumlarını, "öğelerin toplam güçlüğü" kategorisi ise diğer iki kategoriye göre problemin zorluğunun düşük(D), orta(O) ya da yüksek(Y) olduğu bilgisini vermektedir. Temel öğeler ve öğe etkileşimi düşük, orta ve yüksek olacak biçimde değerlendirilmiştir. Bu değerlendirmede "temel öğeler" kategorisi olarak ele alınan programlama yapıları problemler için kullanma zorunluluğuna göre farklılık göstermektedir. Örneğin, bu yapılardan herhangi biri kullanılmadan ilgili problem çözülemiyorsa bu yapının değeri o problem için yüksek olarak değerlendirilmiştir. Diğer göz önüne alınan etken ise söz konusu yapının o problem için stratejik öneme sahip olma durumudur. Örneğin problemin çözümü için koşul yapısının kullanılması zorunlu değilse ancak çözümü kolaylaştıracak stratejik bir öneme sahipse orta derecede değerlendirilirken hem kullanımı zorunlu değil hem de stratejik bir önemi yoksa düşük olarak değerlendirilmektedir. "Öğe etkileşimleri" kategorisi ise yapıların birbiriyle etkileşimlerindeki zorunluluk durumlarına ve bu etkileşimin problem için stratejik önemine göre düşük, orta ve yüksek olarak değerlendirilmiştir. Bu değerlendirmeler nicel olarak problemlerin zorluk durumunu gösterememiş olsa da problemlerin zorluklarını birbirinden ayırt edebilecek şekilde olduğu

düşünülmektedir. Bu değerlendirmeler 3 uzmanın görüşleri çerçevesinde, her bir problemin zorluk derecesi olarak belirlenmiştir.

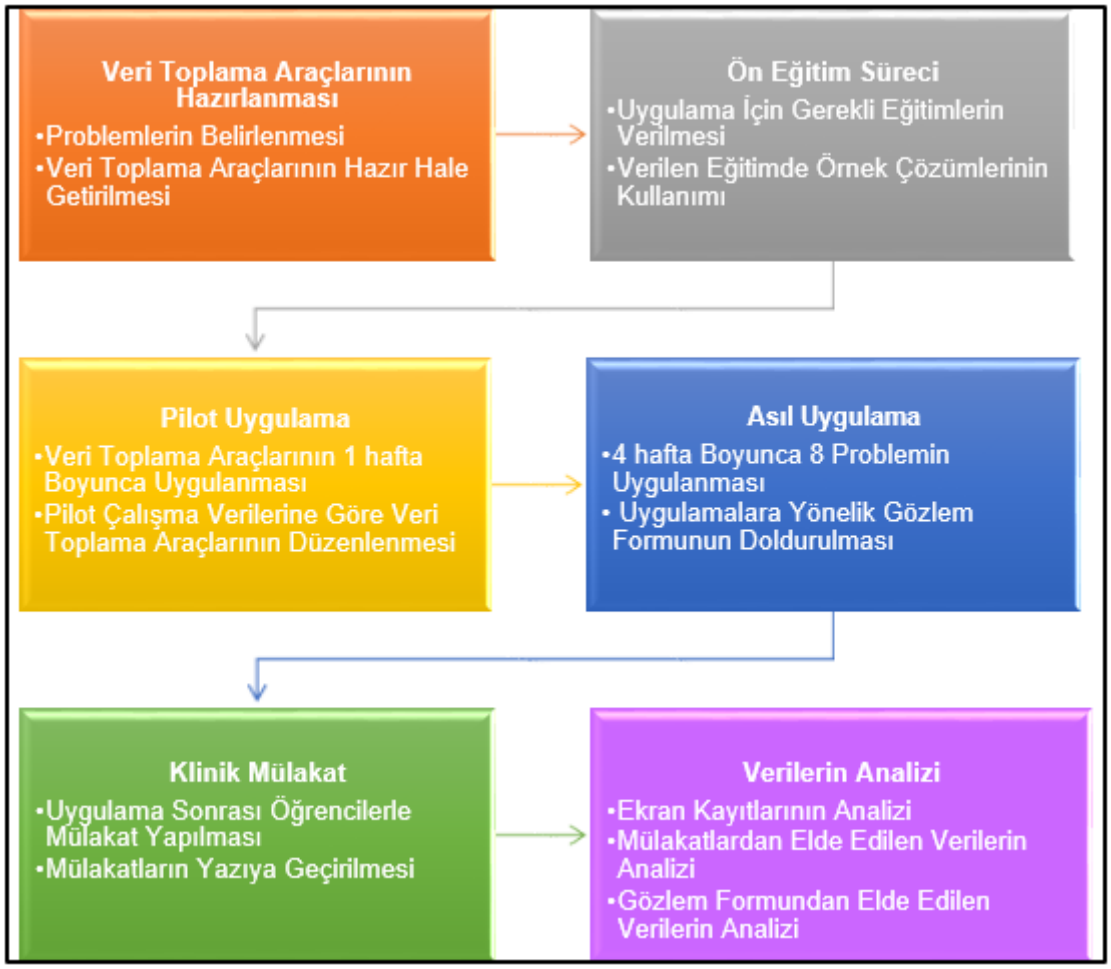
2. Aşama Yöntemin Uygulanması: Öğrencilerin Deneyimledikleri İçsel Bilişsel Yük Kaynaklarının Belirlenmesi: Öğrencilerle yapılan uygulamalardan elde edilen ekran kayıtları, beden dili hareketleri, sesli düşünmeler, araştırmacının uygulama esnasında tuttuğu gözlem formu verileri analiz edilmiş ve bu doğrultuda yapılan hatalar ve karşılaşılan zorluklar belirlenmiştir. Öğrencilerin hangi hataları daha sık yaptıklarını ve hangi zorluklarla daha fazla karşılaştıkları belirlenerek bu veriler üzerinden değerlendirmeler yapılmıştır. Hata ve zorluk türleri, öğrencilerin bu durumu giderip gidermemeleri, bu işlemi yaparken gösterdikleri çaba göz önüne alınarak kodlanmıştır. Öğrencilerin bu hata ve zorluklarla neden karşılaştıkları belirlenmeye çalışılmıştır.

3.Aşama Yöntemin Test Edilmesi: Önerilen Yöntem ile Öğrencilerin Algıladığı İçsel Bilişsel Yük İlişkisi: İlk aşamada belirlenen problemlerin zorluk durumu ile öğrencilerin algıladığı içsel bilişsel yük arasındaki ilişki incelenerek önerilen yöntemin başarılı olup olmadığı durumlar belirlenmiştir. Problemlerin zorluk durumu arttıkça sözdizimsel, anlamsal ve stratejik bilgi türlerinde yapılan hatalar ve zorlukların artış gösterme durumları değerlendirilmiştir.

3. 1. Araştırma Modeli

Bu çalışma betimleyici araştırmalar çerçevesinde ele alınmıştır. Betimleyici araştırmalar, söz konusu verilerde meydana gelen olayları ortaya koyar (Zainal, 2007). Bu araştırmalarda çalışılan olgu ya da süreç ile ilgili örneklem hakkında elde edilen veriler betimlenerek temel özellikleri tasvir edilir. Daha özel olarak, bu araştırma betimleme ile açıklamayı esas aldığı düşünüldüğünde, çalışmayı açıklayıcı/tanımlayıcı durum çalışması çerçevesinde değerlendirmek mümkündür. Açıklayıcı durum çalışmaları bir durum hakkında bilgi vermek için kullanılır (Aytaçlı, 2012; Gökçek ve Davey, 2009). İncelenen durum ayrıntılı bir şekilde araştırılarak açıklayıcı bilgiler elde edilmeye çalışılır (Zainal, 2007). Açıklayıcı durum çalışmasında daha çok araştırmanın amacı “neden” ve “nasıl” sorularını cevaplamaya çalışmaktadır (Leymun, Odabaşı, Yurdakul, 2017). Veriler nitel olarak toplanıp analiz edilmiştir. Bahar döneminde lise öğrencileriyle gerçekleştirilmiş olan uygulamada verilen problemin çözülmesi sürecinde programlama yapılarının nasıl kullanıldığı, hangi tür bilgilerin kullanıldığı bilgilerini içeren ekran kayıtları, sesli düşünme verileri ve bir gözlem formu kullanılmıştır. Bu yönüyle açıklayıcı/tanımlayıcı durum çalışmasıyla örtüşmektedir. Aynı zamanda öğrencilerin problemleri çözme esnasında sesli düşünme (Think aloud) tekniği ve ekran kaydı aracılığıyla öğrencilerden hangi davranışları neden sergilediği konusunda veriler alınmıştır. Bu araştırmada öğrencilerin

programlamada yaşadığı süreci derinlemesine incelemek amaçlandığı için küçük bir gruptaki öğrenci davranışlarını derinlemesine inceleyen bir süreç tasarlanmıştır. Patton'a (2014) göre nitel araştırmalarda örneklemin büyüklüğünden çok araştırmaya yönelik durumları içermesi önemlidir ve bu aynı zamanda araştırmacının gözlem yeteneği ile yakından ilişkilidir. Bu açıdan nitel araştırmalarda araştırmacının da sürece dahil olması elde edilenlerin geçerliliği üzerine olumlu bir etkiye sahiptir. Bütün bu bilgilerden yola çıkarak nitel bir şekilde tasarlanan bu araştırma amacına yönelik yürütülen süreç Şekil 2'de özetlenmiştir.



Şekil 2. Çalışmanın yürütme süreci

3. 2. Araştırma Grubu

Bu çalışma 2018-2019 eğitim öğretim yılında Trabzon ilinde bir lisede 9. ve 10. sınıfa devam eden, Bilgisayar Bilimi dersini alan 5 erkek öğrenci ile gerçekleştirilmiştir. Öğrenciler programlamaya karşı ilgisi olan öğrenciler arasından seçilmiş ve ilk defa bu çalışma kapsamında programlama eğitimi almışlardır. Bu eğitim dışında 9. ve 10. Sınıf

Bilgisayar Bilimi dersi kazanımlarında bulunan Python uygulamasını öğrenmekteydiler. Genel olarak ortaokulda Bilişim Teknolojileri ve Yazılım dersinin kazanımlarında temel bilgisayar kullanımına yönelik becerileri edindikleri için öğrencilerin bilgisayar kullanım seviyeleri birbirine yakın ve çalışmayı yürütebilecek düzeyde olduğu varsayılmıştır.

3. 3. Verilerin Toplanması

Nitel çalışmalarda esas olan insan davranışlarının en iyi şekilde gözlemlenerek yorumlanması, süreç boyunca farklı veri toplama araçları kullanılarak elde edilen verilerin birçok kanaldan doğrulanabilmesi özel durum çalışmalarında tercih edilen bir uygulama şeklidir. Bu çerçevede veriler uygulama süreci boyunca ekran kayıtları, gözlemler ve bunlara ilişkin mülakatlar ile toplanmıştır. Aşağıda bu araçlar tanıtılmaktadır.

3. 3. 1. Veri Toplama Araçları

1. *Gözlem formu:* Araştırmacı tarafından alan uzmanı görüşleri doğrultusunda geliştirilmiş olan form uygulama süreci boyunca kullanılmış ve katılımcının programlama bilgi türleri içerisinde hangi kaynaklarda zorlandığı tespit edilmeye çalışılmıştır. Uygulama esnasında katılımcılar çözmesi gereken problem karşısında sesli düşünme, beden dili, jest ve mimikleri ile birtakım tepkiler vermişlerdir. Bu davranışlar gözlem formu aracılığı ile toplanmıştır.
2. *Sesli düşünme (Think aloud):* Çözüm sırasında hangi davranışın neden sergilendiği gibi birtakım davranışlara yönelik veri almak için uygulama süreci boyunca kullanılmıştır. Katılımcı problem çözümü sırasında birtakım hatalarla karşılaşmıştır. Bu hatalara verdiği sesli tepkiler ve bu esnada yöneltilen sorulara verdiği cevaplar sesli düşünme tekniği ile incelenmiştir. Örneğin katılımcı gideremediği bir hata karşısında “Of ya neden olmadı.” şeklinde sesli düşünmüştür.
3. *Ekran kaydı:* Katılımcının kullanılacak program içerisinde yaptığı işlemleri görebilmek ve ilgili durumları belirlemek için uygulama süreci boyunca kullanılmıştır. Bu süreç programlama öğretimi uygulamalarından Python uygulaması üzerinde Camtasia programıyla kayda alınmıştır. Ekran kayıtları incelenmesiyle öğrencilerin sorun yaşadığı noktalar belirlenerek, ilgili durumların nedenlerine yönelik klinik mülakatlar yapılmıştır.

Bu şekilde elde edilen veriler, birlikte analiz edilerek programlamadaki bilgi türleri ve programlama yapıları çerçevesinde bilişsel yük oluşturan kaynaklar sınıflandırılarak sunulmuştur.

3. 3. 2. Veri Toplama Süreci

Araştırmanın uygulama süreci 5 hafta devam etmiştir. Bu süreçte uygulama boyunca öğrencilere giderek karmaşıklaşan problemler verilmiştir. Problemi çözme aşamasında öğrencilerin beden dilleri ve ekranda yapmış olduğu işlemler kayda alınmış ve bu kayıtlar veri olarak kullanılmıştır. Süreçte öğrencinin sesli düşünmesi sağlanmış ve sürecin belirli aşamalarında öğrenciye birtakım sorular yöneltilmiştir. Böylelikle yapılan davranışların nedenleri ile ilgili veri toplanmıştır.

3. 3. 2. 1. Ön Eğitim Süreci

Bu süreç uygulamaya başlamadan önce öğrencilerle yapılan eğitimi içermektedir. Eğitim, Bilgisayar Bilimi dersi öğretim programı içerisinde bulunan Python programı kazanımları doğrultusunda gerçekleştirilmiştir. Bu süreçte, ele alınan programlama yapıları (değişken atama-operatörler-koşul-döngü-fonksiyon) anlatılmış ve örnekler çözülmüştür. Asıl uygulama esnasında kullanılacak olan sesli düşünme tekniğinden alınacak verilerin güvenirliliği için bu süreçte öğrencilerden örnek çözümünde yaptıklarını anlatarak gerçekleştirmeleri istenmiştir. Eğitim 3 ay boyunca devam etmiştir. Araştırmacı aynı zamanda Bilişim Teknolojileri öğretmeni olduğundan öğretici olarak eğitimi gerçekleştirmiştir. Tüm katılımcılar aynı zamanda ve aynı ortamda bu eğitimi almışlardır.

3. 3. 2. 2. Pilot Uygulama

Bu süreç 7 öğrenci ile yürütülmüştür. Uygulama sonunda araştırmanın detaylı yürütülebilmesi için 5 öğrenci ile devam etme kararı alınmıştır. Uygulamanın yapılacağı ortam incelenmiş ve gerekli düzenlemeler yapılmıştır. Süreç sonunda katılımcıların boy ve yaşlarına uygun masa-sandalye seçimi yapılmıştır. Ekran kaydı için uygun program indirilmiştir. Bir problem çözümü için ortalama süre belirlenmeye çalışılmıştır. Veri toplama araçlarının kullanışlılığına bakılmış, belirlenen amaçlar doğrultusunda kullanılıp kullanılmayacakları test edilmiş ve asıl uygulamaya hazır hale getirilmiştir.

3. 3. 2. 3. Asıl Uygulama

Bu süreç 5 öğrenci ile 4 hafta boyunca yürütülmüştür. Haftada 2 kez farklı günler olması sağlanmış ve her katılımcıya gün ve saat belirlenmiştir. Herhangi bir dış etki olmaması açısından her katılımcının uygulaması tek yapılmıştır. Masa-sandalye, pano ve bilgisayarın olduğu karmaşıklıktan uzak, bir ortam hazırlanmıştır. Katılımcılara istedikleri süre verilmiş, süre açısından bir sınırlama yapılmamıştır. Ön eğitim sürecinde gerçekleşen

örnek çözümleri sırasında öğrencilerden ne yaptıklarını sesli bir şekilde anlatma uygulamaları asıl uygulama sürecinde de devam etmiştir.

3. 3. 2. 4. Araştırmacının Rolü

Araştırmacı süreç boyunca öğretici rolünde olmuştur. Öğrencilerle eğitim ve uygulama dışında da sık sık biraraya gelerek samimi bir ortam ve sağlıklı iletişim sağlamaya çalışmıştır. Böylelikle öğrencilerin sürece adapte olmaları kolaylaşmıştır. Bu durum güvenilir veri elde etmek için gerekli görülmüştür. Süreç içerisinde öğrencilerin dış etkenlere maruz kalması önlenmeye çalışılmış ve öğrenciler süreçte serbest bırakılmıştır.

3. 3. 2. 5. Gözlemler

Gözlemler uygulayıcı olan araştırmacı tarafından gerçekleştirilmiştir. Uygulama esnasında gözlemci her zaman katılımcının yanında olmuştur. Ekranda yaptığı işlemleri, beden dili, jest ve mimikleri, sesli düşünmesi gözlemlenmiştir. Uygulamanın bazı bölümlerinde sorular yöneltmiş ve cevap almaya çalışmıştır. Elde edilen veriler gözlem formuna eklenmiştir. Gözlem esnasında araştırmacı, öğrencinin düşüncelerine müdahale etmeden ortaya çıkan durumları incelemeye çalışmıştır.

3. 3. 2. 6. Klinik Mülakat

Klinik mülakatlar yapılan uygulamaların tamamlanmasının ardından öğrenciler ile soru-cevap şeklinde gerçekleştirilmiştir. “Karşılaştığın … hatanın nedeni ne olabilir, en çok hangi aşamada zorlandın? ” gibi sorulara yer verilmiş ve bu sorular genişletilerek alınan cevaplar not edilmiştir. Bu sorular uzman görüşüne sunularak son şekli verilmiştir. Mülakatlar yaklaşık 8-10 dk. sürmüştür. Mülakat esnasında öğrencilerden istenilen, uygulamada verilen görevleri yerine getirirken göstermiş oldukları davranışları neden yaptıklarına dair sorulan temel soruları cevaplandırmaları şeklinde olmuştur. Problem çözümü esnasında yapmış olduğu davranışların nedenleriyle ilgili bilgiler alınarak öğrencinin problemün çözümü üzerindeki düşünceleri tespit edilmeye çalışılmıştır.

3. 4. Verilerin Analizi

Çalışmada ekran kayıtları, gözlem formu, sesli düşünme verileri ve mülakatlar; birlikte değerlendirilerek analiz edilmiştir. Mülakatlar, beden dili hareketleri ve sesli düşünme tekniği ile elde edilen veriler gözlem formuna işlenmiştir. Ekran kayıtları analiz

edilerek alınan veriler diğer kaynaklardan alınan verilerle birleştirilmiş ve açıklanarak sunulmuştur.

3. 4. 1. Gözlem Formundan Elde Edilen Verilerin Analizi

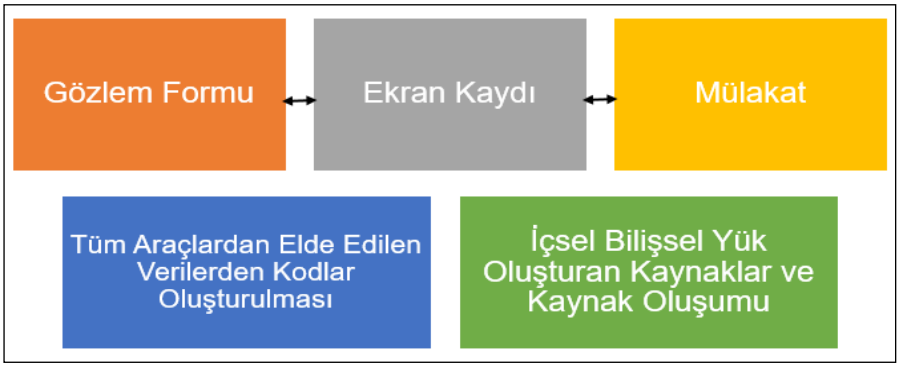
Gözlem formu içerisinde öğrencilerin problemleri çözme esnasında neyi, neden ve nasıl yaptığına dair sesli düşünmesi ve beden dili hareketlerini içeren gözlemci verilerini içermektedir. Bu formdan elde edilen veriler karşılaşılan hata ve zorluk türlerinin giderilip giderilememe esnasında harcadıkları çabayı yorumlamak için kullanılmıştır.

3. 4. 2. Ekran Kaydından Elde Edilen Verilerin Analizi

Uygulama esnasında öğrencilerin bilgisayar üzerinde yapmış olduğu her türlü davranış kayda alınmıştır. Bu kayıtlar alan uzmanı tarafından derinlemesine incelenmiş ve elde edilen veriler, diğer veri toplama araçlarından elde edilenlerle desteklenerek değerlendirilmiştir. Uygulama sürecinde zorluk durumuna göre belirlenen problemler basitten karmaşığa sunulmuştur. Elde edilen veriler bu sıralama dikkate alınarak incelenmiş ve değerlendirilmiştir.

3. 4. 3. Mülakattan Elde Edilen Verilerin Analizi

Öğrencilerle her uygulama sonrası soru-cevap şeklinde yapılmıştır. Her mülakat ortalama 10 dk. sürmüştür. “En çok hangi aşamada zorlandın?” sorusu çerçevesinde genişletilerek gerçekleştirilmiştir. Gözlem formundan elde edilen verilere destekleyici nitelikte olmuştur. Çalışmada kullanılan tüm veri toplama araçlarının analiz süreci Şekil 3’te özetlenmiştir.



Şekil 3. Veri toplama analiz süreci

3. 5. Araştırmanın Geçerliliği ve Güvenirliliği

Geçerlik ve güvenirlik elde edilen sonuçların inandırıcılığı açısından araştırmalarda en yaygın kullanılan iki ölçüttür (Başkale, 2016). Joppe'ye (2000) göre geçerlilik, araştırmanın ölçmek istediğini gerçekten ölçüp ölçmeyeceğini veya araştırma sonuçlarının ne kadar doğru olduğunu belirler. Bu anlamda araştırmacının araştırdığı olguyu olabildiğince ayrıntılı ve yansız sunması önemlidir.

Yıldırım ve Şimşek (2013) güvenirlik kavramının nitel araştırma için farklı bir anlamı olduğunu ifade etmiştir. Nitel araştırmalarda niteliği arttırabilmek için inandırıcılık, aktarılabilirlik, tutarlık, teyit edilebilirlik gibi kriterler belirlenmiştir(Lincoln ve Guba, 1982). Bu unsurlara yönelik tedbirler alınması araştırma için önemlidir. Bu çerçevede bu özelliklerin bu araştırmada alınan tedbirler aşağıda özetlenmektedir.

İnandırıcılık: Araştırma sürecinde birçok veri toplama aracı birbirini destekler nitelikte işe koşulmuştur. Çalışma grubu ile araştırmacının çalışma sürecinde uzun süre boyunca birlikte olması, ayrıntılı bir şekilde veri toplanabilmesini sağlamıştır.

Uzun Süreli Uygulama: Araştırma 3 ay eğitim süreci, 1 hafta pilot uygulama ve 4 hafta boyunca haftada 2 problem olmak üzere 8 problemle yürütülmüştür. Sürenin yeterince uzun olması süreç boyunca öğrencilerin davranışlarının oluşturduğu örüntülerin belirlenmesine katkı sağlamıştır.

Aktarılabilirlik: Çalışma grubunun özelliklerine ilişkin bilgiler ve araştırma sürecinde problemlerin çözümlerinde öğrenci davranışlarının detaylarının elde edilebilmesi aktarılabilirlik çerçevesinde önemli rol oynamıştır. Ayrıca, araştırmanın kuramsal çerçevesine yönelik detaylı açıklamalar ve kullanılan yapı ve yöntemlerin kuramsal çerçeve ile ilişkisinin sunumu araştırmanın aktarılabilir olma özelliklerine işaret eden diğer bir durum olarak görülebilir.

Teyit Edilebilirlik: Araştırmada veriler farklı formattaki veri kaynaklarından toplanmıştır. Bu durum yorumlamayı kolaylaştırmış ve verilerin teyit edilebilir olma durumunu güçlendirmiştir.

Tutarlılık: Veri toplama esnasında ekranda yapılan uygulamalar, öğrencilerin beden dili hareketleri, sesli düşünmeleri konusunda veri kaybına yol açmamak için süreç kaydedilmiştir. Ayrıca verilerin analizinde farklı uzmanlardan yardım alınmıştır. Özellikle beden dili, jest ve mimik kullanımının uzmanlar tarafından değerlendirilmesi araştırmanın tutarlılığına işaret eden bir özellik olarak düşünülebilir.

4. BULGULAR

Programlamada yaşanan bilişsel süreçlerdeki içsel bilişsel yük oluşturan kaynakları araştıran çalışmanın bu bölümünde; belirlenen problemlerin zorluk durumu, katılımcılar tarafından algılanan bilişsel yük, gözlemci verileri, ekran kaydı, sesli düşünme ve beden dili hareketlerinden elde edilen bulgulara yer verilmiştir. Çalışmanın bu bölümü için aşağıda açıklanan tanımları bilmek önemlidir.

Zorluk: Problemleri çözerken sıkıntıyla, güçlükle yapma durumu.

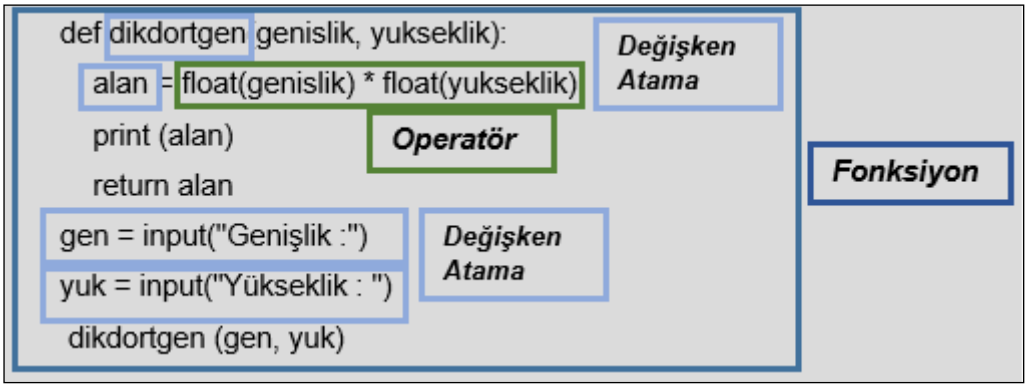
Hata: Problemlerin çözümünde yapılan yanlışlar, yanılmadan doğan durum.

4. 1. Önerilen Yöntem Doğrultusunda Problemlerin Zorluk Durumu

Bu çalışmada problemlerin zorluk dereceleri her bir problem için potansiyel içsel bilişsel yük oluşturma durumu olarak düşünülmüş, öğrencilerin problemleri çözme sürecinde karşılaşmış oldukları bilişsel yükler uzmanlarca belirlenen içsel yük ile ilişkilendirilerek, bilişsel yük kaynakları olarak problem temelinde sunulmuştur. Problemler içerdikleri temel öğelerin ve bu öğelerin etkileşimlerinin farklı kombinasyonlar oluşturmasından kaynaklanmaktadır.

Problem 1**:** *Kullanıcının girdiği verilere göre dikdörtgenin alanını hesaplayan programı fonksiyon kullanarak yazınız.*

Problemin muhtemel çözümü üzerinde alan uzmanlarının belirledikleri zorluk durumları Şekil 4'te sunulmuştur.



Şekil 4. Problem 1'in içerdiği temel öğeler

Özetle; problem 1'de kullanılan temel öğeler ve öğelerin etkileşim durumları Tablo 3'te gösterilmiştir.

Tablo 3. Problem 1 Temel Öğeler ve Öğe Etkileşim Durumu

Temel öğeler	Değişken Atama	Operatör	Fonksiyon
Değişken Atama		√	√ √
Operatör	√		√
Fonksiyon	√ √	√	

Tablo 3 incelendiğinde 1. Problemde değişken atama-operatör yapıları arasında 1, değişken atama-fonksiyon yapıları arasında 2, operatör-fonksiyon yapıları arasında 1 olmak üzere 4 adet etkileşim söz konusu olduğu belirlenmiştir.

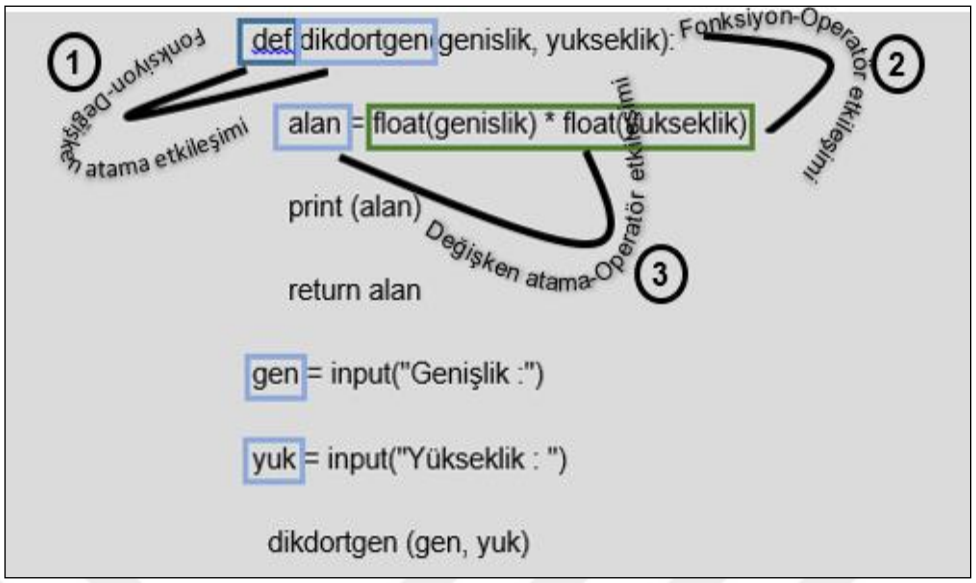
Problem 1 için kullanılması gereken yapılar ve kullanılma durumları aşağıda açıklanmıştır.

Değişken Atama: Bu problemin çözülmesi için değişken atamanın yapılması zorunludur. Problemin fonksiyon kullanılarak yapılması istendiği için ilk olarak fonksiyon içerisinde bir değişken tanımlanmıştır(dikdortgen). Dikdörtgenin alan hesaplaması için genişlik ve yükseklik değerleri gereklidir. Problem cümlesinde bu değerlerin kullanıcının girmesi istenmektedir. Bu iki değişken input komutu yardımıyla kullanıcıdan istenmiştir (gen, yuk). Bu değişkenlerin dışında hesaplamanın yapılıp sonucun atanması için bir değişken (alan) daha tanımlanmıştır. Böylelikle toplamda dört adet değişken tanımlandığı görülmektedir.

Operatör: Problem, dikdörtgenin alan hesaplamasını içermektedir. Dolayısıyla matematiksel birtakım işlemler işe koşulmak zorundadır. Şekil 4'te görüldüğü üzere problemin çözümünde koşul içerisinde operatör(float(genislik)*float(yukseklik)) kullanılması gereklidir. Dolayısıyla bu problem için operatör kullanılması zorunludur.

Fonksiyon: Bu problemin çözümü için fonksiyon kullanımı zorunlu değildir. Ancak problem cümlesine bakıldığında bu problemin fonksiyon kullanılarak yapılması istenmektedir. Problemin muhtemel çözümü değerlendirildiğinde ilk olarak bir fonksiyon(def) tanımlandığı ve bu durumun orta düzeyde zorluk oluşturacağı değerlendirilmiştir.

Problem 1'de etkileşime giren öğeler ve öğe etkileşim durumları Şekil 5'te gösterilmiştir.



Şekil 5. Problem 1 muhtemel çözüm üzerinde öğe etkileşimleri

1.Etkileşim(Fonksiyon-Değişken atama): Bu problemin çözümü için yazılacak programda fonksiyon kullanılması istenmektedir. Fonksiyon komutu(def) kullanıldıktan sonra fonksiyon satırında bir değişken(dikdörtgen) tanımlanmıştır. Fonksiyonu çağırma işleminde (dikdortgen(gen,yuk)) hem bu değişken hem de kullanıcıdan istenen iki değişken(gen,yuk) kullanılacaktır. Aynı zamanda fonksiyon satırından sonra tanımlanan değişkenlerle(alan) birlikte bu problemde *fonksiyon-değişken atama etkileşimi* gerçekleşmiştir.

2.Etkileşim(Fonksiyon-Operatör): Fonksiyon içerisinde dikdörtgenin alan hesaplama formülü(alan=float(genislik)*float(yükseklik)) kullanılmıştır. Fonksiyon içerisinde operatör kullanımı dolaylı da olsa, fonksiyon-operatör etkileşimini gerektirmiştir. Bu çerçevede bu etkileşim “düşük” düzeyde zorluk içeren bir öğe etkileşimi olarak değerlendirilmiştir.

3.Etkileşim(Değişken atama-Operatör): Bir değişken(alan) tanımlanarak dikdörtgenin alan hesaplama formülü(alan=float(genislik)*float(yükseklik)) bu değişkene atanması gereklidir. Dolayısıyla *değişken atama-operatör etkileşimi* yüksek düzeyde değerlendirilmiştir.

Temel öğeler ve öğeler arası etkileşimler temel alındığında 1. Problemin zorluk durumu Tablo 4’te gösterilmektedir.

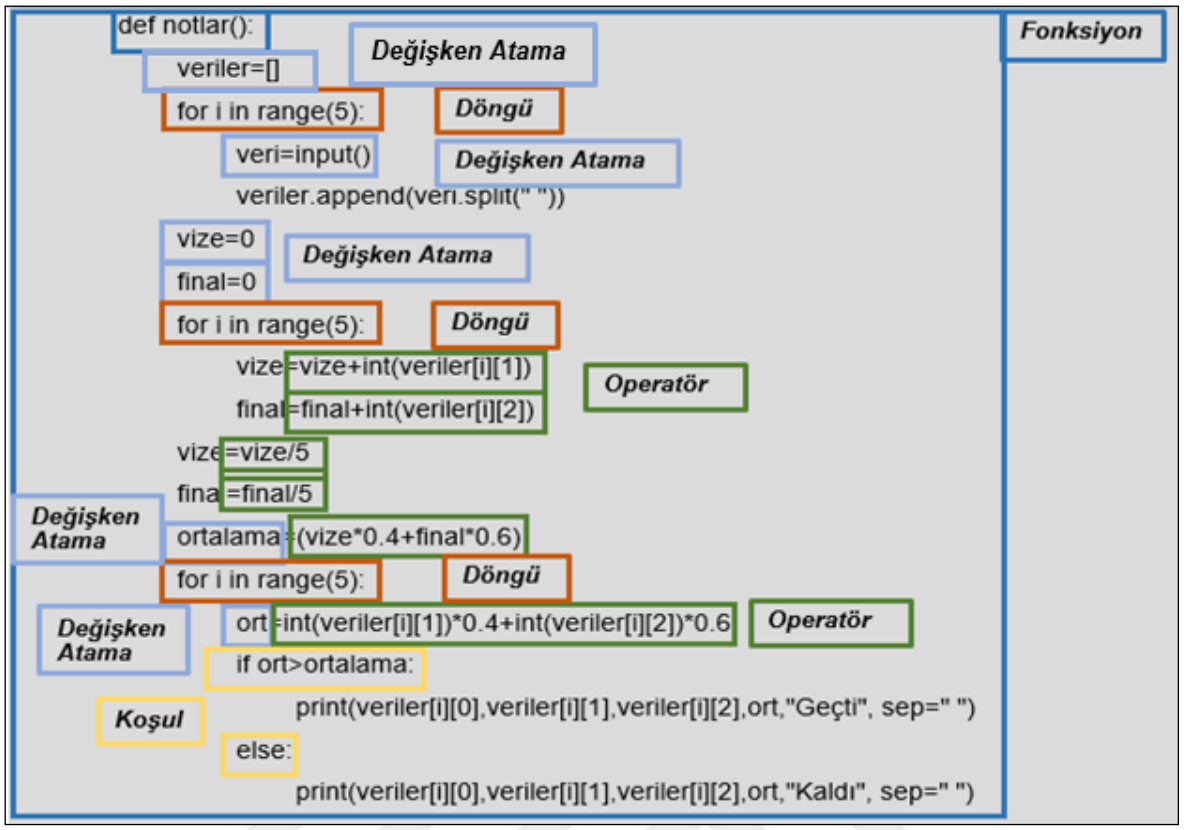
Tablo 4. Problem 1. Zorluk Durumu

	Temel Öğeler		Öğe Etkileşimi		Öğelerin Toplam Güçlüğü
Problem 1	Değişken atama	Y	Değişken atama – Fonksiyon	Y	Yüksek (Y)
	Operatörler	Y	Fonksiyon-Operatör	D	
	Koşul	-	Değişken atama-Operatör	Y	
	Döngü	-			
	Fonksiyon	O			
	Sonuç	Y	Sonuç	Y	

Tablo 4 incelendiğinde 1.problem için değişken atama ve operatörlerin kullanımının zorunlu olduğu görülmektedir. Bu nedenle değişken atama ve operatör yapıları yüksek olarak değerlendirilmiştir. Alan uzmanları tarafından bu problemin fonksiyon kullanılmadan da çözülebileceği değerlendirilmiştir. Dolayısıyla problemin çözümündeki fonksiyon yapısı *temel öğeler* kategorisinde orta derecede zorlukta değerlendirilmiştir. Kullanılan tüm temel öğeler düşünüldüğünde bu problemdeki temel öğelerin genel olarak "yüksek" düzeyde zorluk içerdiği değerlendirilmiştir. Bu problemde kullanılan bu üç yapının birbirleriyle etkileşme durumları *öğe etkileşimleri* kategorisinde yer almaktadır. Söz konusu öğelerin birbirleriyle etkileşme durumları düşünüldüğünde *değişken atama-fonksiyon etkileşimi* ve *değişken atama-operatör etkileşiminin* yüksek olarak *fonksiyon-operatör etkileşiminin* düşük olarak değerlendirildiği görülmektedir. Değişken atama yapısının kullanılan diğer yapılarla etkileşimlerine yüksek değer verilmesinin nedeni problemin muhtemel çözümünde de gösterildiği üzere bu yapıların birbirleriyle etkileşmeden problemin çözümünün yapılamayacağıdır. Fonksiyon-operatör etkileşiminin düşük olarak değerlendirilmesi bu iki yapının birbirleriyle doğrudan etkileşmediğinden kaynaklandığı belirlenmiştir. *Temel öğeler* ve *öğe etkileşiminde ise* bu problemin zorluk derecesi yüksek olarak değerlendirilmiştir.

Problem 2**:** *Kullanıcıdan tek satırda öğrenci adı, vize notu, final notu alınacaktır. Bu şekilde 5 öğrenci için kullanıcıdan veri alıp sınıf not ortalamasını hesaplayınız (vize*0.4+final*0.6) ve her öğrencinin adını, vize notunu, final notunu, ortalamasını ve sınıf ortalamasına göre geçip geçmediğini hesaplayan programı fonksiyon kullanarak yazınız.*

Problemin muhtemel çözümü üzerinde alan uzmanlarının belirledikleri zorluk durumları Şekil 6'da sunulmuştur.



Şekil 6. Problem 2'nin içerdiği temel öğeler

Özetle; problem 2'de kullanılan temel öğeler ve öğelerin etkileşim durumları Tablo 5'te gösterilmiştir.

Tablo 5. Problem 2 Temel Öğeler ve Öğe Etkileşim Durumu

Temel öğeler	Değişken Atama	Operatör	Koşul	Döngü	Fonksiyon
Değişken Atama		√ √ √ √ √ √	√ √	√ √ √	√
Operatör	√ √ √ √ √ √		√	√ √ √	-
Koşul	√ √	√		-	-
Döngü	√ √ √	√ √ √	-		-
Fonksiyon	√	-	-	-	

Tablo 5 incelendiğinde değişken atama, operatör, koşul, döngü ve fonksiyon olmak üzere ele alınan tüm yapıların kullanıldığı görülmektedir. Etkileşimlere bakıldığında değişken atama-operatör yapıları arasında 6, değişken atama-koşul yapıları arasında 2, değişken atama-döngü yapıları arasında 3, değişken atama-fonksiyon yapıları arasında 1, koşul-operatör yapıları arasında 1, koşul-döngü yapıları arasında 3 olmak üzere 16 adet etkileşim söz konusudur.

Problem 2 için kullanılması gereken yapılar ve kullanılma durumları aşağıda açıklanmıştır.

Değişken Atama: Bu problemin çözümünde kullanıcıdan alınacak veriler(veriler) için bir değişken tanımlanmıştır. Problem vize ve final notlarıyla işlem yapılması gerektiğini belirtmektedir. Bu nedenle iki değişken(vize, final) tanımlanarak sıfırlanmıştır. Bu iki değişkenle(vize, final) yapılan işlemleri atamak için bir değişken(ortalama) daha tanımlanmıştır. Son olarak kullanıcıdan alınan verileri aktarmak için bir değişken(ort) atanmıştır. Böylelikle bu problemin çözümü için birden fazla değişken ataması yapılmıştır.

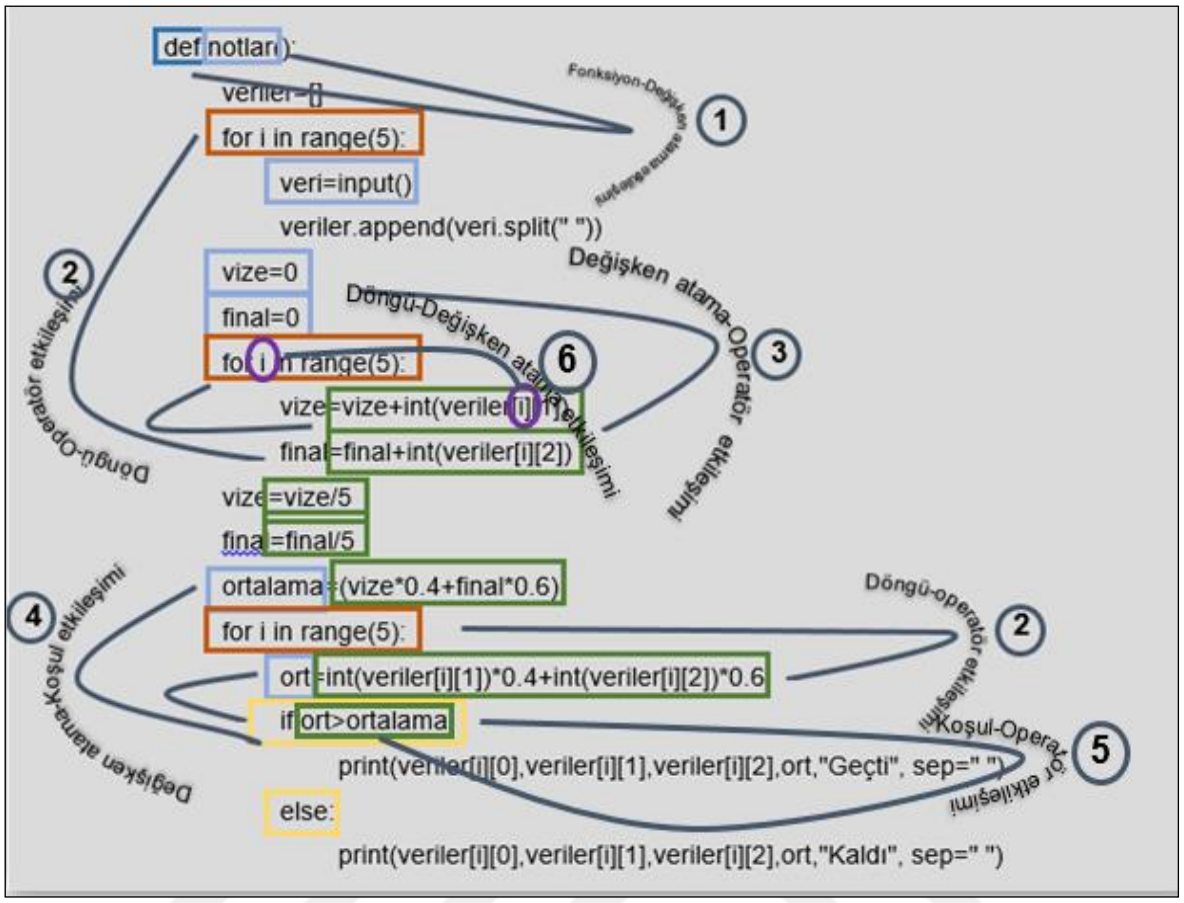
Operatör: Problem, kullanıcıdan alınan verilerle sınıf ortalamasının hesaplanmasını istemektedir. Dolayısıyla matematiksel birtakım işlemlerin işe koşulması gerekir. Değişkenlerin işlemler yapılarak farklı değişkenlere atanmasında ve koşul(if) yapısı içerisinde operatör kullanıldığı görülmektedir. Bu problemin çözümü için operatör kullanımı gereklidir.

Koşul: Problem içerisinde belirlenen değerin ortalamadan büyük olup olmamasına göre geçti ya da kaldı kararı verilecektir. Bu kararın verilmesi aşamasında bir koşul söz konusudur. Dolayısıyla koşul yapısının bu problem için kullanılması gereklidir.

Döngü: Bu problemde 5 kullanıcıdan veri almak için döngü(for) kullanılmıştır. Döngü kullanılarak, yapılacak işlem kısaltılmıştır. Ancak bu problem döngü kullanılmadan da yapılabileceğinden dolayı döngü yapısı orta olarak değerlendirilmiştir.

Fonksiyon: Fonksiyon yapısı bu problemin çözümü için zorunlu olmasa da problemin çözümünün fonksiyon kullanılarak yapılması istenmektedir. Muhtemel çözüme bakıldığında ilk olarak bir fonksiyon(def) tanımlandığı görülmektedir.

Problem 2' de etkileşime giren öğeler ve öğe etkileşim durumları *Şekil 7*'de açıklanmıştır.



Şekil 7. Problem 2 muhtemel çözüm üzerindeki öğe etkileşimleri

1.Etkileşim(Fonksiyon-Değişken): Bu problemin fonksiyon kullanılarak yapılması istenmektedir. Fonksiyon komutu(def) kullanıldıktan sonra fonksiyon satırında bir değişken (notlar) tanımlanmıştır. Fonksiyon satırından sonra tanımlanan değişkenlerle birlikte bu problemde *fonksiyon-değişken atama etkileşimi* gerçekleşmiştir.

2. Etkileşim (Döngü-Operatör): Döngü kullanılarak kullanıcıdan alınan veriler yine döngü komutu altında birtakım matematiksel işlemler içermektedir. Yukarıdaki görselde belirtildiği üzere bu etkileşimin 2 adet olduğu görülmektedir.

3. Etkileşim (Değişken atama-Operatör): Kullanıcıdan veri almak için tanımlanan değişkenler aritmetik işlemler yardımıyla farklı bir değişkene atanmıştır. Böylelikle bu aşamada *değişken atama-operatör etkileşimi* olduğu görülmektedir.

4. Etkileşim (Değişken atama-Koşul): Sınıf ortalamasını bulmak için tanımlanan değişken, koşul komutu içerisinde kullanılmıştır. Bu problemde *değişken atama-koşul* yapıları arasında bir etkileşim olduğu görülmektedir.

5. Etkileşim (Operatör-Koşul): Koşul(if) içerisinde bir karşılaştırma(ort<ortalama) kontrol edilmek istenmiştir. Bu karşılaştırma için operatör kullanılması gereklidir. Bu

operatörün koşul(if) içerisinde kullanılması *operatör- koşul etkileşiminin* gerçekleşmesini sağlamıştır.

6. Etkileşim (Değişken atama-Döngü): Döngü(for) içerisinde tanımlanan değişken(i), bir alt satırda değişken atama(vize=vize+int(veriler[i]..)) kısmında kullanılmıştır. Böylelikle *değişken atama-döngü etkileşimi* gerçekleşmiştir.

Temel öğeler ve öğeler arası etkileşimler temel alındığında 2. Problemin zorluk durumu Tablo 6'da gösterilmektedir.

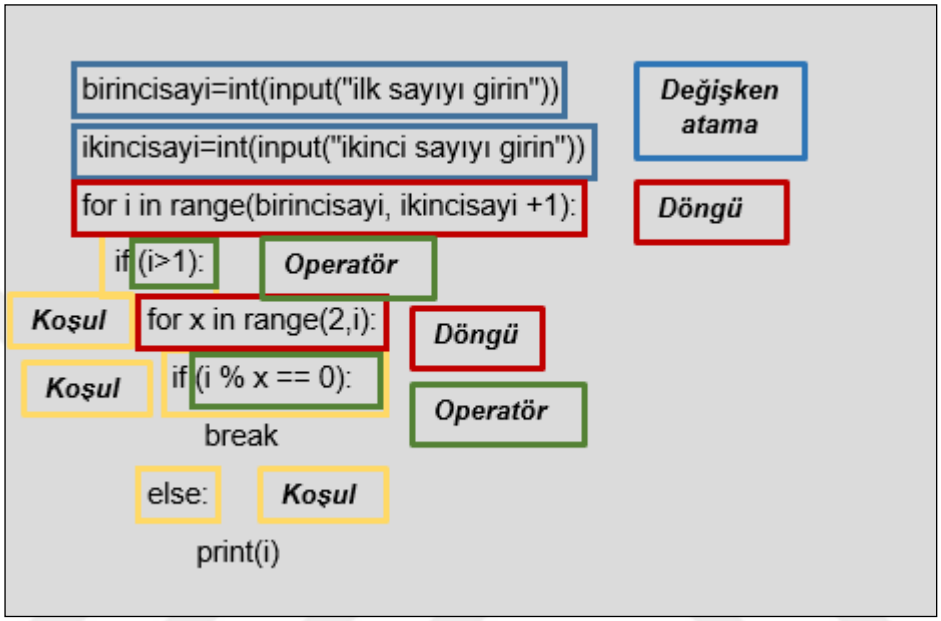
Tablo 6. Problem 2. Zorluk Durumu

	Temel Öğeler		Öğe Etkileşimi		Öğelerin Toplam Güçlüğü
Problem 2	Değişken atama	Y	Koşul-Operatör	Y	Yüksek (Y)
	Operatörler	Y	Döngü - Koşul	D	
	Koşul	Y	Değişken atama-Operatör	Y	
	Döngü	O	Döngü-Değişken atama	Y	
	Fonksiyon	O	Fonksiyon-Diğer Yapılar	D	
			Değişken atama-Koşul	Y	
			Operatör- Döngü	Y	
	Sonuç	Y	Sonuç	Y	

Tablo 6 incelendiğinde 2.problem için değişken atama, operatör ve koşul yapılarının kullanımının zorunlu olduğu görülmektedir. Bu nedenle bu yapılar yüksek olarak değerlendirilmiştir. Problem döngü yapısı kullanılmadan da çözülebilmektedir. Döngü kullanılarak, yapılacak işlemler kısaltılmıştır. Bu nedenle döngü yapısı temel öğeler kategorisinde orta derecede değerlendirilmiştir. Bu problemin fonksiyon kullanılmadan da çözülebileceği alan uzmanları tarafından belirlenmiştir. Ancak problemin fonksiyon kullanılarak yapılması istenmektedir. Dolayısıyla fonksiyon yapısı temel öğeler kategorisinde orta olarak değerlendirilmiştir. Yapılara verilen derecelendirmeler doğrultusunda temel öğeler kategorisi alan uzmanı tarafından yüksek derecede olarak belirlenmiştir. Kullanılan yapıların birbirleriyle etkileşme durumlarına bakıldığında *koşul-operatör, değişken atama-operatör, döngü-değişken atama, değişken atama-koşul, döngü-operatör etkileşiminin* yüksek derecede olduğu görülmektedir. Bu etkileşimlere yüksek değer verilmesinin nedeni problemin muhtemel çözümünde de gösterildiği üzere yapıların birbirleriyle etkileşmeden problemin çözümünün yapılamayacağıdır. Etkileşimler arasında *döngü-koşul* ve *fonksiyonun-diğer yapılarla* etkileşimi düşük olarak değerlendirilmiştir. Öğe etkileşimi kategorisi sonuç olarak yüksek değerdedir. Temel öğeler ve etkileşim öğeleri sonuçlarına bakıldığında bu problemin zorluk derecesinin yüksek olduğu belirtilmiştir.

Problem 3**:** *Kullanıcının girdiği 2 sayı arasındaki asal sayıları bulan programı yazınız.*

Problemin muhtemel çözümü üzerinde alan uzmanlarının belirledikleri zorluk durumları Şekil 8'de sunulmuştur.



Şekil 8. Problem 3'ün içerdiği temel öğeler

Özetle; problem 3'te kullanılan temel öğeler ve öğelerin etkileşim durumları Tablo 7'de gösterilmiştir.

Tablo 7. Problem 3 Temel Öğeler ve Öğe Etkileşim Durumu

Temel Öğeler	Değişken Atama	Operatör	Koşul	Döngü
Değişken Atama		-	-	√ √
Operatör	-		√ √	√ √ √
Koşul	-	√ √		√ √ √
Döngü	√ √	√ √ √	√ √ √	

Tablo 7 incelendiğinde değişken atama, operatör, koşul ve döngü olmak üzere ele alınan yapılardan 4 adedinin kullanıldığı görülmektedir. Etkileşimlere bakıldığında değişken atama-döngü yapıları arasında 2, koşul-operatör yapıları arasında 2, koşul-döngü yapıları arasında 3, operatör-döngü yapıları arasında 3 olmak üzere 10 adet etkileşim söz konusudur.

Problem 3 için kullanılması gereken yapılar ve kullanılma durumları aşağıda açıklanmıştır.

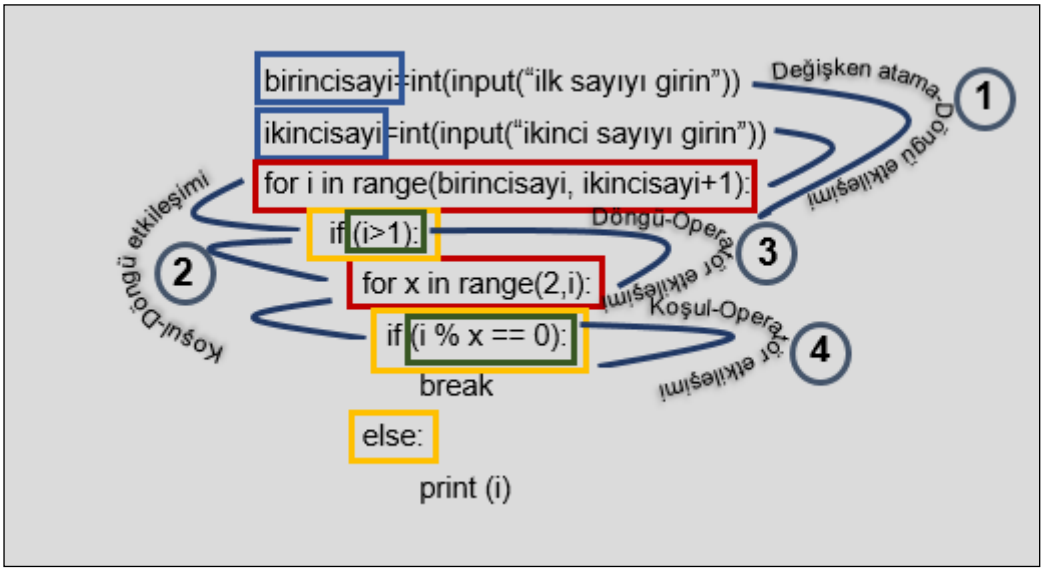
Değişken atama: Bu problem için kullanıcıdan iki sayı girmesi istenmiştir. Bu sayıları atamak için iki değişken (birincisayi, ikincisayi) tanımlanmıştır. Değişken tanımlama satırındaki int komutu girilecek sayının tam sayı olması gerektiğini, input ise bu sayıların kullanıcı tarafından girilmesi gerektiğini belirtmek için kullanılmıştır.

Operatör: Her iki koşul satırı içerisinde birtakım karşılaştırmalar yapmak için operatörler (i>1,i%x==0) kullanılmıştır. Problemin doğru çözümü için bu yapının kullanılması gereklidir.

Koşul: Problemin muhtemel çözümünde toplamda üç adet koşul(if, else) yapısı kullanıldığı görülmektedir. Problem içerisinde birtakım karşılaştırmalar yapmak ve şart sağlanıyorsa yapılması gereken işlemleri işe sokmak için bu yapılar kullanılmıştır.

Döngü: Bu problem için iki adet döngü(for) kullanıldığı görülmektedir. Bunlardan ilki kullanıcının girdiği iki sayı arasındaki elemanları belirlemek için diğeri ise en küçük asal sayı olan 2 ile bir önceki döngüde belirlenen sayı değişkeni arasındaki sayıları tutmak içindir.

Problem 3' te etkileşime giren öğeler ve öğe etkileşim durumları Şekil 9'da açıklanmıştır.



Şekil 9. Problem 3 muhtemel çözüm üzerindeki öğe etkileşimleri

1. Etkileşim (Değişken atama-Döngü): Problemin çözümü içerisinde kullanıcının gireceği sayıları atamak için tanımlanan değişkenler (birincisayi, ikincisayi) döngü(for)

içerisinde kullanılmıştır. Bu iki değişken arasındaki sayıları belirlemek için bu etkileşimin olması gereklidir.

2. Etkileşim (Koşul-Döngü): Döngü(for) içerisinde tanımlanan bir değişken(i) koşul(if) satırında da tanımlanmıştır. Şekil 9'a bakıldığında döngü-koşul arasında birden fazla etkileşim olduğu görülmektedir. Problemin çözümü için bu etkileşimin olması gereklidir.

3. Etkileşim (Döngü-Operatör): Döngü(for) içerisinde tanımlanan bir değişken(i), koşul(if) satırında operatör(i>1, i%x==0) içerisinde kullanılmıştır. Bu değişken(i) iki operatör içerisine de girmiştir.

4. Etkileşim (Koşul-Operatör): Her iki koşul(if) satırında da operatörler kullanılmıştır. Koşul yapılarında genellikle bir karşılaştırma söz konusu olduğundan *koşul-operatör etkileşimi* gerçekleşmiştir.

Temel öğeler ve öğeler arası etkileşimler temel alındığında 3. Problemin zorluk durumu Tablo 8'de gösterilmektedir.

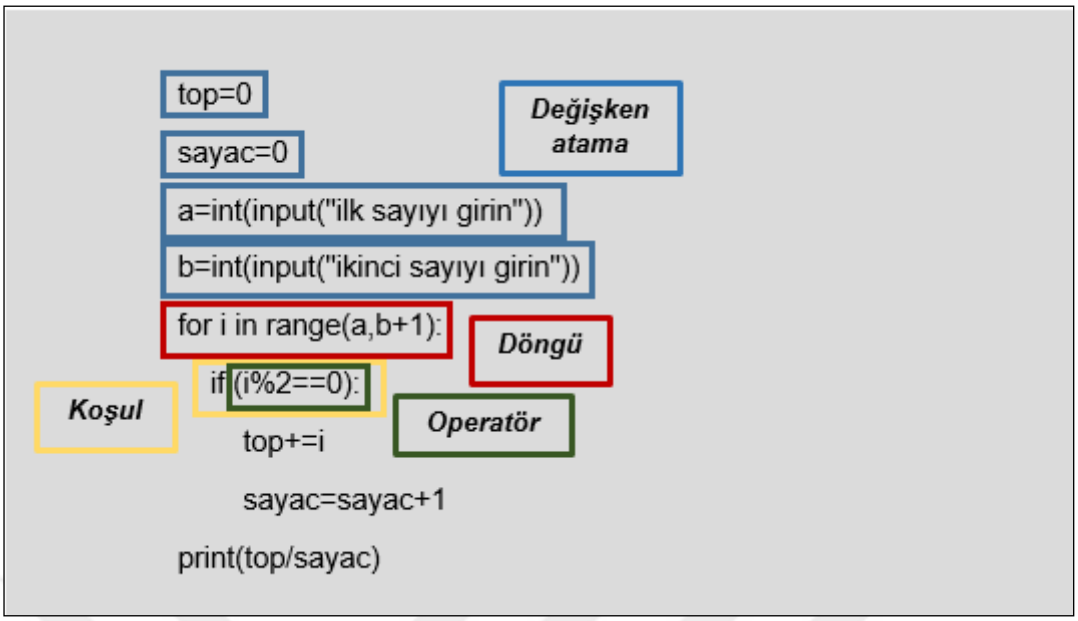
Tablo 8. Problem 3. Zorluk Durumu

	Temel Öğeler		Öğe Etkileşimi		Öğelerin Toplam Güçlüğü
Problem 3	Değişken atama	Y	Koşul-Operatör	Y	Yüksek (Y)
	Operatörler	Y	Döngü-Koşul	Y	
	Koşul	Y	Operatör-Döngü	Y	
	Döngü	Y	Döngü-Değişken atama	Y	
	Fonksiyon	-			
	Sonuç	Y	Sonuç	Y	

Tablo 8 incelendiğinde 3.problemin çözümü için değişken atama, operatör, koşul ve döngü yapılarının kullanılması gerektiği görülmektedir. Bu nedenle bu yapılar yüksek olarak değerlendirilmiştir. Sonuç olarak alan uzmanları *temel öğeler* kategorisine yüksek derece vermiştir. Kullanılan bu yapıların birbirleriyle etkileşme durumları *öğe etkileşimleri* kategorisinde gösterilmiştir. Bu kategoriye bakıldığında *koşul-operatör, döngü-koşul, operatör-döngü, değişken atama-döngü* yapıları arasında etkileşimlerin yüksek olduğu görülmektedir. Bu tablo problemin çözümü için bu etkileşimlerin olması gerektiğini söylemektedir. Bu etkileşimlere bakıldığında *öğe etkileşimi* kategorisi sonuç olarak yüksek olarak değerlendirilmiştir. *Temel öğeler* ve *öğe etkileşimleri* kategorilerinden elde edilen bilgiler kapsamında bu problemin zorluk derecesinin yüksek olduğu belirlenmiştir.

Problem 4**:** *Kullanıcının girdiği 2 sayı arasındaki çift sayıların ortalamasını bulan programı yazınız.*

Problemin muhtemel çözümü üzerinde alan uzmanlarının belirledikleri zorluk durumları Şekil 10'da sunulmuştur.



Şekil 10. Problem 4'ün içerdiği temel öğeler

Özetle; problem 4'te kullanılan temel öğeler ve öğelerin etkileşim durumları Tablo 9'da gösterilmiştir.

Tablo 9. Problem 4 Temel Öğeler ve Öğe Etkileşim Durumu

Temel Öğeler	Değişken Atama	Operatör	Koşul	Döngü
Değişken Atama		√	√ √	√ √ √
Operatör	√		√	√
Koşul	√ √	√		√
Döngü	√ √ √	√	√	

Tablo 9 incelendiğinde değişken atama, operatör, koşul ve döngü olmak üzere ele alınan yapılardan 4 adedinin kullanıldığı görülmektedir. Etkileşimlere bakıldığında değişken atama-operatör yapıları arasında 1, değişken atama-koşul yapıları arasında 2, değişken atama-döngü yapıları arasında 3, değişken atama-operatör yapıları arasında 1, koşul-operatör yapıları arasında 1, koşul-döngü yapıları arasında 1 ve koşul-döngü yapıları arasında 1 olmak üzere 9 adet etkileşim söz konusudur.

Problem 4 için kullanılması gereken yapılar ve kullanılma durumları aşağıda açıklanmıştır.

Değişken Atama: Problem kullanıcının girdiği iki sayı arasındaki çift sayıların ortalamasını istemektedir. Kullanıcıdan alınacak olan sayılar için iki değişken (a,b), bu değişkenler arasında kaç adet çift sayı olduğunu bulmak için bir değişken (sayac) ve çift

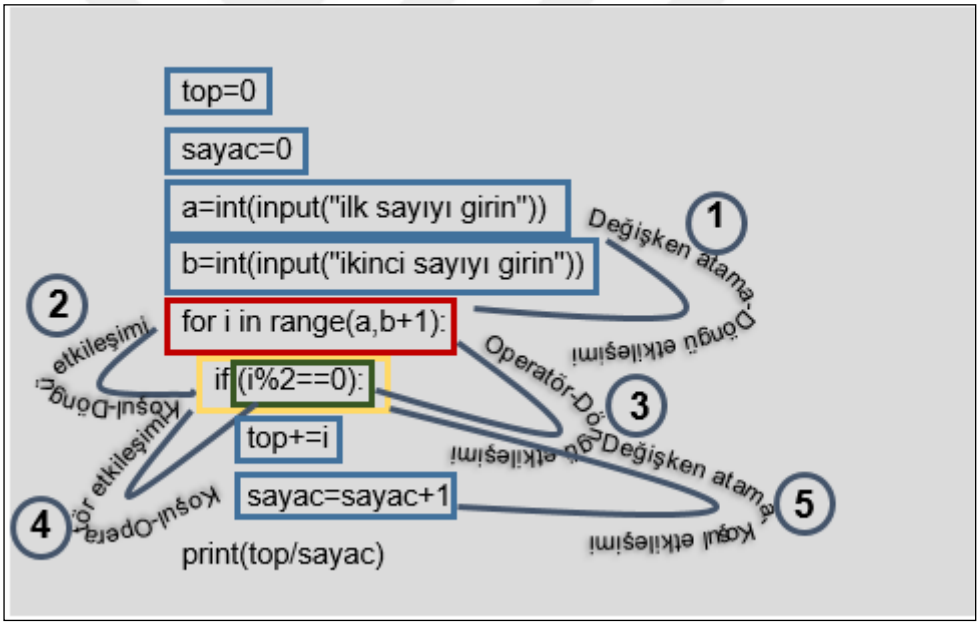
sayıların toplamını atamak için bir değişken(top) tanımlanmıştır. Bu değişkenler yardımıyla kullanıcıdan iki sayı alınmış, çift sayıların toplamı ve kaç adet olduğu bulunmuştur.

Operatör: Çift sayıları bulmak için sayının 2'ye bölümünden kalan 0 olmalıdır. Problemin muhtemel çözümüne bakıldığında bu işlemin operatörler(i%2==0) yardımıyla yapıldığı görülmektedir.

Koşul: Kullanıcı tarafından girilen iki sayı arasındaki hangi sayının çift olduğunu bulmak için koşul yapısı kullanılmıştır. Eğer sayının 2'ye bölümünden kalan 0 ise sayı çift demektir. Bu cümlede bir koşul söz konusudur. Bu koşul(if i%2==0) sağlanırsa o sayı işleme alınacaktır.

Döngü: Kullanıcıdan istenen iki sayı arasındaki sayıları belirlemek için döngü(for) kullanılmıştır. Bu işlemi gerçekleştirmek için döngü yapısının kullanılması zorunludur.

Problem 4'te etkileşime giren öğeler ve öğe etkileşim durumları Şekil 11' de açıklanmıştır.



Şekil 11. Problem 4 muhtemel çözüm üzerindeki öğe etkileşimleri

1. Etkileşim (Değişken atama-Döngü): Kullanıcının girdiği iki sayının atandığı değişkenler(a,b) döngü(for) içerisinde kullanılarak bu iki sayı arasındaki sayılar döngü içerisindeki değişkene(i) aktarılmıştır.

2. Etkileşim (Koşul-Döngü): Döngü içerisinde tanımlanan değişken(i) koşul(if) yapısı içerisinde kullanılmıştır. Böylelikle bu aşamada *döngü-koşul etkileşimi* gerçekleşmiştir.

3. Etkileşim (Operatör-Döngü): Döngü içerisinde tanımlanan değişken(i), koşul(if) satırında operatör(i%2==0) içerisinde kullanılmıştır. Bu değişkenin operatörde kullanılması döngü-operatör arasında etkileşim olmasını sağlamıştır.

4. Etkileşim (Koşul-Operatör): Koşul(if) satırında operatör(i%2==0) kullanımı yapılmıştır. Bu işlemin yapılması *koşul-operatör etkileşimini* sağlamıştır.

5. Etkileşim (Değişken atama-Koşul): Koşul(if) içerisinde kullanılan değişken(i), şart sağlandığı sürece farklı bir değişkene(top) atanmıştır. Bu değişkenin koşul(if) yapısı altında bulunması bu etkileşimin gerçekleştiğini göstermektedir.

Temel öğeler ve öğeler arası etkileşimler temel alındığında 4. Problemin zorluk durumu Tablo 10'da gösterilmektedir.

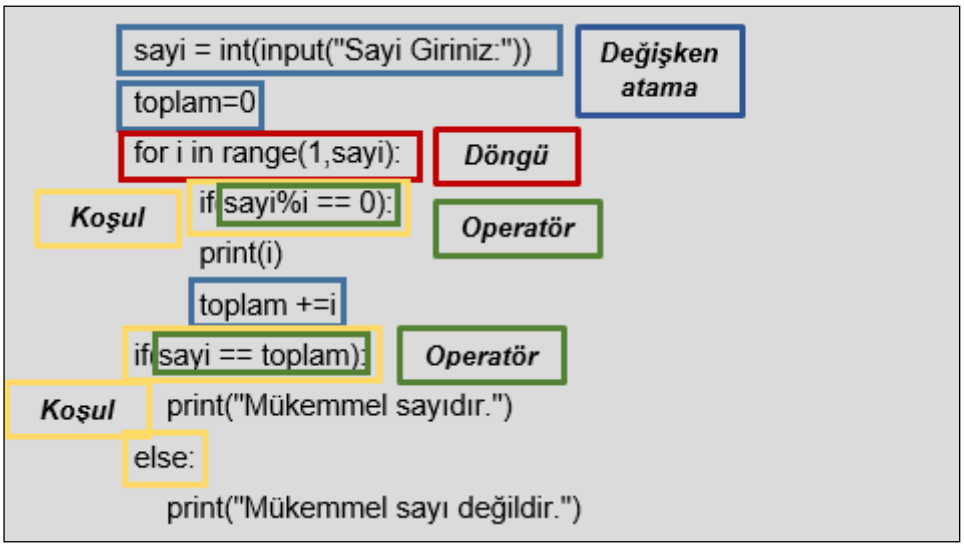
Tablo 10. Problem 4. Zorluk Durumu

	Temel Öğeler		Öğe Etkileşimi		Öğelerin Toplam Güçlüğü
Problem 4	Değişken atama	Y	Koşul-Operatör	Y	Yüksek (Y)
	Operatörler	Y	Döngü-Koşul	D	
	Koşul	Y	Operatör-Döngü	Y	
	Döngü	O	Döngü-Değişken atama	Y	
	Fonksiyon	O	Değişken atama-Koşul	D	
	Sonuç	Y	Sonuç	Y	

Tablo 10 incelendiğinde 4.problemin çözümü için değişken atama, operatör, koşul ve döngü yapılarının kullanılması gerektiği görülmektedir. Bu nedenle bu yapılar yüksek olarak değerlendirilmiştir. Kullanılan bu yapıların birbirleriyle etkileşme durumlarına bakıldığında tüm etkileşimlerin yüksek değerde olduğu görülmektedir. Bu değerlerin yüksek olmasının nedeni problemin çözümü için bu etkileşimlerin gerçekleşmesinin zorunlu olmasındandır. Temel öğeler ve öğe etkileşimi kategorisi sonuçlarına göre bu problemin zorluk derecesi yüksek olarak değerlendirilmiştir.

Problem 5**:** *Bir sayının kendisi dışında bütün pozitif bölenlerinin toplamı kendisine eşit olan sayılara mükemmel sayı denir. Kullanıcının girdiği sayının mükemmel sayı olup olmadığını kontrol eden kodu yazınız.*

Problemin muhtemel çözümü üzerinde alan uzmanlarının belirledikleri zorluk durumları Şekil 12'de sunulmuştur.



Şekil 12. Problem 5'in içerdiği temel öğeler

Özetle; problem 5'te kullanılan temel öğeler ve öğelerin etkileşim durumları Tablo 11' de gösterilmiştir.

Tablo 11. Problem 5 Temel Öğeler ve Öğe Etkileşim Durumu

Temel öğeler	Değişken Atama	Operatör	Koşul	Döngü
Değişken Atama		√ √ √ √	√ √ √	√
Operatör	√ √ √ √		√ √	√ √ √
Koşul	√ √ √	√ √		√ √ √
Döngü	√	√ √ √	√ √ √	

Tablo 11 incelendiğinde değişken atama, operatör, koşul ve döngü olmak üzere ele alınan yapıların 4 adedinin kullanıldığı görülmektedir. Etkileşimlere bakıldığında değişken atama-operatör yapıları arasında 4, değişken atama-koşul yapıları arasında 3, değişken atama-döngü yapıları arasında 1, koşul-operatör yapıları arasında 2, operatör-döngü yapıları arasında 3, koşul-döngü yapıları arasında 3 olmak üzere 16 adet etkileşim söz konusudur.

Problem 5 için kullanılması gereken yapılar ve kullanılma durumları aşağıda açıklanmıştır.

Değişken atama: Problem kullanıcının girdiği sayının mükemmel olup olmadığını bulan program komutlarını istemektedir. Görsel 9'a bakıldığında kullanıcının sayı girmesini sağlamak için bir değişken(sayi) tanımlandığı görülmektedir. Mükemmel sayı, problem cümlesinde de belirtildiği gibi sayının bölenlerinin kendisine eşit olması şartını

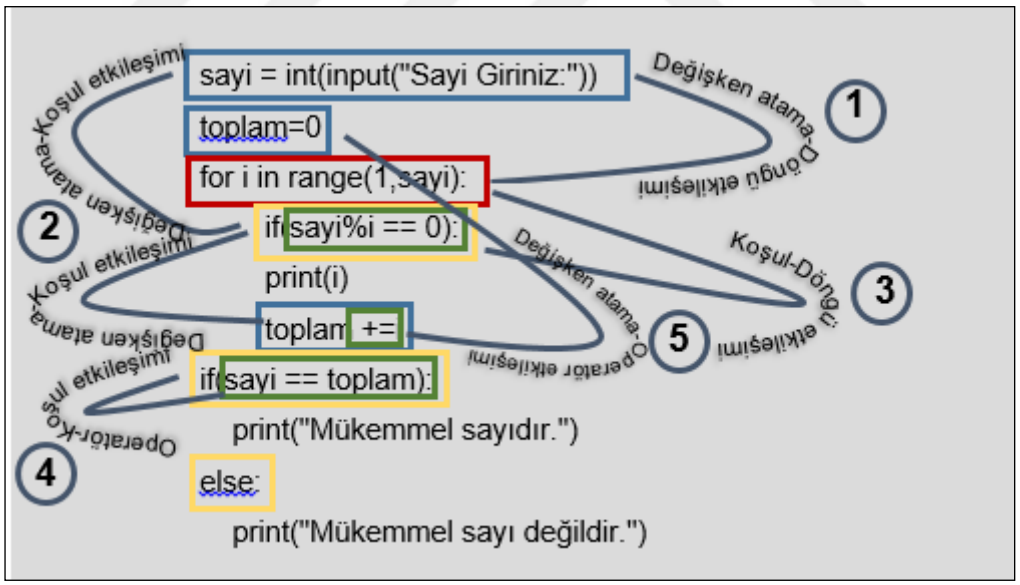
sağlamalıdır. Bunun için bir adet daha değişken(toplam) tanımlanmıştır. 1'den o sayıya kadar olan bölenler bu değişkene atanmıştır.

Operatör: Koşul satırlarında gerekli işlemlerin yapılması için operatörler (sayi%i==0, sayi==toplam) kullanıldığı görülmektedir. Bu problemin çözümü için operatörlerin kullanılması gereklidir.

Koşul: Kullanıcının girdiği sayının bölenlerini bulmak için bu problemde koşul(if) yapısı kullanılmalıdır. Kullanılan ilk koşul yapısında 1'den girilen sayıya kadar olan sayılar bölen olarak değerlendirilmiştir. Girilen sayı bölen olarak alınan sayılara tam bölünüyorsa, bu sayılar değişkene atanmıştır. Bu koşulun sağlanması için if kullanılmıştır. Diğer bir koşul ifadesinde bölen sayıların atandığı değişken ile kullanıcının girdiği değişkenin eşitliği sorgulanmaktadır. Bu koşul için yine if yapısı kullanılmalıdır.

Döngü: Kullanıcı tarafından girilen sayının bölenlerini bulmak için 1'den o sayıya kadar olan sayıların bir değişkende tutulması gerekir. Bu işlemi sağlayan döngü(for) yapısıdır. Bu problemin çözümü için döngü yapısının kullanılması gereklidir.

Problem 5' de etkileşime giren öğeler ve öğe etkileşim durumları Şekil 13'te açıklanmıştır



Şekil 13. Problem 5 muhtemel çözümü üzerindeki öğe etkileşimi

1. Etkileşim (Değişken atama-Döngü): Kullanıcı tarafından girilen sayının atandığı değişken(sayi) döngü(for) içerisinde kullanılmıştır. Böylelikle bu aşamada *değişken atama-döngü etkileşimi* gerçekleşmiştir.

2. Etkileşim (Değişken atama-Koşul): Kullanıcı tarafından girilen sayının atandığı değişken(sayi) her iki koşul(if) yapısında da kullanılmıştır. Böylelikle birden fazla *değişken atama-koşul etkileşimi* gerçekleşmiştir.

3. Etkileşim (Koşul-Döngü): Problem çözümü için döngü(for) satırında 1 ile kullanıcının girmiş olduğu sayı arasında olan sayıları aktarmak için bir değişken(i) tanımlanmıştır. Bu değişkenin koşul(if) satırında da kullanıldığı görülmektedir. Bu aşamada *koşul-döngü etkileşimi* gerçekleşmiştir.

4. Etkileşim (Operatör-Koşul): Her iki koşul(if) içerisinde de operatör(sayi%i==0, sayi==toplam) kullanımı yapılmıştır. Bu işlemlerle bu problemde iki adet *operatör-koşul etkileşimi* gerçekleşmiştir.

5. Etkileşim (Değişken atama-Operatör): Girilen sayının bölenlerinin toplamını atamak için tanımlanan değişken(toplam) ile 1'den başlayarak girilen sayı arasındaki sayıların atandığı değişkenin(i) işleme sokulduğu(toplam+=i) aşamada *değişken atama-operatör etkileşimi* gerçekleşmiştir.

Temel öğeler ve öğeler arası etkileşimler temel alındığında 5. Problemin zorluk durumu Tablo 12'de gösterilmektedir.

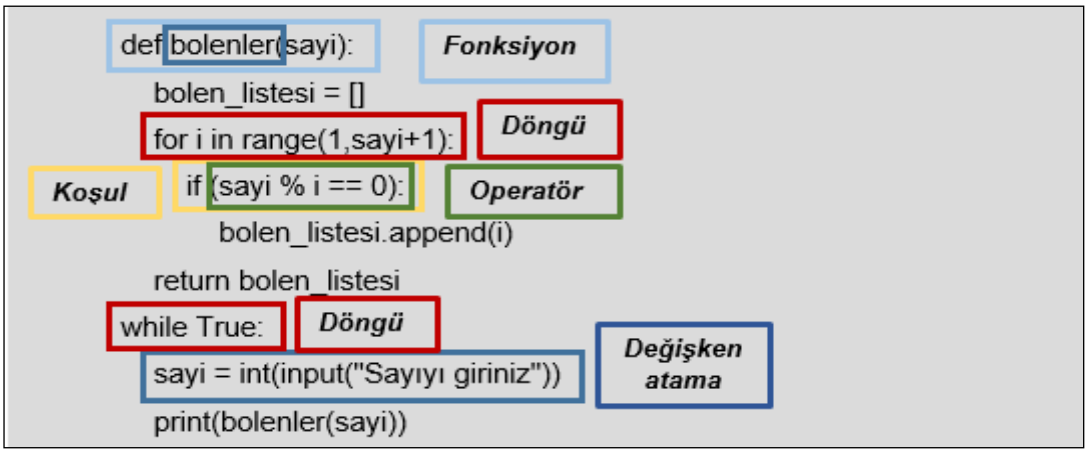
Tablo 12. Problem 2. Zorluk Durumu

	Temel Öğeler		Öğe Etkileşimi		Öğelerin Toplam Güçlüğü
Problem 5	Değişken atama	Y	Koşul-Operatör	Y	Yüksek (Y)
	Operatörler	Y	Döngü-Koşul	Y	
	Koşul	Y	Operatör-Döngü	Y	
	Döngü	Y	Döngü-Değişken atama	Y	
	Fonksiyon	-	Değişken atama-Koşul	Y	
	Sonuç	Y	Sonuç	Y	

Tablo 12 incelendiğinde 5.problemin çözümü için değişken atama, operatör, koşul ve döngü yapılarının kullanımının gerekli olduğu görülmektedir. Bu nedenle bu yapılar yüksek olarak değerlendirilmiştir. Kullanılan bu yapıların birbirleriyle etkileşme durumlarına bakıldığında tüm etkileşimlerin yüksek olduğu görülmektedir. Bu etkileşimlere yüksek değer verilmesinin nedeni problemin muhtemel çözümünde de gösterildiği üzere yapıların birbirleriyle etkileşmeden problemin çözümünün yapılamayacağıdır.

Problem 6*: Kullanıcının girdiği sayının pozitif tam bölenlerini bulan programı fonksiyon kullanarak yazınız.*

Problemin muhtemel çözümü üzerinde alan uzmanlarının belirledikleri zorluk durumları Şekil 14'te sunulmuştur.



Şekil 14. Problem 6'nın içerdiği temel öğeler

Özetle; problem 6'da kullanılan temel öğeler ve öğelerin etkileşim durumları Tablo 13'te gösterilmiştir.

Tablo 13. Problem 6 Temel Öğeler ve Öğe Etkileşim Durumu

Temel öğeler	Değişken Atama	Operatör	Koşul	Döngü	Fonksiyon
Değişken Atama		√	√	√	√
Operatör	√		√	√	-
Koşul	√	√		√	-
Döngü	√	√	√		-
Fonksiyon	√	-	-	-	

Tablo 13 incelendiğinde değişken atama, operatör, koşul, döngü ve fonksiyon olmak üzere ele alınan tüm yapıların kullanıldığı görülmektedir. Etkileşimlere bakıldığında fonksiyon yapısının operatör, koşul, döngü yapılarıyla etkileşime girmedikleri bunun dışında kalan tüm yapıların birbirleriyle birer kez etkileşime girdikleri belirlenmiştir. Bu etkileşimlerin sayısal olarak ifadesi 7 adet şeklindedir.

Problem 6 için kullanılması gereken yapılar ve kullanılma durumları aşağıda açıklanmıştır.

Değişken atama: Kullanıcının girdiği sayının bölenlerini bulmak için bir çözüm geliştirilen bu problemde kullanıcının girdiği sayıyı atamak için bir değişken(sayi) tanımlanmıştır.

Operatör: Problemin çözümü için koşul satırı içerisinde aritmetiksel işlem yapmak gereklidir. Bu işlemi yapmak için operatör(sayi%i==0) kullanılmıştır.

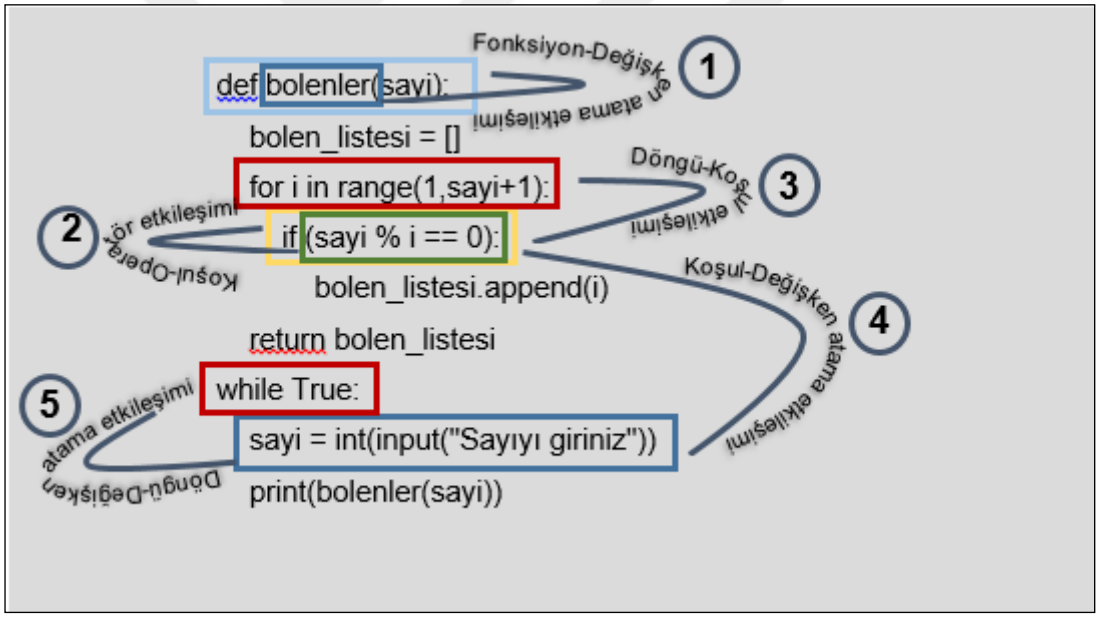
Koşul: Girilen sayının bölenlerini bulmak için sayıyı, 1'den başlayarak sayının kendisine kadar olan tüm sayılara bölünmesi gerekir. Eğer bölünüyorsa o sayı bölünen sayının böleni demektir. Bu şartı sağlayan sayıları bulmak için koşul(if) kullanılması

gerekir. Çözümde görüldüğü üzere bu işlem koşul satırı yardımıyla yapılmış ve bölen sayılar bulunmuştur.

Döngü: Girilen sayının bölenlerini bulmak için sayıyı, 1'den başlayarak sayının kendisine kadar olan tüm sayılara bölünmesi gerekir. Problemin muhtemelen çözümünde de 1'den o sayıya kadar olan sayıları belirlemek için döngü(for) kullanıldığı görülmektedir. Çözüm için döngünün kullanılması gereklidir. İkinci olarak bir döngü(while true) daha kullanılmıştır. Bu döngü bir alt satırında bulunan ifade sağlanana kadar işlemin tekrar etmesi gerektiğini belirtmek için kullanılmıştır.

Fonksiyon: Bu problemin, fonksiyon kullanılarak çözülmesi istenmektedir. Problemin çözümü için ilk olarak bir fonksiyon(def) tanımlanmıştır. Fonksiyon içerisinde gerekli işlemler yapılıp son kısımda fonksiyon çağırılarak işlem kısaltılmıştır.

Problem 6'da etkileşime giren öğeler ve öğe etkileşim durumları Şekil 15'te açıklanmıştır.



Şekil 15. Problem 6 muhtemel çözüm üzerindeki öğe etkileşimleri

1. Etkileşim(Fonksiyon-Değişken atama): Fonksiyon kullanılarak çözülmesi istenen bu problemin çözümünde ilk olarak bir fonksiyon(def) tanımlanmış ve fonksiyon içerisine bir değişken(bolenler) atanmıştır. Bu durum fonksiyon-değişken atama etkileşiminin gerçekleştiğini göstermektedir.

2. Etkileşim(Koşul-Operatör): Koşul içerisinde bir bölme işlemi gerçekleşmiştir. Bu işlem operatör(sayi%i==0) yardımıyla yapılmıştır. Problemin çözümü için bu etkileşimin gerçekleşmesi gereklidir.

3. Etkileşim (Döngü-Koşul): Döngü(for) içerisinde tanımlanan değişkenin(i) koşul içerisinde kullanılması *döngü-koşul etkileşiminin* gerçekleşmesini sağlamıştır.

4. Etkileşim (Koşul-Değişken atama): Kullanıcının girdiği sayının atandığı değişken(sayi) koşul(if) içerisinde kullanılmıştır. Problemin doğru çözümü için değişkenin koşul içerisinde kullanılması gereklidir.

5. Etkileşim (Döngü-Değişken atama): Kullanıcının girdiği sayının atandığı değişken(sayi) döngü(for) içerisinde kullanılmıştır. Problemin doğru çözümü için değişkenin döngü içerisinde kullanılması gereklidir.

Temel öğeler ve öğeler arası etkileşimler temel alındığında 6. Problemin zorluk durumu Tablo 14'te gösterilmektedir.

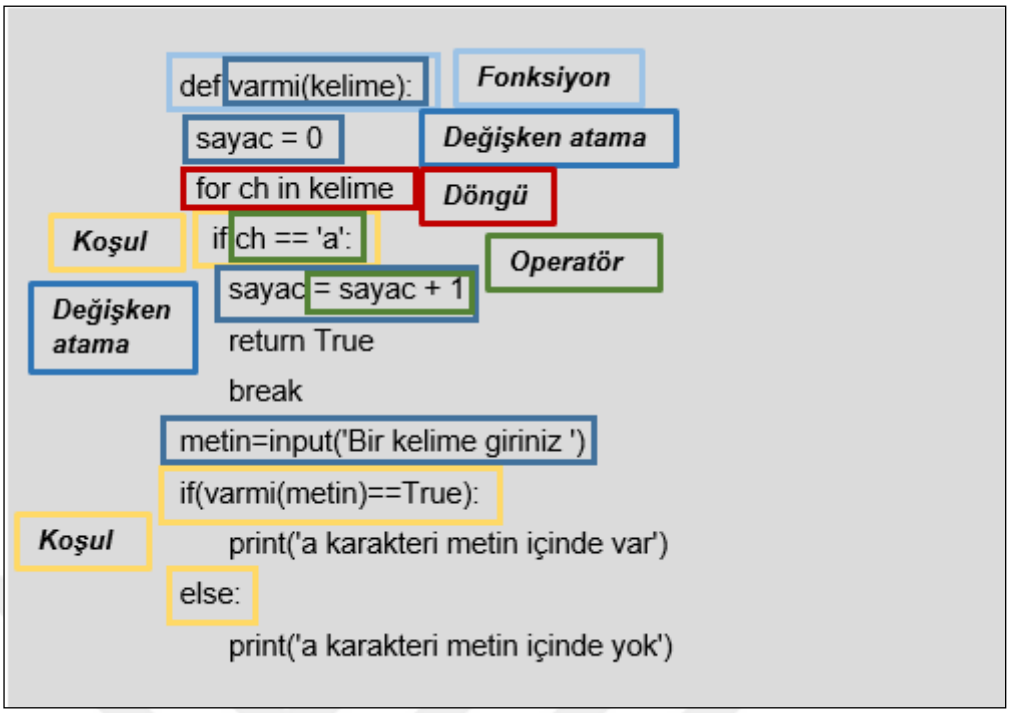
Tablo 14. Problem 6. Zorluk Durumu

	Temel Öğeler		Öğe Etkileşimi		Öğelerin Toplam Güçlüğü
Problem 6	Değişken atama	Y	Koşul-Operatör	Y	Yüksek (Y)
	Operatörler	Y	Döngü-Koşul	Y	
	Koşul	Y	Operatör-Döngü	Y	
	Döngü	Y	Döngü-Değişken atama	Y	
	Fonksiyon	O	Değişken atama-Koşul	Y	
			Fonksiyon –Diğer Yapılar	D	
	Sonuç	Y	Sonuç	Y	

Tablo 14 incelendiğinde 6.problemin çözümü için değişken atama, operatör, koşul, döngü ve fonksiyon olmak üzere ele alınan tüm yapıların kullanıldığı görülmektedir. Bu problem fonksiyon kullanılmadan da çözebilmektedir. Ancak bu problemin fonksiyon kullanılarak çözülmesi istenmektedir. Fonksiyon dışında diğer yapıların bu problemin çözümü için gerekli olduğu görülmektedir. Bu nedenle bu yapılar yüksek olarak fonksiyon yapısı ise orta derece olarak değerlendirilmiştir. *Temel öğeler* kategorisi sonuç olarak yüksek olarak belirlenmiştir. Kullanılan bu yapıların birbirleriyle etkileşme durumlarına bakıldığında *fonksiyon-diğer yapılarla etkileşiminin* düşük derecede, diğer tüm yapıların birbirleriyle etkileşme derecesinin yüksek olduğu görülmektedir. Bu etkileşimlere yüksek değer verilmesinin nedeni problemin çözümü için bu etkileşimlerin gerçekleşmesi gerektiğindendir.

Problem 7**:** *Bir string içerisinde belirlenen bir karakterin olup olmadığını kontrol eden programı fonksiyon kullanarak yazınız.*

Problemin muhtemel çözümü üzerinde alan uzmanlarının belirledikleri zorluk durumları Şekil 16'da sunulmuştur.



Şekil 16. Problem 7'nin içerdiği temel öğeler

Özetle; problem 7'de kullanılan temel öğeler ve öğelerin etkileşim durumları Tablo 15 'de gösterilmiştir.

Tablo 15. Problem 7 Temel Öğeler ve Öğe Etkileşim Durumu

Temel öğeler	Değişken Atama	Operatör	Koşul	Döngü	Fonksiyon
Değişken Atama		√	√	-	-
Operatör	√		√ √	√	-
Koşul	√	√ √		√	√
Döngü	-	√	√		√
Fonksiyon	-	-	√	√	

Tablo 15 incelendiğinde değişken atama, operatör, koşul, döngü ve fonksiyon olmak üzere ele alınan tüm yapıların kullanıldığı görülmektedir. Etkileşimlere bakıldığında değişken atama-operatör yapıları arasında 1, değişken atama-koşul yapıları arasında 1, koşul-operatör yapıları arasında 2, operatör-döngü yapıları arasında 1, koşul-döngü yapıları arasında 1, koşul-fonksiyon yapıları arasında 1 ve döngü fonksiyon yapıları arasında 1 olmak üzere 8 adet etkileşim söz konusudur.

Problem 7 için kullanılması gereken yapılar ve kullanılma durumları aşağıda açıklanmıştır.

Değişken atama: Problem cümlesi kullanıcıdan bir kelime girmesini istemektedir. Bunun için bir değişken(metin) tanımlanmıştır. Girilen kelimenin içerisinde belirlenen karakterin kaç adet olduğunu bulmak için bir değişken(sayac) daha tanımlanmıştır. Problemin çözümü için bu değişkenlerin tanımlanması gereklidir.

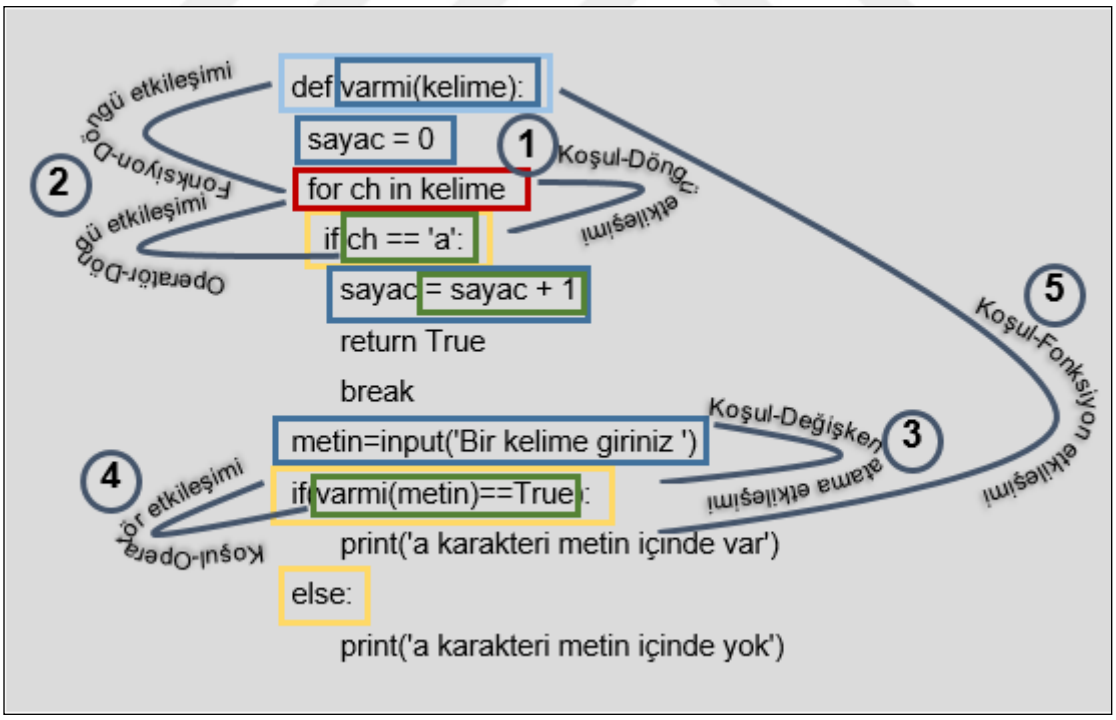
Operatör: Problemin muhtemel çözümü incelendiğinde koşul(if) ve değişken(sayac) satırında operatör(ch==a, sayac+1) kullanıldığı görülmektedir.

Koşul: Kullanıcının girdiği kelime içerisinde belirlenen karakterin var olup olmadığını sorgulamak için koşul(if) kullanılmıştır. Eğer şart sağlanıyorsa kaç adet olduğu işlemine geçilmiştir. Dolayısıyla bu problem için koşul kullanılması zorunludur.

Döngü: Kullanıcının girdiği kelime içerisindeki harfleri karakter olarak almak için döngü(for) kullanılmıştır.

Fonksiyon: Problem cümlesi çözümün fonksiyon kullanılarak yapılmasını istemektedir. Bu nedenle çözüme bir fonksiyon(def) tanımlayarak başlanmıştır. Ancak bu problem fonksiyon kullanılmadan da çözülebilir.

Problem 7' de etkileşime giren öğeler ve öğe etkileşim durumları Şekil 17'de açıklanmıştır.



Şekil 17. Problem 7 Muhtemel Çözüm Üzerindeki Öğe Etkileşimleri

1. Etkileşim (Koşul-Döngü): Döngü içerisinde tanımlanan değişkenin(ch) koşul(if) yapısında kullanılmasıyla *koşul-döngü etkileşimi* gerçekleşmiştir.

2. Etkileşim (Operatör-Döngü): Döngü içerisinde tanımlanan değişkenin(ch) koşul(if) yapısında operatör(ch==a) içerisinde kullanılmıştır. Aynı zamanda bu işlem döngü(for) satırı altında gerçekleşmiştir. Dolayısıyla bu aşamada *operatör-döngü etkileşimi* gerçekleşmiştir.

3. Etkileşim (Değişken atama-Koşul): Kullanıcının girdiği kelimeyi atamak için tanımlanan değişkenin(metin) koşul(if) içerisinde kullanılması *değişken atama-koşul etkileşiminin* gerçekleşmesini sağlamıştır.

4. Etkileşim (Koşul-Operatör): Döngü(for) yardımıyla kullanıcının girdiği kelime harflere(ch) ayrılmıştır. Bu harfin belirlenen karakter(a) olup olmadığı koşul(if) içerisinde operatörler(ch==a) yardımıyla kontrol edilmiştir. Böylelikle *koşul-operatör etkileşimi* gerçekleşmiştir.

5. Etkileşim (Koşul-Fonksiyon): Problemin çözümü için ilk başta tanımlanan fonksiyon(def) içerisindeki değişken(varmi) koşul(if) içerisinde belirlenen karakterin olup olmadığı kontrol edilmiştir. Böylelikle *koşul-fonksiyon etkileşimi* gerçekleşmiştir.

Problemin muhtemel çözümü üzerinde alan uzmanlarının belirledikleri zorluk durumları Şekil 16' da sunulmuştur.

Tablo 16. Problem 7. Zorluk Durumu

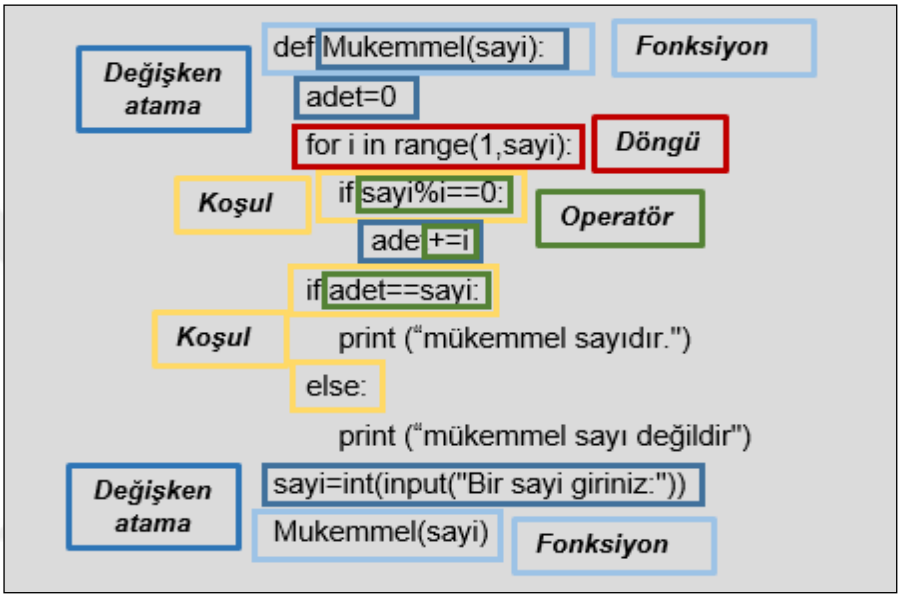
	Temel Öğeler		Öğe Etkileşimi		Öğelerin Toplam Güçlüğü
Problem 7	Değişken atama	Y	Fonksiyon-Döngü	Y	Yüksek (Y)
	Operatörler	Y	Fonksiyon-Koşul	Y	
	Koşul	Y	Döngü-Koşul	Y	
	Döngü	Y	Değişken atama-Döngü	Y	
	Fonksiyon	Y	Değişken atama-Koşul	Y	
			Fonksiyon-Değişken atama	Y	
			Operatör-Değişken atama/Döngü/Koşul/Fonk.	Y	
	Sonuç	Y	Sonuç	Y	

Tablo 16 incelendiğinde 7.problem için değişken atama, operatör, koşul ve döngü yapılarının kullanımının zorunlu olduğu görülmektedir. Bu nedenle bu yapılar yüksek olarak değerlendirilmiştir. Bu problemin fonksiyon kullanılmadan da çözülebileceği alan uzmanları tarafından belirlenmiştir. Dolayısıyla fonksiyon yapısı temel öğeler kategorisinde düşük olarak değerlendirilmiştir. Kullanılan bu üç yapının birbirleriyle etkileşme durumları yüksek olmalıdır. Öğe etkileşimlerine bakıldığında sadece *döngü-koşul etkileşiminin* orta derece olduğu görülmektedir. Bu etkileşimlere yüksek değer

verilmesinin nedeni problemin muhtemel çözümünde de gösterildiği üzere yapıların birbirleriyle etkileşmeden problemin çözümünün yapılamayacağıdır.

Problem 8**:** *Bir sayının kendisi dışında bütün pozitif bölenlerinin toplamı kendisine eşit olan sayılara mükemmel sayı denir. Kullanıcının girdiği sayının mükemmel sayı olup olmadığını fonksiyon kullanarak yazınız.*

Problemin muhtemel çözümü üzerinde alan uzmanlarının belirledikleri zorluk durumları Şekil 18'de sunulmuştur.



Şekil 18. Problem 8'in içerdiği temel öğeler

Özetle; problem 8'de kullanılan temel öğeler ve öğelerin etkileşim durumları Tablo 17'de gösterilmiştir.

Tablo 17. Problem 8 Temel Öğeler ve Öğe Etkileşim Durumu

Temel öğeler	Değişken Atama	Operatör	Koşul	Döngü	Fonksiyon
Değişken Atama		√	√ √ √	√	√
Operatör	√		√ √	√	-
Koşul	√ √ √	√ √		√	√
Döngü	√	√	√		√
Fonksiyon	√	-	√	√	

Tablo 17 incelendiğinde değişken atama, operatör, koşul, döngü ve fonksiyon olmak üzere ele alınan tüm yapıların kullanıldığı görülmektedir. Etkileşimlere bakıldığında değişken atama-operatör yapıları arasında 1, değişken atama-koşul yapıları arasında 3, değişken atama-döngü yapıları arasında 1, değişken atama-fonksiyon yapıları arasında 1,

koşul-operatör yapıları arasında 2, operatör-döngü yapıları arasında 1, koşul-döngü yapıları arasında 1 ve döngü-fonksiyon yapıları arasında 1 olmak üzere 12 adet etkileşim söz konusudur.

Problem 8 için kullanılması gereken yapılar ve kullanılma durumları aşağıda açıklanmıştır.

Değişken atama: Kullanıcının girdiği sayının mükemmel olup olmadığını kontrol eden bu problemde girilen sayıyı atamak için bir değişken(sayi) tanımlanmıştır. Bu sayının bölenlerini atamak için bir değişken(adet) daha tanımlanmıştır.

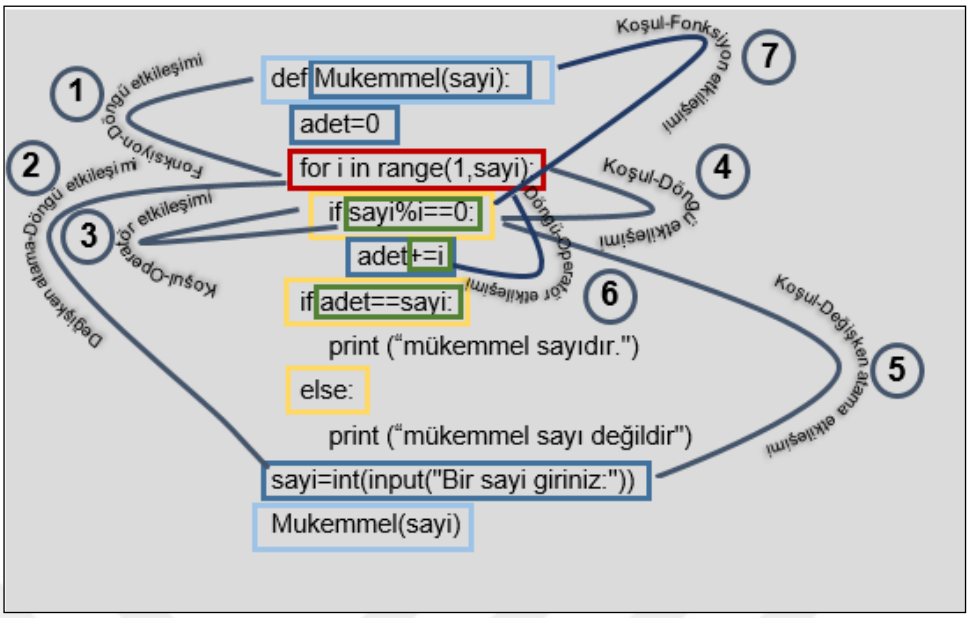
Operatör: Her iki koşul(if) içerisinde operatör(sayi%i==0, adet==sayi) kullanılmıştır. Bu problemin doğru çözümü için bu işlemlerin yapılması gereklidir.

Koşul: Bir sayının bölenleri toplamı o sayıya eşitse mükemmel sayıdır. Bu işlemi yapmak için ilk olarak sayının bölenlerinin bulunması gerekir. Girilen sayının 1'den o sayıya kadar olan sayılara bölünmesi işlemi için koşul(if) yapısı kullanılmalıdır. Eğer bu sayı aradaki sayılara tam bölünüyorsa bölen olarak alınabilir. Bu durumda bir koşul söz konudur. Bu sayının bölenleri toplamına eşit olup olmadığı kontrolü yine koşul(if) yapısı yardımıyla bulunabilir. Böylelikle bu problemin çözümü için koşul kullanılması gereklidir.

Döngü: Girilen sayının 1'den o sayıya kadar olan sayılara bölünmesi işlemi için aradaki sayıların bir değişkene aktarılarak tutulması gerekir. Bu işlemi gerçekleştiren döngü(for) yapısıdır.

Fonksiyon: Bu problemin fonksiyon kullanılarak yapılması istenmektedir. Problemin çözümü için ilk olarak bir fonksiyon(def) tanımlanmıştır. Fonksiyon içerisinde gerekli işlemler yapıldıktan sonra fonksiyon çağırma işlemi yapılarak çözüm gerçekleştirilmiştir.

Problem 8'de etkileşime giren öğeler ve öğe etkileşim durumları Şekil 19'da açıklanmıştır.



Şekil 19. Problem 8 muhtemel çözüm üzerindeki öğe etkileşimleri

1. Etkileşim (Fonksiyon-Döngü): Problem çözümünün fonksiyon kullanılarak yapılması istenmektedir. Bu nedenle ilk olarak tanımlanan fonksiyon(def) içerisindeki değişkenin döngü(for) yapısı içerisinde kullanılmasıyla bu etkileşim gerçekleşmiştir.

2. Etkileşim (Değişken atama-Döngü): Kullanıcıdan girilmesi istenen sayının atandığı değişken(sayi), döngü(for) içerisinde kullanılmıştır. Böylelikle değişken *atama-döngü etkileşimi* gerçekleşmiştir.

3. Etkileşim (Koşul-Operatör): Her iki koşul(if) içerisinde bölme işlemi ve eşitleme kontrolü işlemleri yapılmıştır. Yapılan bu işlemler operatörler(sayi%i==0, adet==sayi) yardımıyla yapılmıştır. Böylelikle *koşul-operatör etkileşimi* gerçekleşmiştir.

4. Etkileşim (Koşul-Döngü): Döngü(for) içerisinde tanımlanan değişkenin(i), koşul(if) içerisinde kullanılmasıyla bu etkileşim gerçekleşmiştir.

5. Etkileşim (Koşul-Değişken atama): Kullanıcıdan girilmesi istenen sayının atandığı değişken(sayi), koşul(if) içerisinde kullanılmıştır. Böylelikle değişken *atama-koşul etkileşimi* gerçekleşmiştir.

6. Etkileşim (Döngü-Operatör): Döngü(for) satırında tanımlanan değişken(i), operatör içerisinde kullanılmıştır. Böylelikle *döngü-operatör etkileşimi* gerçekleşmiştir.

7. Etkileşim (Fonksiyon-Koşul): Fonksiyon(def) içerisinde kullanılan değişkenin(sayi), koşul(if) içerisinde kullanılmasıyla *fonksiyon-koşul etkileşimi* gerçekleşmiştir.

8. Etkileşim (Değişken atama-Operatör): Tanımlanan değişkene(adet) başka bir değişkenin(i) ataması işleminde operatör(adet+=i) kullanılmıştır. Bu aşamada ***değişken atama-operatör etkileşimi*** gerçekleşmiştir.

Problemin muhtemel çözümü üzerinde alan uzmanlarının belirledikleri zorluk durumları Şekil 18' de sunulmuştur.

Tablo 18. Problem 8. Zorluk Durumu

	Temel Öğeler		Öğe Etkileşimi		Öğelerin Toplam Güçlüğü
Problem 8	Değişken atama	Y	Fonksiyon-Döngü/Koşul	O	Yüksek (Y)
	Operatörler	Y	Döngü-Koşul	Y	
	Koşul	Y	Operatör-Döngü/Koşul	Y	
	Döngü	Y	Döngü-Değişken atama	Y	
	Fonksiyon	O	Değişken atama-Koşul	Y	
			Değişken atama-Operatör	Y	
	Sonuç	Y	Sonuç	Y	

Tablo 18 incelendiğinde 8.problemin çözümü için değişken atama, operatör, koşul, döngü ve fonksiyon yapıları olmak üzere ele alınan tüm yapıların kullanımının gerekli olduğu görülmektedir. Ancak bu problem fonksiyon kullanılmadan da çözülebilmektedir. Problemin çözümünde fonksiyon kullanılması istendiğinden bu yapı temel öğeler kategorisinde derecelendirilmiştir. Bu nedenle fonksiyon yapısı orta, diğer yapılar yüksek olarak değerlendirilmiştir. Kullanılan bu yapıların birbirleriyle etkileşme durumları için öğe etkileşimleri kategorisine bakıldığında toplamda 8 adet etkileşim olduğu görülmektedir. Fonksiyon yapısının döngü *ve* koşul yapılarıyla girmiş olduğu etkileşim orta olarak değerlendirilirken diğer tüm yapı etkileşimleri yüksek olarak değerlendirilmiştir. Bu etkileşimlere yüksek değer verilmesinin nedeni problemin çözümü için bu etkileşimlerin olması gerektiğidir. Temel öğeler ve öğe etkileşimleri kategorileri sonuçlarına göre bu problemin zorluk derecesi alan uzmanları tarafından yüksek olarak değerlendirilmiştir.

Problemlerin tamamında yer alan temel öğeler ve öğe etkileşim durumları Tablo 19'da gösterilmektedir.

Tablo 19. Tüm Problemler Temel Öğe ve Öğe Etkileşimleri

Temel Öğeler		Değişken Atama	Operatör	Koşul	Döngü	Fonksiyon
		Öğe Etkileşimleri				
Problem_1	D.A		√			√ √
	O					√
	K					
	D					
	F					
Problem_2	D.A		√ √ √ √ √ √	√ √	√ √ √	√
	O			√	√ √ √	
	K					
	D					
	F					
Problem_3	D.A				√ √	
	O			√ √	√ √ √	
	K				√ √ √	
	D					
	F					
Problem_4	D.A		√	√ √	√ √ √	
	O			√	√	
	K				√	
	D					
	F					
Problem_5	D.A		√ √ √ √	√ √ √	√	
	O			√ √	√ √ √	
	K				√ √ √	
	D					
	F					
Problem_6	D.A		√	√	√	√
	O			√	√	
	K				√	
	D					
	F					
Problem_7	D.A		√	√		
	O			√ √	√	
	K				√	√
	D					√
	F					
Problem_8	D.A		√	√ √ √	√	√
	O			√ √	√	
	K				√	√
	D					√
	F					

4. 2. Problem Çözümleri Sırasında Öğrencilerin Deneyimledikleri İçsel Bilişsel Yük

Bu çalışmada öğrencilerin problemleri çözmek için program yazmaları esnasında karşılaştıkları çözüme yönelik zorluk durumları içsel bilişsel yüke işaret eden kaynaklar olarak ele alınmıştır. Bu durumlar programlama bilgi türlerinde kategorilendirilmiş ve bu kategoriler programlama yapıları temelinde incelenmiştir.

Öğrencilerin problemleri çözerken karşılaştıkları zorluklar bazen hata yapmalarına yol açmış ve öğrenciler bu hataları düzeltmek için uğraşmışlardır. Bu uğraşıların sonucu bazı durumlarda uygun çözümü (yazılması gereken kodu, yapılması gereken düzenlemeyi, düşünülmesi gereken stratejiyi vb.) bulmalarıyla, bazen de bulamamalarıyla sonuçlanmıştır. Bu çerçevede zorluk olarak ele alınan durumlar kodlarla ifade edilmiştir. Her bir zorluk türünün hangi problemde kaç kez deneyimlendiği, hatanın giderilip giderilemediğine ilişkin bulgular Tablo 20'de özetlenen bu kodlar aracılığıyla sunulmuştur. Bu tablo oluşturulurken, problem çözme sürecindeki öğrenci davranışlarına yönelik gözlemler, sesli düşünme protokolleri ve mülakatlar birlikte değerlendirilmiştir. Bu çerçevede karşılaşılan zorluklara ilişkin durumlar Tablo 20' de özetlenmiştir.

Tablo 20. Problemlerin Çözümünde Karşılaşılan Zorluklara İlişkin Durumların Kodlanması

	1	2	3
Zorluk/Hata Türü	Ç	DÇ	UÇ
	D	DB	UB

Tablo 20'de problemlerin çözümünde öğrencilerin karşılaştığı zorluklar, hatalarla uğraşma sırasında ne kadar çaba sarf edip çözdüğünü kısaca betimleyebilmektedir. Tablo içerisinde bulunan kodlar (*Ç:* çözdü, *DÇ:* düşünüp çözdü, *UÇ:* uğraşıp çözdü, *D:* durdu, *DB:* düşünüp bulamadı, *UB:* uğraşıp bulamadı) ekran kayıtları, sesli düşünme verileri, beden dili hareketleri ve gözlem formundan elde edilen veriler doğrultusunda oluşturulmuştur. Bu kısaltmaların açılımları şu şekildedir. Bu kodlar çerçevesinde öğrencinin hatalarla uğraşma durumunu göstermektedir. Bu durum 1 ile 3 arasında derecelendirilmiştir. *1:* uğraşmadan çözüp çözememesi, *2:* biraz uğraşarak çözüp çözememesi, *3:* çok uğraşarak çözüp çözememesi şeklindedir.

Öğrencilerin problem çözümleri sırasında ilk olarak ekranda yapmış oldukları hareketler analiz edilmiştir. Bu veriler ile öğrencilerin beden dili hareketleri, jest ve mimikleri, araştırmacı tarafından tutulan gözlem formu notları, uygulama esnasında öğrencinin sesli düşünmeleri ve uygulama sonunda öğrenciye sorulan soruların cevapları

birleştirilerek kodlar oluşturulmuş ve ilgili yerlere entegre edilmiştir. Bu kodların oluşturulma örneği Şekil 20'de şematize edilmiştir.

Şekil 20. Kodların oluşturulma süreci

Öğrencilerin problemlerin çözümü esnasında karşılaştığı zorluklar ve hatalar karşısında göstermiş oldukları davranışlar şu şekildedir.

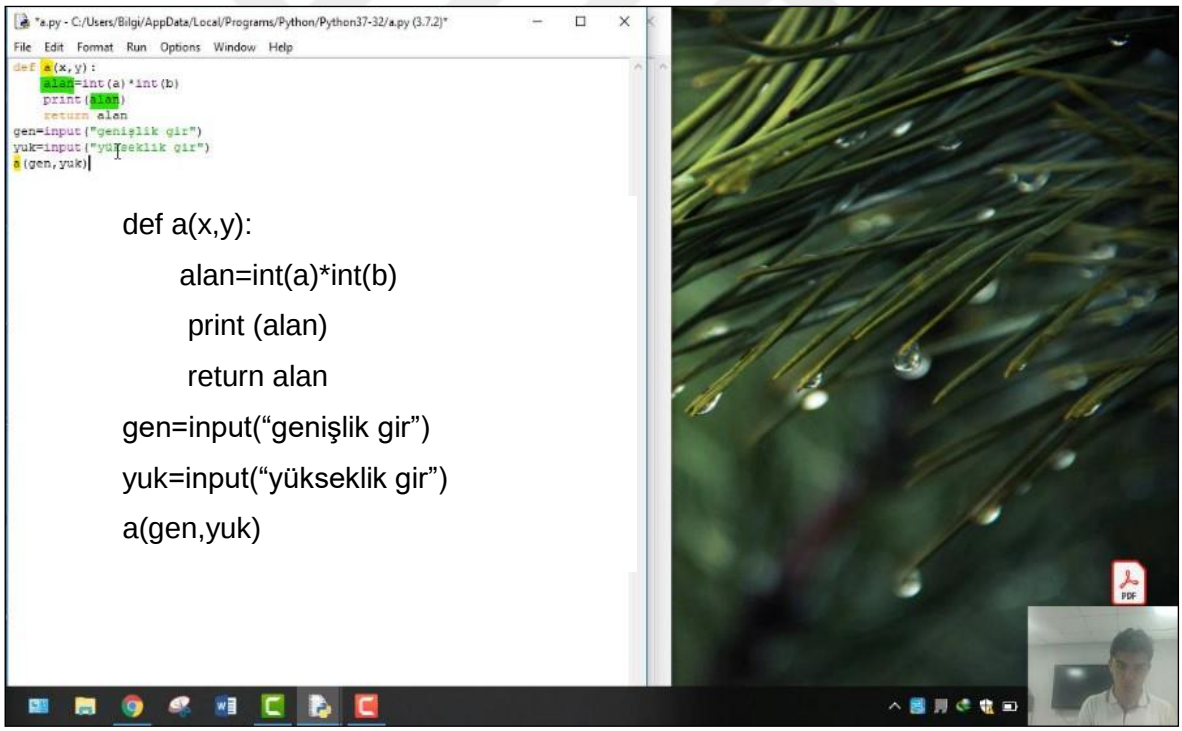
Problem 1**:** *Kullanıcının girdiği verilere göre dikdörtgenin alanını hesaplayan programı fonksiyon kullanarak yazınız.*

1. problemde karşılaşılan zorlanmalar ve hatalar Tablo 21'de gösterilmiştir.

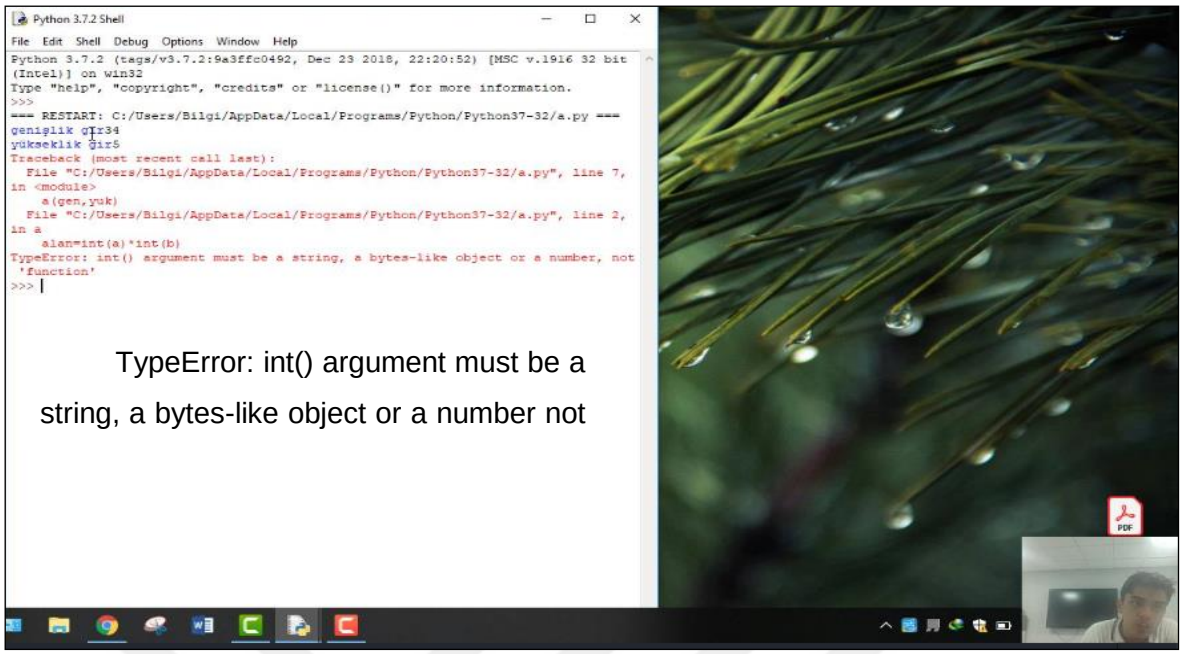
Tablo 21. Problem 1 Hata ve Zorluk Kaynakları

			Programlama Yapıları						
	Bilgi Türleri	Hata ve Zorluk Türleri	Değişken Atama			Fonksiyon			Top
Problem_1	Sözdizimsel	Girinti	4	6	8	5	6	8	37
			-			3	6	2	11
		Noktalama				7	7	5	19
						1	1	-	2
		Harf Hatası				-	1	1	2
									-
	Anlamsal	Yanlış Kullanma Durumu	3	1	1	3	3	2	13
			-				-		-
		Eksik Kullanma Durumu	2	3	1	-	2	3	11
							-		-
	Stratejik	Yanlış Yerde Kullanma Durumu	4	4	6				14
			5	5	4				14
		Eksik Kullanma Durumu	2	1	1				4
			-						-
	Toplam		61			66			127

Tablo 21 incelendiğinde sözdizimsel, anlamsal ve stratejik bilgi türlerinde zorluklarla karşılaşıldığı ve birçok hata yapıldığı görülmektedir. Bu zorlanmaların büyük bir kısmının sözdizimsel bilgi türünde(71) olduğu belirlenmiştir. Bu bilgi türünü stratejik bilgi türü (32) takip ederken son olarak anlamsal bilgi türünde(24) zorlanmalar olduğu görülmektedir. Bu problemin çözümü için ele alınan yapılardan değişken atama, operatör ve fonksiyon yapısının kullanılması gerekmektedir. Kullanılan bu yapılar sözdizimsel boyutta incelendiğinde değişken atama(18) ve fonksiyon(53) yapılarında zorlanmalar ve hatalarla karşılaşıldığı görülmektedir. Bu zorlanmaların bir kısmının(16) hemen çözüldüğü(Ç), bir kısmının(20) biraz düşünülüp çözüldüğü(DÇ), diğer kısmının(22) uğraşılıp çözüldüğü(UÇ) tespit edilmiştir. Karşılaşılan hataların bir kısmının(4) durup çözülemediği(D), bir kısmının(7) düşünülüp bulunamadığı(DB) diğer kısmının(2) uğraşılıp bulunamadığı(UB) durumlar olduğu görülmektedir. Bu durumda karşılaşılan zorlukların özellikle *girinti* ve *noktalama* türünde olduğu görülmektedir. Bu probleme ilişkin Ö5'in çözümü *Şekil 21'* de ve karşılaştığı hata mesajları *Şekil 22*'de gösterilmektedir. Örneğin;



Şekil 21. Ö5 problem 1 çözümü



Şekil 22. Ö5'in problem 1 çözümü esnasında aldığı hata mesajı

Yukarıda Ö5'in 1. Problemin çözümü esnasında çekilmiş olan ekran görüntüsü yer almaktadır. Ö5, Şekil 22'de görüldüğü gibi değişken atamada int(sayısal ifadeler için kullanılır.) komutunda yanlışlık yaptığından dolayı int ve string ayrımının yapılması ile ilgili hata almıştır. Bu esnada düşünceli ve hatanın nerede olduğunu anlamaya çalışan bir yüz ifadesiyle ekrana kilitlenen öğrenci tepkisini "Değişkende sözdizimi hatası verdi değiştirdim ama yine hata veriyor, vermemesi gerekir." sesli düşünme ifadesiyle ortaya koymuştur. Öğrenci programı tekrar çalıştırdıktan sonra "Doğru veriyor ama çoklu veriyor." şeklinde durumu ifade etmiş ancak gözlemlerden elde edildiği gibi çok uğraşmasına(UB) rağmen bu hata giderilememiştir. Uygulamadan sonra yapılan mülakatta Ö5 bu problem çözümü sırasındaki davranışlarını değerlendirirken "*Problem çözüm sırasında biraz zorlandım*. *Değişken satırında hata verdi, niye verdi anlamadım. Değiştirdim ama yine verdi "* şeklinde yaşadığı zorluğu ifade etmiştir.

Problem çözümünde kullanılması gereken programlama yapılarına anlamsal boyutta bakıldığında değişken atama(11) ve fonksiyon(13) yapılarında zorlanmalar olduğu belirlenmiştir. Bu zorlanmaların bir kısmının(8) hemen çözüldüğü(Ç), bir kısmının(9) biraz düşünülüp çözüldüğü(DÇ), diğer kısmının(7) uğraşılıp çözüldüğü(UÇ) tespit edilmiştir. Anlamsal boyutta karşılaşılan hataların giderilemediği bir durumla karşılaşılmadığı görülmektedir. Bu durumda karşılaşılan zorlukların ve hataların %100'ü giderilmiştir. Giderilen zorlanmalar ise özellikle *yanlış kullanma* ve *eksik kullanma* türlerinde zorluk yaşandığını göstermektedir. Bu zorluk durumları öğrenciler tarafından aşağıdaki gibi örneklendirilebilir. Örneğin;

Ö4: Programı çalıştırdığında dikdörtgenin alan hesaplama formülünü yazdığı değişken atama satırını *yanlış kullandığından* dolayı hata almıştır. *Girinti* hatası olabileceğini düşüncesiyle hatayı gidermeye çalışırken sesli bir şekilde *" Bu satırda bir geri gelelim."* ifadesini söylemiştir. Bu şekilde fonksiyon içine yazdığı tüm satırları bir tık geri almış, programı çalıştırdığında aynı hata ile tekrar karşılaşmıştır. Ö4 bu durumda *"Yine aynı oluyor ya.. Allah Allah"* şeklinde sesli düşünmüştür. Sonuç olarak çok uğraşsa(UB) da problemi çözememiştir. Uygulama sonrasında yapılan mülakatta Ö4 "*Çok zorlandım, hataları çözemedim*." şeklinde yaşadığı zorluğu ifade etmiştir.

Bu yapılar stratejik boyutta incelendiğinde değişken atama(32) yapısında zorlanmalar olduğu, fonksiyon yapısında herhangi zorluk ya da hata ile karşılaşılmadığı görülmektedir. Bu zorlanmaların bir kısmının(6) hemen çözüldüğü(Ç), bir kısmının(5) biraz düşünülüp çözüldüğü(DÇ), diğer kısmının(7) uğraşılıp çözüldüğü(UÇ) tespit edilmiştir. Karşılaşılan hataların bir kısmının(5) durup çözülemediği(D), bir kısmının(5) düşünülüp bulunamadığı(DB) diğer kısmının(4) uğraşılıp bulunamadığı(UB) durumlar olduğu görülmektedir. Bu durum problem çözülürken hangi yolun kullanılması gerektiği konusunda özellikle *yanlış yerde kullanma* ve *eksik kullanma* türlerinde zorlanıldığını göstermektedir.

Problem 2**:** *Kullanıcıdan tek satırda öğrenci adı, vize notu, final notu alınacaktır. Bu şekilde 5 öğrenci için kullanıcıdan veri alıp sınıf not ortalamasını hesaplayınız(vize*0.4+final*0.6) ve her öğrencinin adını, vize notunu, final notunu, ortalamasını ve sınıf ortalamasına göre geçip geçmediğini hesaplayan programı fonksiyon kullanarak yazınız.*

2. problemde karşılaşılan zorlanmalar ve hatalar Tablo 22'de gösterilmiştir.

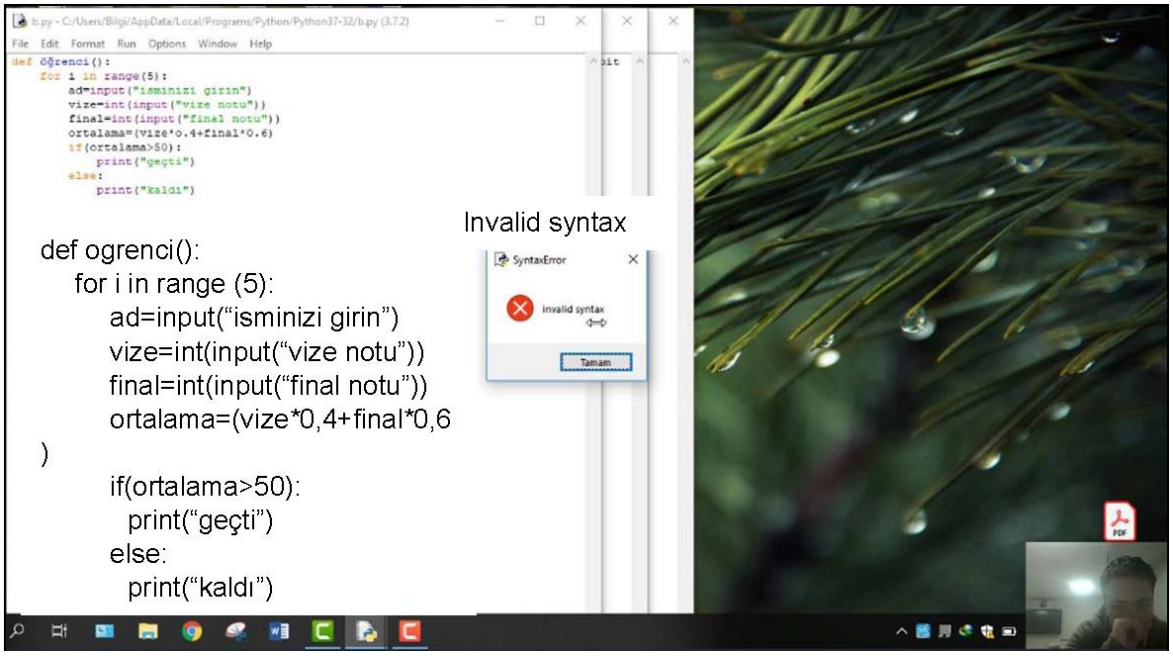
Tablo 22. Problem 2 Hata ve Zorluk Kaynakları

			Programlama Yapıları												
	Bilgi Türleri	Hata ve Zorluk Türleri	Değişken Atama			Koşul			Döngü			Fonksiyon			Toplam
Problem_2	Sözdizimsel	Girinti	5	6	6	7	9	11	7	8	2	8	6	3	78
			5	7	9	7	4	3		-		4	4	1	44
		Noktalama				8	9	5	8	7	3	4	3	2	49
							-			-		-	-	1	1
		Harf Hatası	1	1	3							1	-	-	6
				-											-
		Parantez	4	2	-										6
				-											-
	Anlamsal	Yanlış Kullanma	2	3	3	5	5	4	4	3	3	3	1	-	36
			-	1	2	2	-	-	1	2	-				8
		Eksik Kullanma	3	2	1	1	4	2	2	3	2	2	2	-	24
				-						-					-
		Belirleyememe	2	3	1				2	1	1				10
				-						-					-
		Karar Verememe	3	-	2	3	2	-	3	3	4				20
			4	3	2		-		2	1	-				12
	Stratejik	Yanlış Kullanma	4	3	6	2	3	3	4	3	3				31
			5	5	5		-		2	2	1				20
		Eksik Kullanma							3	2	3				8
									3	2	1				6
		Belirleyememe	5	5	1	2	2	3	3	2	1	2	1	1	28
			3	2	2				1	1	-	1	3	2	15
		Karar verememe	2	-	2										4
			4	5	-										9
	Toplam		145			106			109			55			415

Tablo 22 incelendiğinde sözdizimsel, anlamsal ve stratejik bilgi türlerinde zorluklarla karşılaşıldığı ve birçok hata yapıldığı görülmektedir. Bu zorlanmaların büyük bir kısmının sözdizimsel bilgi türünde(183) olduğu belirlenmiştir. Bu bilgi türünü stratejik bilgi türü(121) takip ederken son olarak anlamsal bilgi türünde(110) zorlanmalar olduğu görülmektedir. Bu problemin çözümü için ele alınan tüm yapıların(değişken atama, operatör, koşul, döngü ve fonksiyon) kullanılması gerekmektedir. Kullanılan bu yapılar sözdizimsel boyutta incelendiğinde değişken atama(49), koşul(63), döngü(35) ve fonksiyon(37) yapılarında zorlanmalar ve hatalarla karşılaşıldığı görülmektedir. Bu durumda bu problemde sözdizimsel boyutta karşılaşılan zorluklar çoğunlukla değişken atama ve koşul yapılarında rastlanmıştır.

Karşılaşılan zorlanmaların bir kısmının(53) hemen çözüldüğü(Ç), bir kısmının(51) biraz düşünülüp çözüldüğü(DÇ), diğer kısmının(35) uğraşılıp çözüldüğü(UÇ) tespit

edilmiştir. Karşılaşılan hataların bir kısmının(16) durup çözülemediği(D), bir kısmının(15) düşünülüp bulunamadığı(DB) diğer kısmının(14) uğraşılıp bulunamadığı(UB) durumlar olduğu görülmektedir. Bu durumda karşılaşılan zorlukların ve hataların yaklaşık %76'sı giderilmiştir. Bu zorlanmalar çoğunlukla *girinti* ve *noktalama* türünde zorluk yaşandığını göstermektedir. Bu zorluk durumları öğrenciler tarafından aşağıdaki gibi örneklendirilebilir. Örneğin;



Şekil 23. Ö3'ün problem 2 çözümü ve aldığı hata mesajı

Yukarıda Ö3'ün 2. Problemin çözümü esnasında çekilmiş olan ekran görüntüsü yer almaktadır. Ö3, Şekil 23'te görüldüğü üzere komutları *yanlış yerde* kullandığından dolayı "geçersiz sözdizimi*"* hatası almıştır. Hatanın ne olduğunu ve nerede olduğunu anlamaya çalışan bir yüz ifadesiyle elini yüzüne götürmüş ve düşünceli bir şekilde yazmış olduğu kod üzerinde tek tek incelemeler yapmaya başlamıştır. Uzun uzun düşünmüştür. Ö3 hatanın neden kaynaklandığını bulamayınca tepkisini "Her şeyi yaptım, nerede hata var ki?" sesli düşünme ifadesiyle ortaya koymuştur. Ö3 programı tekrar çalıştırdıktan sonra aynı sonucu almıştır. Hatanın nereden kaynaklandığını bulamadığı için öylece durmuş ve hatayı gidememiştir(D). Uygulamadan sonra yapılan mülakatta Ö3 bu problem çözümü sırasındaki davranışlarını değerlendirirken *"Problem ilk aşamada çok zor geldi ama yapabileceğimi düşündüm. Neden hata verdiğini anlamadım. Bu nedenle biraz zorlandım."* şeklinde yaşadığı zorluğu ifade etmiştir.

Problem çözümünde kullanılması gereken programlama yapıları anlamsal boyutta incelendiğinde değişken atama(37), koşul(28), döngü(37) ve fonksiyon(8) yapılarında

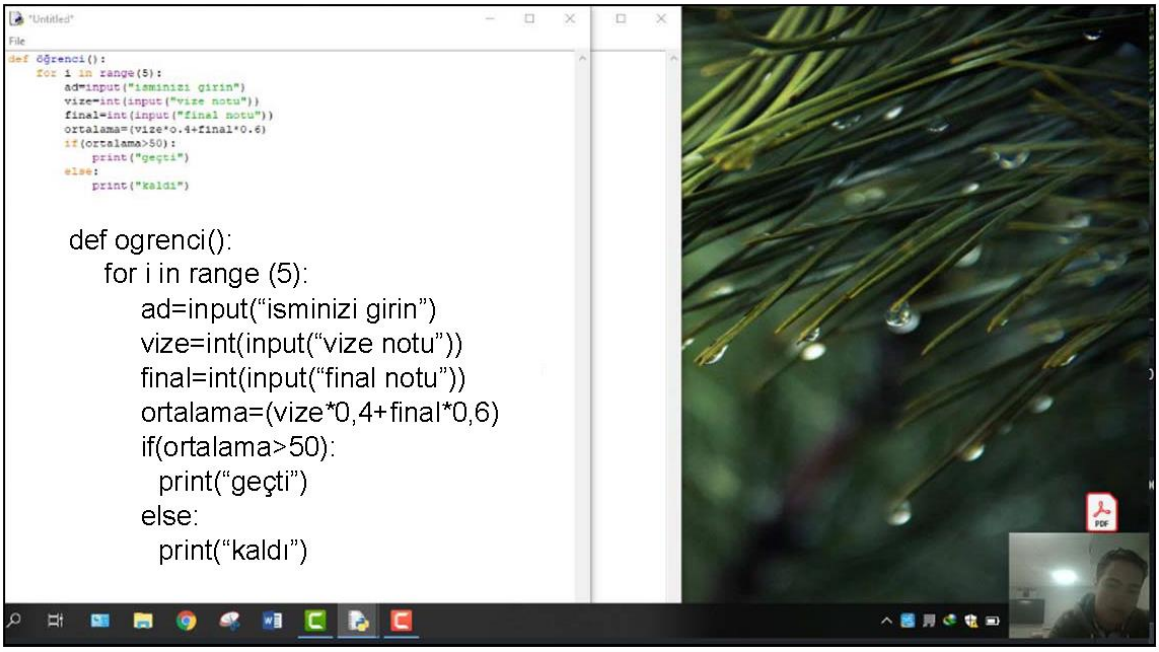
zorlanmalar olduğu görülmektedir. Bu durumda anlamsal boyutta karşılaşılan zorluklar çoğunlukla değişken atama, koşul ve döngü yapılarında rastlanmıştır.

Karşılaşılan zorlanmaların bir kısmının(35) hemen çözüldüğü(Ç), bir kısmının(32) biraz düşünülüp çözüldüğü(DÇ), diğer kısmının(23) uğraşılıp çözüldüğü(UÇ) tespit edilmiştir. Karşılaşılan hataların bir kısmının(9) durup çözülemediği(D), bir kısmının(7) düşünülüp bulunamadığı(DB) diğer kısmının(4) uğraşılıp bulunamadığı(UB) durumlar olduğu görülmektedir. Bu durumda karşılaşılan zorlukların ve hataların yaklaşık %82'si giderilirken %18'i giderilememiştir. Bu zorlanmalar özellikle *yanlış kullanma*, *eksik kullanma, belirleyememe ve karar verememe* türlerinde zorluk yaşandığını göstermektedir. Bu zorluk durumları öğrenciler tarafından aşağıdaki gibi örneklendirilebilir. Örneğin;

Ö5: Problem cümlesi, problemin çözümünde fonksiyon kullanılması gerektiğini söylemektedir. Ö5, fonksiyon kullanmış ancak fonksiyon içindeki değişkeni çözüme dahil etmemiş ve fonksiyon çağırma işlemini yapmamıştır. Dolayısıyla *eksik kullanım* hatası yapmıştır. Program çalıştırıldığında doğru sonuç vermemiştir. Ö5 çözümde eksiklik olduğunu fark etmiş ve "*Bir şeyi unuttuk ama neyi unutmuş olabiliriz, bir yerde hata mı yaptık ki?*" şeklinde sesli düşünerek belirtmiştir. Bunun üzerine dikkatli bakan gözlerle çözüme odaklanmış ve en baştan neler yaptığını ve yapması gerektiğini düşünmeye başlamıştır. Ancak yine de yapması gereken şeyi bulamamıştır. Uzun uzun düşünmesine rağmen zorluk giderilememiştir(DB). Uygulama sonrasında yapılan mülakatta Ö5 üzgün ve yorgun bir yüz ifadesiyle "*Ya aslında zor değildi*. Ama yapamadım. Biraz uğraştım, olmadı." şeklinde yaşadığı zorluğu ifade etmiştir.

Bu yapılar stratejik boyutta incelendiğinde değişken atama(59), koşul(15), döngü(37) ve fonksiyon(10) yapılarında zorlanmalar olduğu görülmektedir. Bu durumda stratejik boyutta karşılaşılan zorluklar çoğunlukla değişken atama ve döngü yapılarında rastlanmıştır.

Karşılaşılan zorlanmaların bir kısmının(27) hemen çözüldüğü(Ç), bir kısmının(21) biraz düşünülüp çözüldüğü(DÇ), diğer kısmının(23) uğraşılıp çözüldüğü(UÇ) tespit edilmiştir. Karşılaşılan hataların bir kısmının(19) durup çözülemediği(D), bir kısmının(20) düşünülüp bulunamadığı(DB) diğer kısmının(11) uğraşılıp bulunamadığı(UB) durumlar olduğu görülmektedir. Bu durumda karşılaşılan zorlukların ve hataların yaklaşık %59'u giderilirken %41'i giderilememiştir. Bu durum problem çözülürken hangi yolun kullanılması gerektiği konusunda özellikle *yanlış yerde kullanma* ve *eksik kullanma* türlerinde zorlanıldığını göstermektedir. Bu zorluk durumları öğrenciler tarafından aşağıdaki gibi örneklendirilebilir. Örneğin;



Şekil 24. Ö3'ün problem 2 çözümü

Yukarıda Ö3'ün 2. Problemin çözümü esnasında çekilmiş olan ekran görüntüsü yer almaktadır. Ö3, Şekil 24'te görüldüğü üzere problemin çözümü için yanlış planlama yapmış ve izlediği adımlar onu hataya sürüklemiştir. Ö3 özellikle *yanlış yerde kullanım* hatası yapmıştır. Problem içerisinde istenmediği halde ortalama 50'den büyükse geçti, küçükse kaldı ifadesi tamamen öğrenciye aittir. Bu anlamda doğru çözüme ulaşamamıştır. Ö3 programın neden çalışmadığını anlayamamış ve tepkisini "Her şeyi yaptım, nerede hata var ki?" sesli düşünme ifadesiyle ortaya koymuştur. Hatanın nereden kaynaklandığını bulamadığı için öylece durmuş ve hatayı gideremeyiştir(D). Uygulamadan sonra yapılan mülakatta Ö3 bu problem çözümü sırasındaki davranışlarını değerlendirirken "*Problem ilk aşamada çok zor geldi ama yapabileceğimi düşündüm...*" şeklinde yaşadığı zorluğu ifade etmiştir.

Problem 3: *Kullanıcının girdiği 2 sayı arasındaki asal sayıları bulan programı* yazınız.

3. problemde karşılaşılan zorlanmalar ve hatalar Tablo 23'te gösterilmiştir.

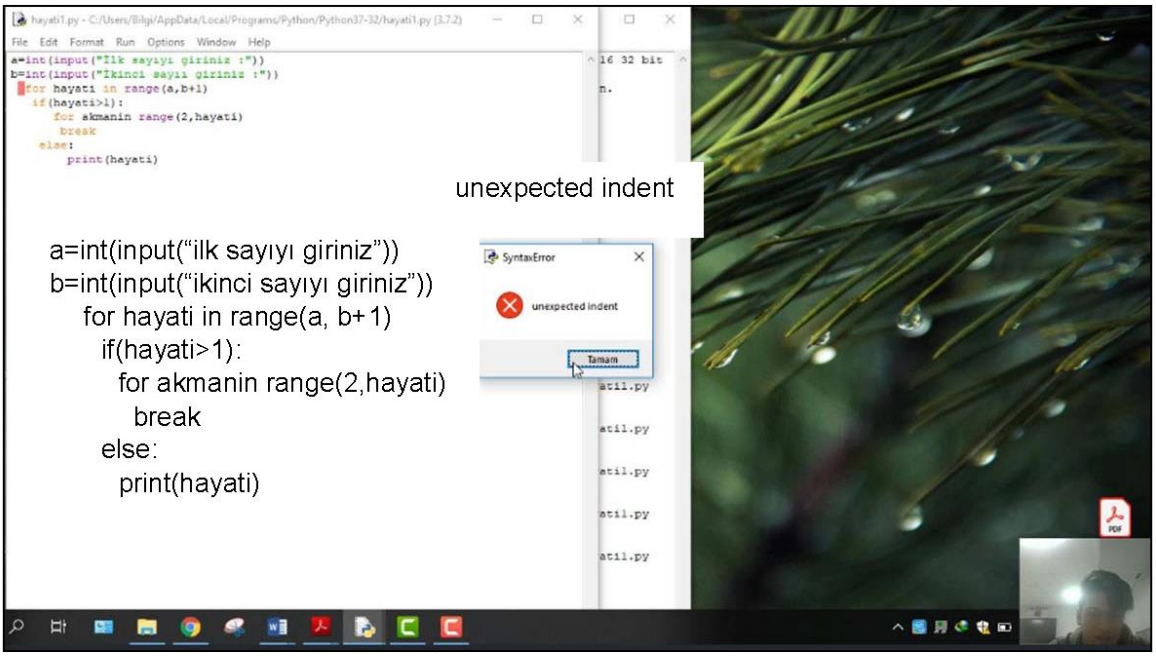
Tablo 23. Problem 3 Hata ve Zorluk Kaynakları

			Programlama Yapıları									
	Bilgi Türleri	Hata ve Zorluk Türleri	Değişken Atama			Koşul			Döngü			Toplam
Problem_3	Sözdizimsel	Girinti	6	3	4	8	4	11	6	5	2	49
			4	5	4	4	5	1	2	1	-	26
		Noktalama				5	6	4	5	4	3	27
						2	1	-	2	-	-	5
		Harf Hatası	2	1	-							3
				-								-
		Parantez	1	1	3							5
				-								-
	Anlamsal	Yanlış Kullanma	3	-	2	2	5	-	4	4	2	22
			2	2	1	2	3	-	1	-	3	14
		Eksik Kullanma	1	-	2	2	2	3	4	2	3	19
			1	-	-	1	-	-	3	2	3	10
		Belirleyememe							1	1	2	4
										-		-
		Karar Verememe	2	1	2	3	2	2	4	3	5	24
			3	3	1	3	2	2	1	-	-	15
	Stratejik	Yanlış Kullanma	5	5	4	2	2	2	2	4	4	30
			6	4	5		-		3	2	2	22
		Eksik Kullanma	3	4	4				2	3	4	20
				-					3	2	2	7
		Belirleyememe	4	3	4	3	3	3	1	1	2	24
			3	-	4		-		2	1	1	11
		Karar Verememe	-	-	3				1	1	2	7
			5	6	1							12
		Toplam	133			100			123			356

Tablo 23 incelendiğinde sözdizimsel, anlamsal ve stratejik bilgi türlerinde zorluklarla karşılaşıldığı ve birçok hata yapıldığı görülmektedir. Bu zorlanmaların büyük bir kısmının stratejik bilgi türünde(133) olduğu belirlenmiştir. Bu bilgi türünü sözdizimsel bilgi türü(115) takip ederken son olarak anlamsal bilgi türünde(108) zorlanmalar olduğu görülmektedir. Bu problemin çözümü için fonksiyon dışında ele alınan tüm yapıların(değişken atama, operatör, koşul ve döngü) kullanılması gerekmektedir. Kullanılan bu yapılara sözdizimsel boyutta bakıldığında değişken atama(34), koşul(51) ve döngü(30) yapılarında zorlanmalar ve hatalarla karşılaşıldığı görülmektedir. Bu durumda sözdizimsel boyutta karşılaşılan zorluklar çoğunlukla koşul yapılarında rastlanmıştır.

Karşılaşılan zorlanmaların bir kısmının(33) hemen çözüldüğü(Ç), bir kısmının(24) biraz düşünülüp çözüldüğü(DÇ), diğer kısmının(27) uğraşılıp çözüldüğü(UÇ) tespit edilmiştir. Karşılaşılan hataların bir kısmının(14) durup çözülemediği(D), bir kısmının(12)

düşünülüp bulunamadığı(DB) diğer kısmının(5) uğraşılıp bulunamadığı(UB) durumlar olduğu görülmektedir. Bu durumda karşılaşılan zorlukların ve hataların yaklaşık %73'ü giderilirken %27'si giderilememiştir. Bu zorlanmalar özellikle *girinti* ve *noktalama* türünde zorluk yaşandığını göstermektedir. Bu zorlanmanın öğrencilere yansıması şu şekildedir. Örneğin;



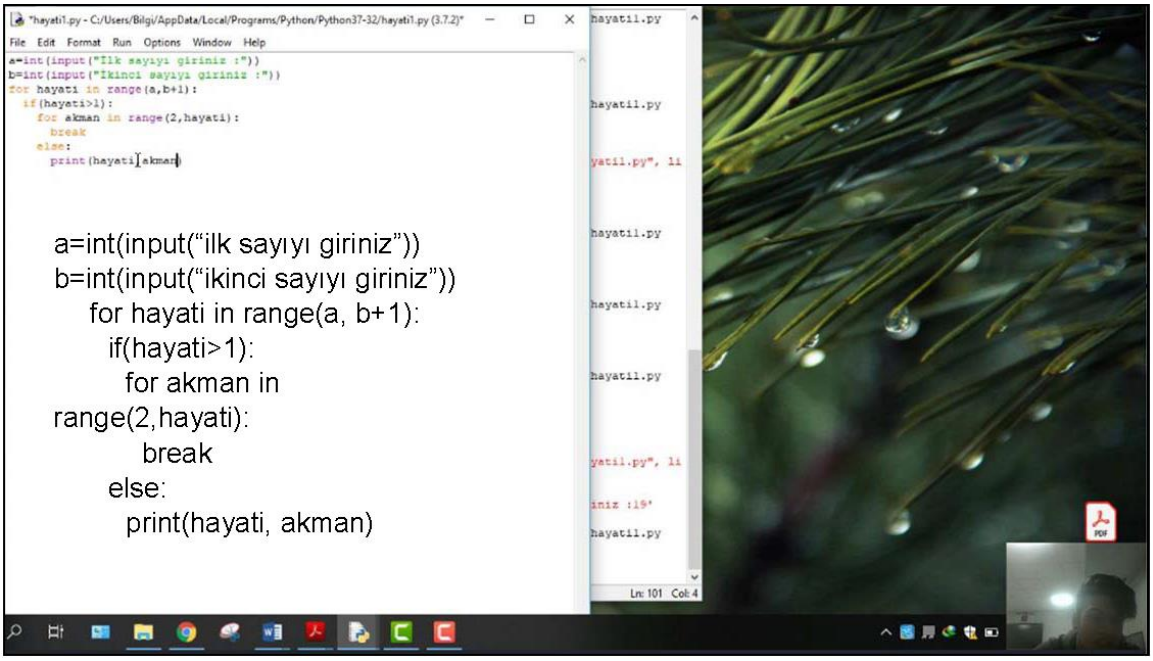
Şekil 25. Ö1 problem 3 çözümü ve aldığı hata mesajı

Yukarıda Ö1'in 3. problemin çözümü esnasında çekilmiş olan ekran görüntüsü yer almaktadır. Ö1, Şekil 25'te görüldüğü gibi değişken atamada girinti hatası yaptığından dolayı "*beklenmeyen girinti"* hatası almıştır. Hata aldıktan sonra 'şaşkın' ve hatanın nereden kaynaklandığını anlamaya çalışan yüz ifadesiyle ekrana kilitlenen öğrenci tepkisini "Nerede hata verdi. Bir dakika ya…" sesli düşünme ifadesiyle ortaya koymuştur. Hatayı gidermek adına satırı bir tık geriye çekmiş ve programı tekrar çalıştırmıştır. Böylelikle hatayı çözmüştür(Ç). Uygulamadan sonra yapılan mülakatta Ö1 bu problem çözümü sırasındaki davranışlarını değerlendirirken "*Hep hata veriyor ya! Hata verince zorlanıyorum.*" şeklinde yaşadığı zorluğu ifade etmiştir.

Problem çözümünde kullanılması gereken programlama yapılarına anlamsal boyutta bakıldığında değişken atama(26), koşul(68) ve döngü(30) yapılarında zorlanmalar olduğu görülmektedir. Bu durumda anlamsal boyutta karşılaşılan zorluklar çoğunlukla koşul yapısında rastlanmıştır.

Karşılaşılan zorlanmaların bir kısmının(26) hemen çözüldüğü(Ç), bir kısmının(20) biraz düşünülüp çözüldüğü(DÇ), diğer kısmının(23) uğraşılıp çözüldüğü(UÇ) tespit

edilmiştir. Karşılaşılan hataların bir kısmının(17) durup çözülemediği(D), bir kısmının(12) düşünülüp bulunamadığı(DB) diğer kısmının(10) uğraşılıp bulunamadığı(UB) durumlar olduğu görülmektedir. Bu durumda karşılaşılan zorlukların ve hataların yaklaşık %64'ü giderilirken %36'sı giderilememiştir. Bu zorlanmalar özellikle *yanlış kullanma* ve *eksik kullanma* türlerinde zorluk yaşandığını göstermektedir. Bu zorluk durumları öğrenciler tarafından aşağıdaki gibi örneklendirilebilir. Örneğin;



Şekil 26. Ö1 problem 3 çözümü

Yukarıda Ö1'in 3. Problemin çözümü esnasında çekilmiş olan ekran görüntüsü yer almaktadır. Ö1, problemin çözümü için yazdırma komutu(print) içerisinde yanlış değişkeni yazdırmıştır. Bu anlamda problemin doğru çözümüne ulaşma noktasında planlama hatası yapmıştır. Ekrana kilitlenmiş ve hatayı giderebilmek için farklı denemeler yapmıştır. Hatayı gidermek için defalarca uğraşmış ancak hatayı gideremeyiştir(UB). Ö1 sürekli hata gidermekle uğraşmış, gideremeyince tepkisini "*Bunu bana yapma, burada da hata verme yaa…*" sesli düşünme ifadesiyle ortaya koymuştur. Uygulamadan sonra yapılan mülakatta Ö3 bu problem çözümü sırasındaki davranışlarını değerlendirirken "*Ya deniyorum deniyorum olmuyor ya, bir yerde bir hata var ama"* şeklinde yaşadığı zorluğu anlatarak kafasını sağa sola sallamış ve çözemediğini ifade etmiştir.

Bu yapılar stratejik boyutta incelendiğinde değişken atama(73), koşul(15) ve döngü(45) yapılarında zorlanmalar olduğu görülmektedir. Bu durumda stratejik boyutta karşılaşılan zorluklar çoğunlukla değişken atama ve döngü yapısında rastlanmıştır.

Karşılaşılan zorlanmaların bir kısmının(23) hemen çözüldüğü(Ç), bir kısmının(26) biraz düşünülüp çözüldüğü(DÇ), diğer kısmının(32) uğraşılıp çözüldüğü(UÇ) tespit edilmiştir. Karşılaşılan hataların bir kısmının(22) durup çözülemediği(D), bir kısmının(15) düşünülüp bulunamadığı(DB) diğer kısmının(15) uğraşılıp bulunamadığı(UB) durumlar olduğu görülmektedir. Bu durumda karşılaşılan zorlukların ve hataların yaklaşık %61'i giderilirken %39'u giderilememiştir. Bu durum problem çözülürken hangi yolun kullanılması gerektiği konusunda özellikle *yanlış yerde kullanma*, *eksik kullanma*, *belirleyememe ve karar verememe* türlerinde zorlanıldığını göstermektedir. Bu zorlanmanın öğrencilere yansıması şu şekildedir. Örneğin;

Ö4: Bu problemin çözümü için 2 koşul kullanılmalıdır. Ö4 yalnızca bir koşul kullanmış ve kullanması gereken diğer koşulu kullanmamıştır. Program çalıştırıldığında doğru sonuç vermemiştir. Ö4 kodları eksik yazdığını fark etmemiş, hatayı yazmış olduğu kodlarda aramıştır. Bir süre hiç konuşmadan düşünmüş ve ekrandaki kodlara odaklanmıştır. Farklı satırlarda değişiklikler yapmış, tekrar eski hale getirmiştir. Defalarca programı çalıştırmış ancak hep aynı sonuçla karşılaşmıştır. 'Derin bir nefes' alarak "*Yok çalışmıyor.*" diyerek sesli düşünmüş ve uygulamayı bitirmiştir. Çok uğraşmasına rağmen doğru çözümü yapamamıştır(UB). Uygulama sonrasında yapılan mülakatta Ö4 "*Yapamadım, nerede yanlış yaptım…Anlamadım.*" şeklinde yaşadığı zorluğu ifade etmiştir.

Problem 4: *Kullanıcının girdiği 2 sayı arasındaki çift sayıların ortalamasını bulan programı yazınız.*

4. problemde karşılaşılan zorlanmalar ve hatalar Tablo 24'te gösterilmiştir.

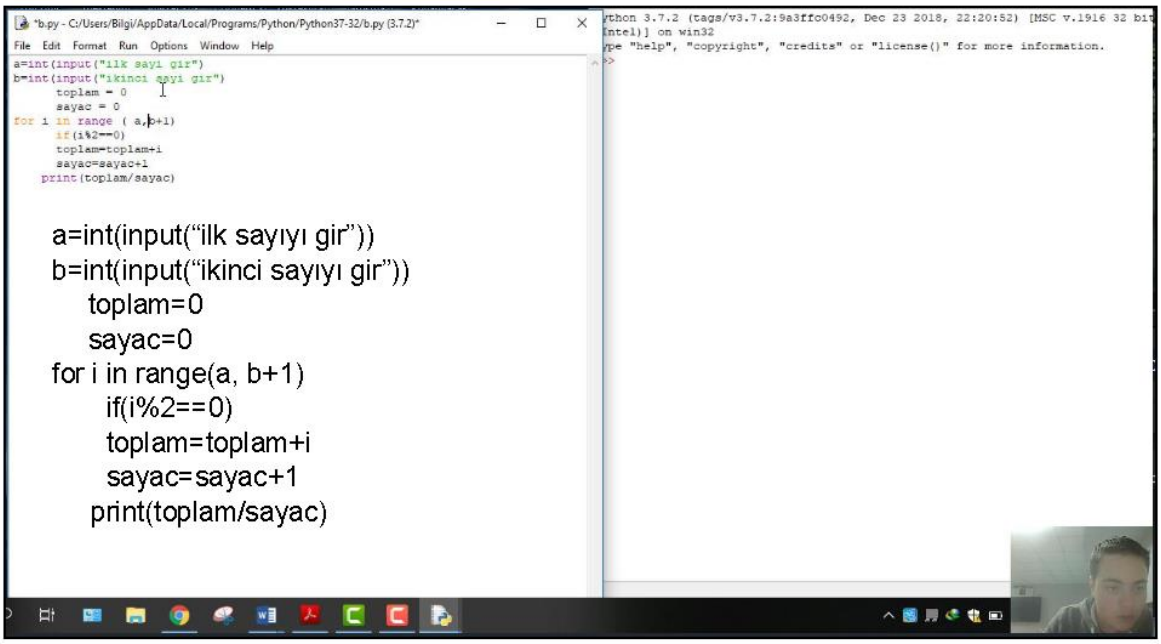
Tablo 24. Problem 4 Hata ve Zorluk Kaynakları

			Programlama Yapıları									Toplam
	Bilgi Türleri	Hata ve Zorluk Türleri	Değişken Atama			Koşul			Döngü			
Problem_4	Söz dizimsel	Girinti	3	4	3	16	12	9	5	4	4	60
			6	4	3	2	3	2	2	1	-	23
		Noktalama				6	4	5	4	4	-	23
						1	-	-	1	-	-	2
	Anlamsal	Yanlış Kullanma	2	4	4	3	2	-	3	4	4	26
			2	-	-	3	2	1	2	-	-	10
		Eksik Kullanma	2	3	2	3	4	2	2	4	4	26
			-			3	1	-	3	2	2	11
		Belirleyememe	2	2	1				1	1	1	8
			-	1	1					-		2
		Karar Verememe	-	-	3	3	3	1	4	4	5	23
			-	-	1	3	4	1	2	1	1	13
	Stratejik	Yanlış Kullanma	5	6	7	5	5	2	5	2	3	40
			7	5	3		-		3	2	2	22
		Eksik Kullanma	2	5	5	4	3	4	4	4	2	33
			1	2	-	2	2	1	2	1	1	12
		Belirleyememe	2	3	4	2	4	3	3	3	2	26
			-	2	1	2	1	1	2	2	1	12
		Karar Verememe	3	2	4	3	2	2	3	2	1	22
			-	4	3		-		1	-	-	8
		Toplam	129			147			126			402

Tablo 24 incelendiğinde sözdizimsel, anlamsal ve stratejik bilgi türlerinde zorluklarla karşılaşıldığı ve birçok hata yapıldığı görülmektedir. Bu zorlanmaların büyük bir kısmının stratejik bilgi türünde(175) olduğu belirlenmiştir. Bu bilgi türünü anlamsal bilgi türü(119) takip ederken son olarak sözdizimsel bilgi türünde(108) zorlanmalar olduğu görülmektedir. Bu problemin çözümü için fonksiyon dışında ele alınan tüm yapıların(değişken atama, operatör, koşul ve döngü) kullanılması gerekmektedir. Kullanılan bu yapılar sözdizimsel boyutta ele alındığında; değişken atama(23), koşul(60) ve döngü(25) yapılarında zorlanmalar ve hatalarla karşılaşıldığı görülmektedir. Bu durumda sözdizimsel boyutta karşılaşılan zorluklara çoğunlukla koşul yapısında rastlanmıştır.

Karşılaşılan zorlukların bir kısmının(34) hemen çözüldüğü(Ç), bir kısmının(28) biraz düşünülüp çözüldüğü(DÇ), diğer kısmının(21) uğraşılıp çözüldüğü(UÇ) tespit edilmiştir. Karşılaşılan hataların bir kısmının(12) durup çözülemediği(D), bir kısmının(8) düşünülüp bulunamadığı(DB) diğer kısmının(5) uğraşılıp bulunamadığı(UB) durumlar olduğu görülmektedir. Bu durumda karşılaşılan zorlukların ve hataların yaklaşık %77'si

giderilirken %23'ü giderilememiştir. Bu zorlanmalar özellikle *girinti* ve *noktalama* türünde zorluk yaşandığını göstermektedir. Bu zorlanmanın öğrencilere yansıması şu şekildedir. Örneğin;

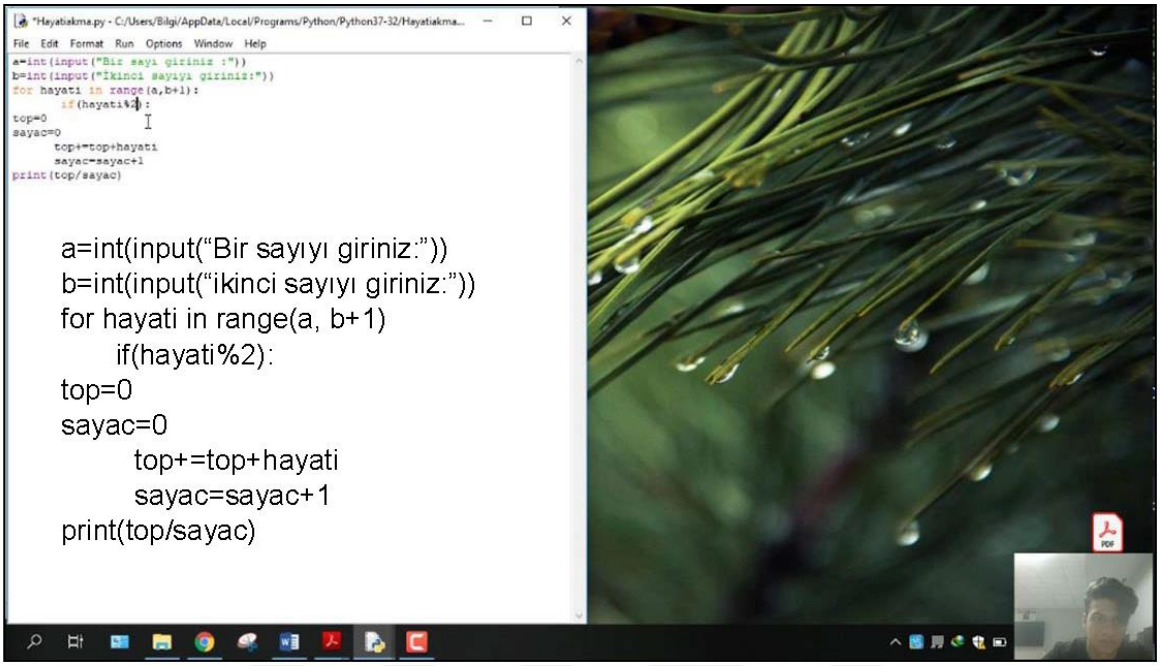


Şekil 27. Ö3 problem 4 çözümü

Yukarıda Ö3'ün 4. Problemin çözümü esnasında çekilmiş olan ekran görüntüsü yer almaktadır. Ö3, problemin çözümü için değerlerini kullanıcının girmesini istediği 2 değişken(a,b) tanımlamıştır. Bu değişkenlerin sonunda kapatması gereken *parantezi* unuttuğu için sözdizimi hatası almıştır. Ö3 tepkisini "*Ha parantezleri unuttuk*." şeklinde sesli düşünerek göstermiştir. Hatayı alır almaz unuttuğunu fark etmiş ve hatayı gidermiştir(Ç). Uygulamadan sonra yapılan mülakatta Ö3 bu problem çözümü sırasındaki davranışlarını değerlendirirken "*Programı çalıştırdığımda birkaç kez hata verdi ama hallettik onları. Bazı hatalar uğraştırdı.*" şeklinde yaşadığı zorluğu ifade etmiştir.

Problem çözümünde kullanılması gereken programlama yapılarına anlamsal boyutta bakıldığında değişken atama(30), koşul(39) ve döngü(50) yapılarında zorlanmalar olduğu görülmektedir. Bu durumda anlamsal boyutta karşılaşılan zorluklar çoğunlukla döngü yapısında rastlanmıştır. Karşılaşılan zorlanmaların bir kısmının(25) hemen çözüldüğü(Ç), bir kısmının(31) biraz düşünülüp çözüldüğü(DÇ), diğer kısmının(27) uğraşılıp çözüldüğü(UÇ) tespit edilmiştir. Karşılaşılan hataların bir kısmının(18) durup çözülemediği(D), bir kısmının(11) düşünülüp bulunamadığı(DB) diğer kısmının(7) uğraşılıp bulunamadığı(UB) durumlar olduğu görülmektedir. Bu durumda karşılaşılan zorlukların ve hataların yaklaşık %70'i giderilirken %30'u giderilememiştir. Bu zorlanmalar özellikle

yanlış kullanma, *eksik kullanma* ve *karar verememe* türlerinde zorluk yaşandığını göstermektedir. Bu zorlanmanın öğrencilere yansıması şu şekildedir. Örneğin;



Şekil 28. Ö1 problem 4 çözümü

Yukarıda Ö1'in 4. Problemin çözümü esnasında çekilmiş olan ekran görüntüsü yer almaktadır. Ö1, problemin çözümü için yaptığı değişken atama satırında(top+=top+hayati) hata yapmıştır. İlk aşamada yaptığı bu hatayı fark etmemiştir. Programı çalıştırdığında bu satırda hata alınca kodları incelemeye başlamış ve hatanın neden kaynaklandığını bulmaya çalışmıştır. Bir süre düşündükten sonra hatayı gidermeyi(Ç) başarmıştır. Uygulama boyunca konuşan Ö1 bu hata karşısındaki tepkisini "*Sen de hata ver*." şeklinde sesli düşünerek göstermiştir. Uygulamadan sonra yapılan mülakatta Ö1 bu problem çözümü sırasındaki davranışlarını değerlendirirken "*Öncelikle tüm arkadaşlara sesleniyorum, yazılım okumayın arkadaşlar. Ya hep hata veriyor.*" şeklinde yaşadığı zorluğu ifade etmiştir.

Bu yapılar stratejik boyutta incelendiğinde değişken atama(76), koşul(48) ve döngü(51) yapılarında zorlanmalar olduğu görülmektedir. Bu durumda stratejik boyutta karşılaşılan zorluklar çoğunlukla değişken atama yapısında rastlanmıştır.

Karşılaşılan zorlanmaların bir kısmının(41) hemen çözüldüğü(Ç), bir kısmının(41) biraz düşünülüp çözüldüğü(DÇ), diğer kısmının(39) uğraşılıp çözüldüğü(UÇ) tespit edilmiştir. Karşılaşılan hataların bir kısmının(20) durup çözülemediği(D), bir kısmının(21) düşünülüp bulunamadığı(DB) diğer kısmının(13) uğraşılıp bulunamadığı(UB) durumlar olduğu görülmektedir. Bu durumda karşılaşılan zorlukların ve hataların yaklaşık %69'u

giderilirken %31'i giderilememiştir. Bu durum problem çözülürken hangi yolun kullanılması gerektiği konusunda özellikle *yanlış yerde kullanma*, *eksik kullanma*, *belirleyememe ve karar verememe* türlerinde zorlanıldığını göstermektedir. Bu zorlanmanın öğrencilere yansıması şu şekildedir. Örneğin;

Şekil 28'de Ö1'in 4.problemin çözümü esnasındaki yazmış olduğu kodlar ve kendisinin ekran görüntüsü yer almaktadır. Ö1 tanımlamış olduğu 2 değişkeni(top, sayac) döngü ve koşul komutlarından önce tanımlaması gerekirken sonra tanımlamıştır. Bu nedenle yanlış yerde kullanım hatası yapmıştır. Programı çalıştırdığında bu satırda hata almıştır. "*Burada ne yanlış ki, bunu doğru yaptım. Allah Allah!*" diyerek sesli düşünmüş ve ekrana odaklanarak yazmış olduğu kodları incelemeye başlamıştır. Hatayı gidermek için farklı hamleler yapmış ancak ilk aşamada gideremediğini. "*Bu değişkenleri burada tanımlamayacağız sanırım, bu yüzden hata veriyor.*" şeklinde sesli düşünmüş ve uzun uğraşlar sonucu hatayı gidermiştir(UÇ). Uygulamadan sonra yapılan mülakatta Ö1 bu problem çözümü sırasındaki davranışlarını değerlendirirken "*Öncelikle tüm arkadaşlara sesleniyorum, yazılım okumayın arkadaşlar. Ya hep hata veriyor.*" şeklinde yaşadığı zorluğu ifade etmiştir.

Problem 5*: Bir sayının kendisi dışında bütün pozitif bölenlerinin toplamı kendisine eşit olan sayılara mükemmel sayı denir. Kullanıcının girdiği sayının mükemmel sayı olup olmadığını kontrol eden kodu yazınız.*

5. problemde karşılaşılan zorlanmalar ve hatalar Tablo 25'te gösterilmiştir.

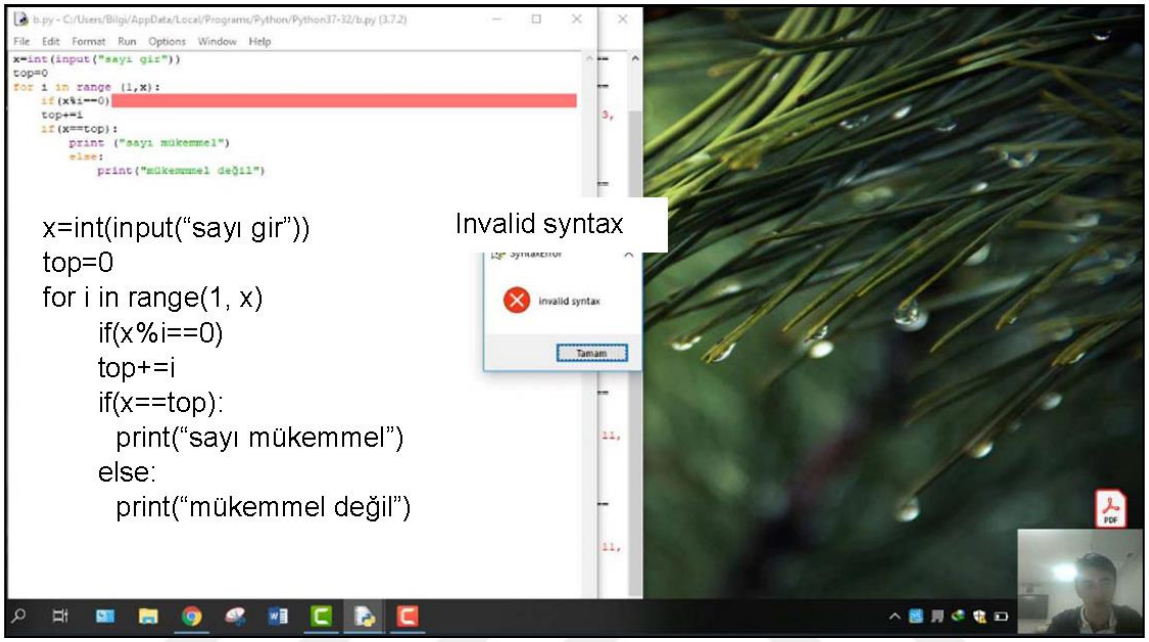
Tablo 25. Problem 5 Hata ve Zorluk Kaynakları

			Programlama Yapıları									Toplam
	Bilgi Türleri	Hata ve Zorluk Türleri	Değişken Atama			Koşul			Döngü			
Problem_5	Sözdizimsel	Girinti	6	4	4	10	11	7	6	3	1	52
			5	3	2	2	2	3	1	-	1	19
		Noktalama				6	6	4	5	-	-	21
							-			-		-
	Anlamsal	Yanlış Kullanma	2	-	-	2	3	3	3	3	4	20
			2	1	-	1	2	-	-	-	1	7
		Eksik Kullanma	3	3	2	2	1	2	3	3	4	23
			3	2	-		-		2	2	2	11
		Belirleyememe	2	-	-				3	2	2	9
			1	1	2				1	1	2	8
		Karar Verememe	-	4	2	5	3	2	3	3	2	24
			2	2	-	4	4	2	3	2	-	19
	Stratejik	Yanlış Kullanma	4	6	4	3	2	-	4	2	1	26
			5	6	6	2	1	1	2	2	1	26
		Eksik Kullanma	4	3	3	3	3	2	4	4	2	28
				-		1	1	-	-	1	1	4
		Belirleyememe	5	6	3	3	1	3	3	4	4	32
			-	3	3	2	2	3	-	2	1	16
		Karar Verememe	1	2	2	3	4	2	1	-	-	15
			2	2	2	2	1	-	-	2	2	13
		Toplam	130			132			111			373

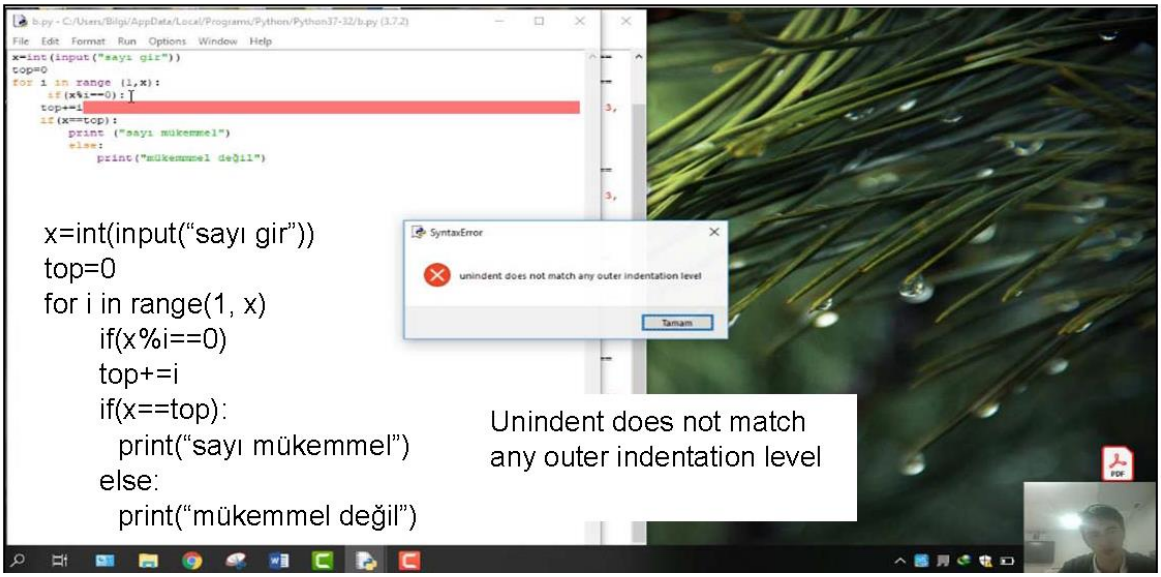
Tablo 25 incelendiğinde sözdizimsel, anlamsal ve stratejik bilgi türlerinde zorluklarla karşılaşıldığı ve birçok hata yapıldığı görülmektedir. Bu zorlanmaların büyük bir kısmının stratejik bilgi türünde(160) olduğu belirlenmiştir. Bu bilgi türünü anlamsal bilgi türü(121) takip ederken son olarak sözdizimsel bilgi türünde(92) zorlanmalar olduğu görülmektedir. Bu problemin çözümü için fonksiyon dışında ele alınan tüm yapıların(değişken atama, operatör, koşul, döngü ve fonksiyon) kullanılması gerekmektedir. Kullanılan bu yapılara sözdizimsel boyutta bakıldığında değişken atama(24), koşul(51) ve döngü(17) yapılarında zorlanmalar ve hatalarla karşılaşıldığı görülmektedir. Bu durumda sözdizimsel boyutta karşılaşılan zorluklara çoğunlukla koşul yapısında rastlanmıştır.

Karşılaşılan zorlanmaların bir kısmının(33) hemen çözüldüğü(Ç), bir kısmının(24) biraz düşünülüp çözüldüğü(DÇ), diğer kısmının(16) uğraşılıp çözüldüğü(UÇ) tespit edilmiştir. Karşılaşılan hataların bir kısmının(8) durup çözülemediği(D), bir kısmının(5) düşünülüp bulunamadığı(DB) diğer kısmının(6) uğraşılıp bulunamadığı(UB) durumlar

olduğu görülmektedir. Bu durumda karşılaşılan zorlukların ve hataların yaklaşık %68'i giderilirken %32'si giderilememiştir. Bu zorlanmalar özellikle *girinti* ve *noktalama* türünde zorluk yaşandığını göstermektedir. Bu zorlanmanın öğrencilere yansıması şu şekildedir. Örneğin;



Şekil 29. Ö4 problem 5 çözümü ve aldığı hata mesajı



Şekil 30. Ö4 problem 5 çözümü ve aldığı hata mesajı

Yukarıda Ö4'ün 5. problemin çözümü esnasında çekilmiş olan ekran görüntüsü yer almaktadır. Ö4, Şekil 29'da görüldüğü gibi koşul satırının sonuna iki nokta karakterini

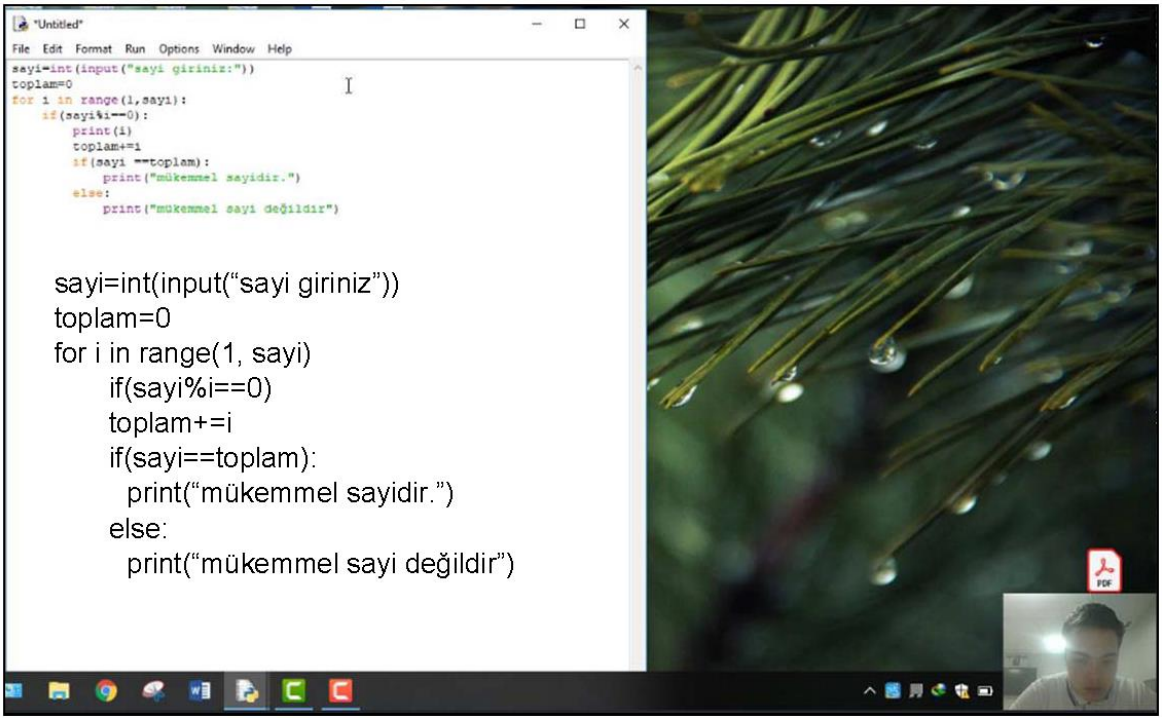
koymadığından dolayı *noktalama* hatası yapmıştır. Programı çalıştırdığında "*invalid syntax/geçersiz sözdizimi"* hatası almıştır. Hata aldıktan sonra hatanın nereden kaynaklandığını anlamaya çalışan yüz ifadesiyle ekrana kilitlenmiş ve gözleriyle hatayı anlamaya çalışmıştır. "*Bir yerde hata verdi*." diyerek sesli düşünmüştür. İki nokta karakterini koymadığını fark edip hatayı gidermiştir(Ç). Şekil 30'da değişken atamada girinti hatası yaptığından dolayı "*unındent does not match any outer indentation level/dış girinti ile uyuşmuyor"* hatası almıştır. Bu hatayı anlamakta biraz zorluk çekmiştir. Programın İngilizce hata vermesi de hatanın anlaşılmasını zorlaştırmaktadır. Ö4 tepkisini "*Niye hata veriyor. Kodlar doğru bence*" sesli düşünme ifadesiyle ortaya koymuş ve 'derin bir nefes' alarak programı tekrar çalıştırmıştır. Ancak yine aynı hatayı almıştır. Hatanın nedenini anlayamamış ve öylece durup(D) çözümü bırakmıştır. Uygulamadan sonra yapılan mülakatta "*Problem biraz zordu, bazı yerlerde hata verdi. Niye çalışmadı anlamadım, kodları doğru yazdım ama hata verdi."* şeklinde yaşadığı zorluğu ifade etmiştir.

Problem çözümünde kullanılması gereken programlama yapılarına anlamsal boyutta bakıldığında değişken atama(34), koşul(36) ve döngü(51) yapılarında zorlanmalar olduğu görülmektedir. Bu durumda anlamsal boyutta karşılaşılan zorluklara çoğunlukla döngü yapısında rastlanmıştır. Karşılaşılan zorlanmaların bir kısmının(33) hemen çözüldüğü(Ç), bir kısmının(28) biraz düşünülüp çözüldüğü(DÇ), diğer kısmının(23) uğraşılıp çözüldüğü(UÇ) tespit edilmiştir. Karşılaşılan hataların bir kısmının(19) durup çözülemediği(D), bir kısmının(17) düşünülüp bulunamadığı(DB) diğer kısmının(9) uğraşılıp bulunamadığı(UB) durumlar olduğu görülmektedir. Bu durumda karşılaşılan zorlukların ve hataların yaklaşık %69'u giderilirken %31'i giderilememiştir. Bu zorlanmalar özellikle *yanlış kullanma, karar verememe* ve *eksik kullanma* türlerinde zorluk yaşandığını göstermektedir. Bu zorlanmanın öğrencilere yansıması şu şekildedir. Örneğin;

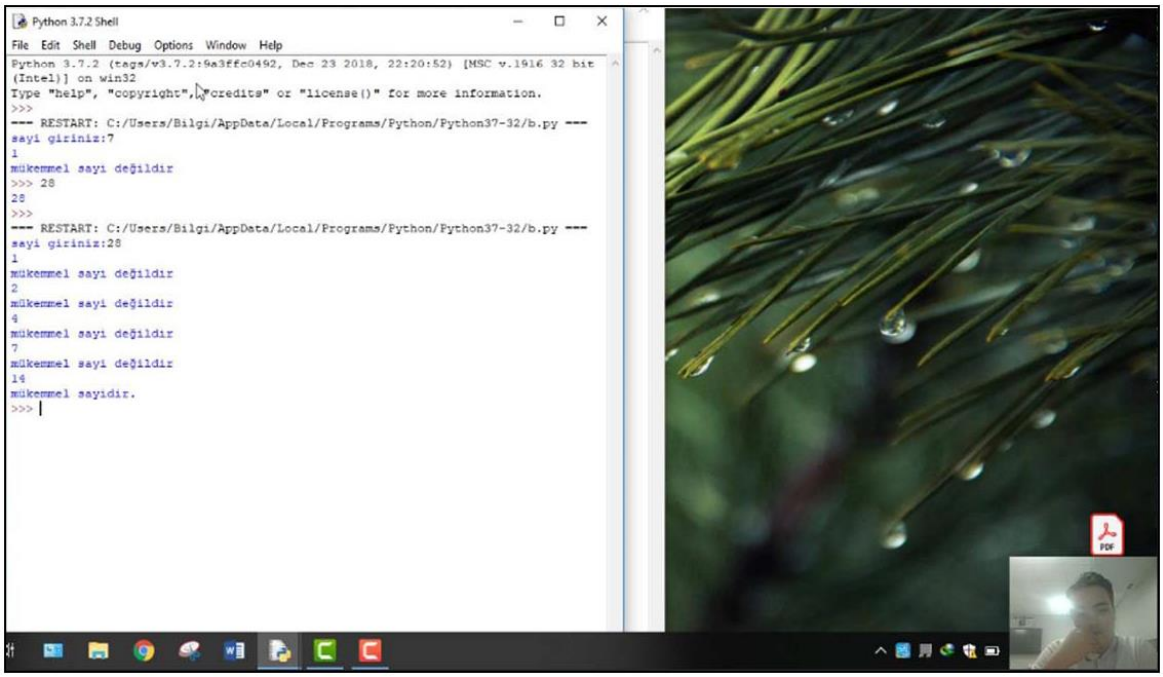
Ö2: Problemin çözümü için koşul içinde kullanması gereken değişkeni kullanmamıştır. *Yanlış kullanım* hatası yaptığından dolayı program doğru çalışmamıştır. Ö2 şaşkın gözlerle ekrana kilitlenmiş ve nerede hata yaptığını bulmaya çalışmıştır. Tepkisini "*Nerede hata olabilir, hata görünmüyor.*" şeklinde sesli düşünerek göstermiştir. Bir süre kodları inceleyerek düşünmüş ancak bulamadığını söyleyerek uygulamayı bitirmiştir. Böylelikle düşünmesine rağmen hatayı gideremeyiştir(DB). Uygulamadan sonra yapılan mülakatta "*Program çalışmadı. Bir yerde hata vardı ama bulamadım. Bence her şey doğruydu."* şeklinde yaşadığı zorluğu ifade etmiştir.

Bu yapılar stratejik boyutta incelendiğinde değişken atama(72), koşul(45) ve döngü(43) yapılarında zorlanmalar olduğu görülmektedir. Bu durumda stratejik boyutta karşılaşılan zorluklar çoğunlukla değişken atama yapısında rastlanmıştır.

Karşılaşılan zorlanmaların bir kısmının(38) hemen çözüldüğü(Ç), bir kısmının(37) biraz düşünülüp çözüldüğü(DÇ), diğer kısmının(26) uğraşılıp çözüldüğü(UÇ) tespit edilmiştir. Karşılaşılan hataların bir kısmının(15) durup çözülemediği(D), bir kısmının(23) düşünülüp bulunamadığı(DB) diğer kısmının(26) uğraşılıp bulunamadığı(UB) durumlar olduğu görülmektedir. Bu durumda karşılaşılan zorlukların ve hataların yaklaşık %62'ü giderilirken %38'i giderilememiştir. Bu durum problem çözülürken hangi yolun kullanılması gerektiği konusunda özellikle *yanlış yerde kullanma*, *eksik kullanma*, *belirleyememe ve karar verememe* türlerinde zorlanıldığını göstermektedir. Bu zorlanmanın öğrencilere yansıması şu şekildedir. Örneğin;



Şekil 31. Ö3 problem 5 çözümü



Şekil 32. Ö3 problem 5 çözümü çıktı ekranı

Yukarıda Ö3'ün 5. problemin çözümü esnasında çekilmiş olan ekran görüntüsü yer almaktadır. Ö3, problemin çözümünde *yanlış kullanım* hatası yaptığından dolayı program çalıştırıldığında doğru sonuç vermemiştir. Ö3 tepkisini "*Ya nerde hata var, bakıyorum bakıyorum hata göremiyorum, niye olmuyor.*" sesli düşünme ifadesiyle ortaya koymuş ve Şekil 32'de de görüldüğü üzere elini ağzına götürerek hatanın nerede olduğunu bulmaya çalışmıştır. Bir süre ekrana kilitlenmiştir. Uzun uzun düşünmesine rağmen hatayı gideremeyiştir(DB). Uygulamadan sonra yapılan mülakatta "*Yaptım ama doğru sonuç vermiyor. Ben de anlamadım. Hatanın nerede olduğunu bulamadığım için biraz zorlandım.*" şeklinde yaşadığı zorluğu ifade etmiştir.

Problem 6*: Kullanıcının girdiği sayının pozitif tam bölenlerini bulan programı fonksiyon kullanarak yazınız.*

6. problemde karşılaşılan zorlanmalar ve hatalar Tablo 26'da gösterilmiştir.

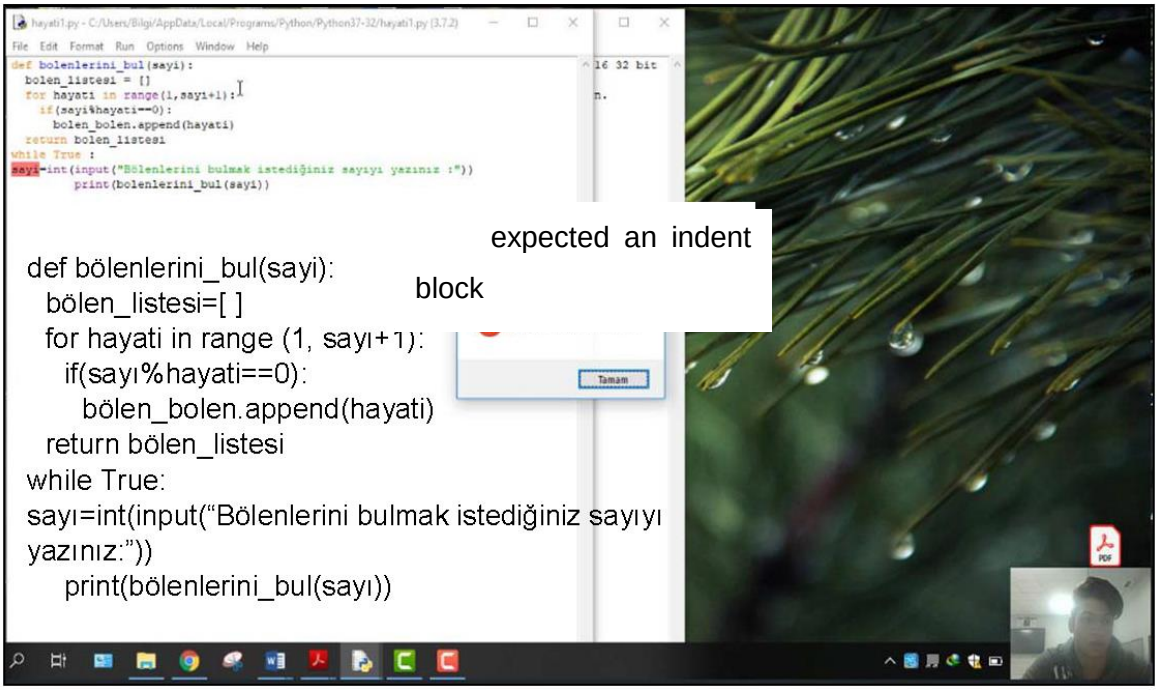
Tablo 26. Problem 6 Hata ve Zorluk Kaynakları

			Programlama Yapıları												
	Bilgi Türleri	Hata ve Zorluk Türleri	Değişken Atama			Koşul			Döngü			Fonksiyon			Toplam
Problem_6	Sözdizimsel	Girinti	5	4	3	11	9	8	7	5	2	4	7	4	69
			-	2	2	-	-	1		-		1	1	-	7
		Noktalama				-	4	3	3	1	1	4	2	1	19
							-			-			-		-
		Parantez	1	1	-										2
				-											-
	Anlamsal	Yanlış Kullanma	3	3	2	3	1	1	4	4	2	4	1	2	30
			1	-	1	-	-	1	2	1	1	1	1	4	13
		Eksik Kullanma	3	1	-	3	2	1	4	6	3	2	1	-	26
			3	3	1	2	1	-	2	4	3		-		19
		Belirleyememe	2	-	4	3	-	-	-	-	1	2	1	1	14
			2	-	-		-		1	1	1	2	-	-	7
		Karar Verememe	3	4	3	3	3	3	4	3	4				30
			2	1	-	2	2	1	2	-	-				10
	Stratejik	Yanlış Kullanma	6	5	5	4	4	4	3	2	5	2	3	3	46
			7	5	6	-	-	1	1	1	-		-		21
		Eksik Kullanma	2	3	5	-	-	2	3	2	3	4	2	2	28
			1	2	1	1	2	-	1	2	3		-		13
		Belirleyememe	2	3	4	3	2	4	5	4	1	2	2	2	34
			2	2	-	2	-	1	2	1	1	-	-	3	14
		Karar Verememe	3	4	-	3	2	1	2	1	1				17
			3	-	-	2	1	1	1	1	-				9
	Toplam		131			108			118			71			428

Tablo 26 incelendiğinde sözdizimsel, anlamsal ve stratejik bilgi türlerinde zorluklarla karşılaşıldığı ve birçok hata yapıldığı görülmektedir. Bu zorlanmaların büyük bir kısmının stratejik bilgi türünde(182) olduğu belirlenmiştir. Bu bilgi türünü anlamsal bilgi türü(149) takip ederken son olarak sözdizimsel bilgi türünde(97) zorlanmalar olduğu görülmektedir. Bu problemin çözümü için ele alınan tüm yapıların(değişken atama, operatör, koşul, döngü ve fonksiyon) kullanılması gerekmektedir. Kullanılan bu yapılara sözdizimsel boyutta bakıldığında değişken atama(18), koşul(36), döngü(19) ve fonksiyon(24) yapılarında zorlanmalar ve hatalarla karşılaşıldığı görülmektedir. Bu durumda sözdizimsel boyutta karşılaşılan zorluklara çoğunlukla koşul yapısında rastlanmıştır.

Karşılaşılan zorlanmaların bir kısmının(35) hemen çözüldüğü(Ç), bir kısmının(33) biraz düşünülüp çözüldüğü(DÇ), diğer kısmının(22) uğraşılıp çözüldüğü(UÇ) tespit edilmiştir. Karşılaşılan hataların bir kısmının(1) durup çözülemediği(D), bir kısmının(3) düşünülüp bulunamadığı(DB) diğer kısmının(3) uğraşılıp bulunamadığı(UB) durumlar olduğu görülmektedir. Bu durumda karşılaşılan zorlukların ve hataların yaklaşık %93'ü giderilirken %7'si giderilememiştir. Bu zorlanmalar özellikle *girinti* ve *noktalama* türünde

zorluk yaşandığını göstermektedir. Bu zorlanmanın öğrencilere yansıması şu şekildedir. Örneğin;



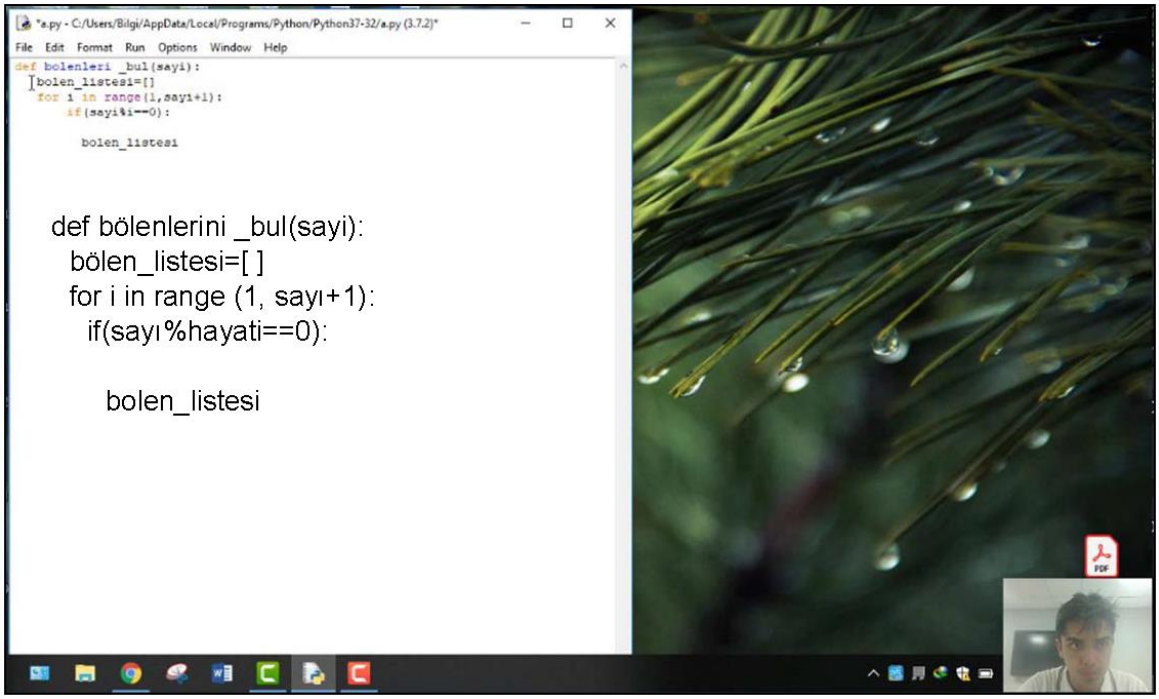
Şekil 33. Ö1 problem 6 çözümü ve aldığı hata mesajı

Yukarıda Ö1'in 6. problemin çözümü esnasında çekilmiş olan ekran görüntüsü yer almaktadır. Ö1, değişken atamada girinti hatası yaptığından dolayı "*expected an indent block/girintili bir blok bekleniyor"* hatası almıştır. Hata aldıktan sonra hatanın nereden kaynaklandığını anlamaya çalışan yüz ifadesiyle ekrana kilitlenmiş ve gözleriyle hatayı anlamaya çalışmıştır. Ö1 tepkisini "*Hayda yine niye hata verdi.*" sesli düşünme ifadesiyle ortaya koymuş ve ekranda incelemeler yapmaya devam etmiştir. Hatayı gidermek için değişkeni bir tık geriye almış ve programı tekrar çalıştırmıştır. Çok uğraşmış ve hatayı gidermeyi başarmıştır(UB). Uygulamadan sonra yapılan mülakatta "*Çok hata veriyor, çok uğraştırıyor. "* şeklinde yaşadığı zorluğu ifade etmiştir.

Problem çözümünde kullanılması gereken programlama yapılarına anlamsal boyutta bakıldığında değişken atama(42), koşul(32), döngü(53) ve fonksiyon(22) yapılarında zorlanmalar olduğu görülmektedir. Bu durumda anlamsal boyutta karşılaşılan zorluklara çoğunlukla değişken atama ve döngü yapısında rastlanmıştır.

Karşılaşılan zorlanmaların bir kısmının(43) hemen çözüldüğü(Ç), bir kısmının(30) biraz düşünülüp çözüldüğü(DÇ), diğer kısmının(27) uğraşılıp çözüldüğü(UÇ) tespit edilmiştir. Karşılaşılan hataların bir kısmının(22) durup çözülemediği(D), bir kısmının(14) düşünülüp bulunamadığı(DB) diğer kısmının(13) uğraşılıp bulunamadığı(UB) durumlar

olduğu görülmektedir. Bu durumda karşılaşılan zorlukların ve hataların yaklaşık %67'si giderilirken %33'ü giderilememiştir. Bu zorlanmalar özellikle *yanlış kullanma, karar verememe* ve *eksik kullanma* türlerinde zorluk yaşandığını göstermektedir. Bu zorlanmanın öğrencilere yansıması şu şekildedir. Örneğin;



Şekil 34. Ö5 problem 6 çözümü

Yukarıda Ö5'in 6. problemin çözümü esnasında çekilmiş olan ekran görüntüsü yer almaktadır. Ö5, değişken tanımlamada *yanlış kullanım* hatası yaptığından dolayı programı çalıştıramamıştır. Şekil 'de görüldüğü üzere gergin bir yüz ifadesiyle ekrana kilitlenmiş ve kodları incelemeye başlamıştır. Ö5 tepkisini "*Bu kısımda bir şey yapmamız gerekiyor.*" sesli düşünme ifadesiyle ortaya koymuş Değişken tanımlama satırda ne yapması gerektiğine *karar verememesi* nedeniyle ilk aşamada öylece ekrana bakmaya devam etmiş, daha sonra uzun düşünmeler sonucu hatayı gidermiştir(DÇ). Uygulamadan sonra yapılan mülakatta "*Bu problem beni biraz zorladı. "* şeklinde yaşadığı zorluğu ifade etmiştir.

Bu yapılar stratejik boyutta incelendiğinde değişken atama(71), koşul(40), döngü(46) ve fonksiyon(25) yapılarında zorlanmalar olduğu görülmektedir. Bu durumda stratejik boyutta karşılaşılan zorluklara çoğunlukla değişken atama, koşul ve döngü yapısında rastlanmıştır.

Karşılaşılan zorlanmaların bir kısmının(44) hemen çözüldüğü(Ç), bir kısmının(39) biraz düşünülüp çözüldüğü(DÇ), diğer kısmının(42) uğraşılıp çözüldüğü(UÇ) tespit

edilmiştir. Karşılaşılan hataların bir kısmının(23) durup çözülemediği(D), bir kısmının(17) düşünülüp bulunamadığı(DB) diğer kısmının(17) uğraşılıp bulunamadığı(UB) durumlar olduğu görülmektedir. Bu durumda karşılaşılan zorlukların ve hataların yaklaşık %69'u giderilirken %31'i giderilememiştir. Bu durum problem çözülürken hangi yolun kullanılması gerektiği konusunda özellikle *yanlış yerde kullanma*, *eksik kullanma*, *belirleyememe ve karar verememe* türlerinde zorlanıldığını göstermektedir.

Problem 7: *Bir string içerisinde belirlenen bir karakterin olup olmadığını kontrol eden programı fonksiyon kullanarak yazınız.*

7. problemde karşılaşılan zorluklar ve hatalar Tablo 27'de gösterilmiştir.

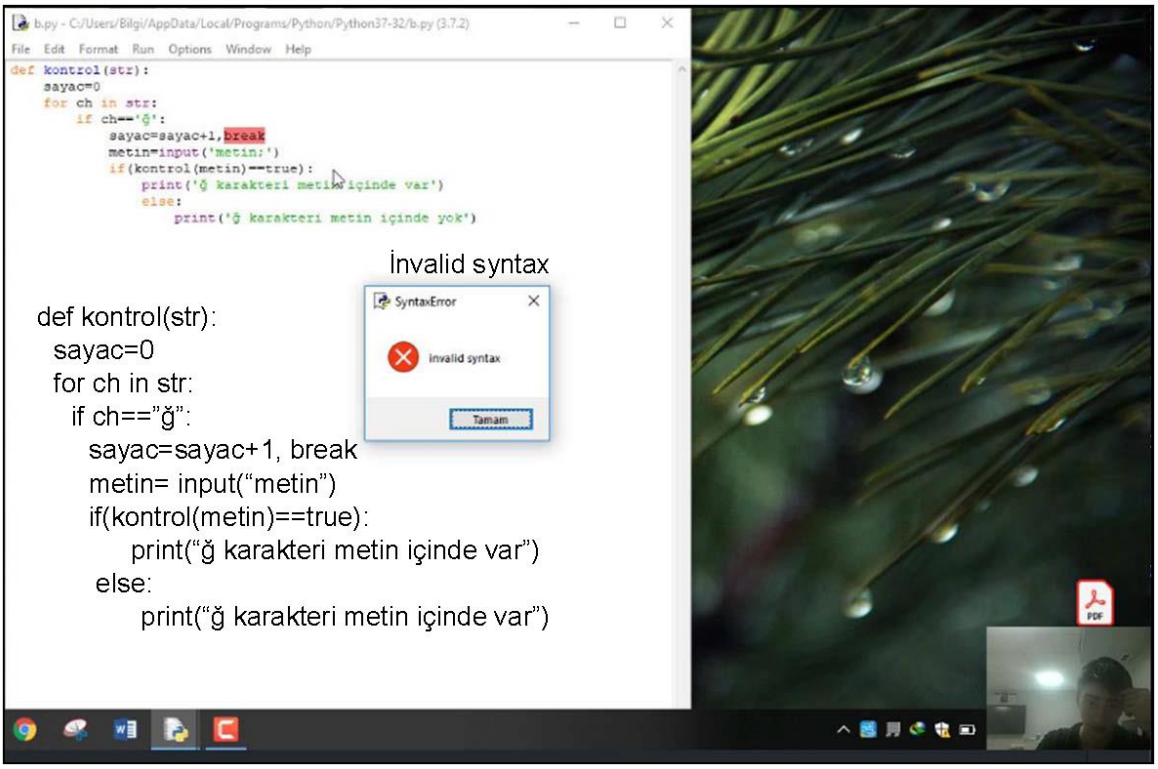
Tablo 27. Problem 7 Hata ve Zorluk Kaynakları

			Programlama Yapıları												
	Bilgi Türleri	Hata ve Zorluk Türleri	Değişken Atama			Koşul			Döngü			Fonksiyon			Toplam
Problem_7	Sözdizimsel	Girinti	4	4	3	9	9	6	6	4	2	2	1	1	51
			1	1	-	2	3	-	1	1	-	3	-	-	12
		Noktalama				-	-	5	-	2	1	3	4	1	16
							-			-		1	-	-	1
		Harf Hatası	1	1	1							-	2	-	5
				-									-		-
	Anlamsal	Yanlış Kullanma	3	2	-	3	4	6	3	2	3	-	2	-	28
			1	1	2	2	1	-	2	2	2	1	2	3	19
		Eksik Kullanma	2	1	-	4	-	6	2	4	4	-	-	1	24
			3	2	1		-		2	1	1	1	1	-	12
		Belirleyememe	-	-	3	3	5	2	1	1	2	3	4	3	27
			-	-	3	1	-	-	1	1	2	2	1	-	11
		Karar Verememe	2	-	1	4	4	5	4	5	3	2	2	2	34
			3	3	1	1	1	-	-	-	2	2	2	-	15
	Stratejik	Yanlış Kullanma	4	4	6	2	2	1	2	4	2	4	3	2	36
			6	6	5	1	2	-	2	-	1	3	2	2	30
		Eksik Kullanma	4	4	-	3	2	1	3	3	4	3	2	2	31
			2	1	3	2	3	1	1	-	2	2	1	1	19
		Belirleyememe	4	4	5	4	2	3	4	2	3	3	2	2	38
			4	3	2	1	2	3	1	1	2	-	2	1	22
		Karar Verememe	2	-	4	2	2	4	2	1	2				19
			2	3	-	1	1	2	2	1	2				14
	Toplam		128			133			114			89			464

Tablo 27 incelendiğinde sözdizimsel, anlamsal ve stratejik bilgi türlerinde zorluklarla karşılaşıldığı ve birçok hata yapıldığı görülmektedir. Bu zorlanmaların büyük bir kısmının stratejik bilgi türünde(209) olduğu belirlenmiştir. Bu bilgi türünü anlamsal bilgi türü(169) takip ederken son olarak sözdizimsel bilgi türünde(85) zorlanmalar olduğu

görülmektedir. Bu problemin çözümü için ele alınan tüm yapıların(değişken atama, operatör, koşul, döngü ve fonksiyon) kullanılması gerekmektedir. Kullanılan bu yapılara sözdizimsel boyutta bakıldığında değişken atama(16), koşul(34), döngü(17) ve fonksiyon(18) yapılarında zorlanmalar ve hatalarla karşılaşıldığı görülmektedir. Bu durumda sözdizimsel boyutta karşılaşılan zorluklara çoğunlukla koşul yapısında rastlanmıştır.

Karşılaşılan zorlanmaların bir kısmının(25) hemen çözüldüğü(Ç), bir kısmının(27) biraz düşünülüp çözüldüğü(DÇ), diğer kısmının(20) uğraşılıp çözüldüğü(UÇ) tespit edilmiştir. Karşılaşılan hataların bir kısmının(8) durup çözülemediği(D), bir kısmının(5) düşünülüp bulunamadığı(DB) görülürken çok uğraşılıp bulunamayan(UB) hiçbir durumla karşılaşılmamıştır. Bu durumda karşılaşılan zorlukların ve hataların yaklaşık %85'i giderilirken %15'i giderilememiştir. Bu zorlanmalar özellikle *girinti* ve *noktalama* türünde zorluk yaşandığını göstermektedir. Bu zorlanmanın öğrencilere yansıması şu şekildedir. Örneğin;



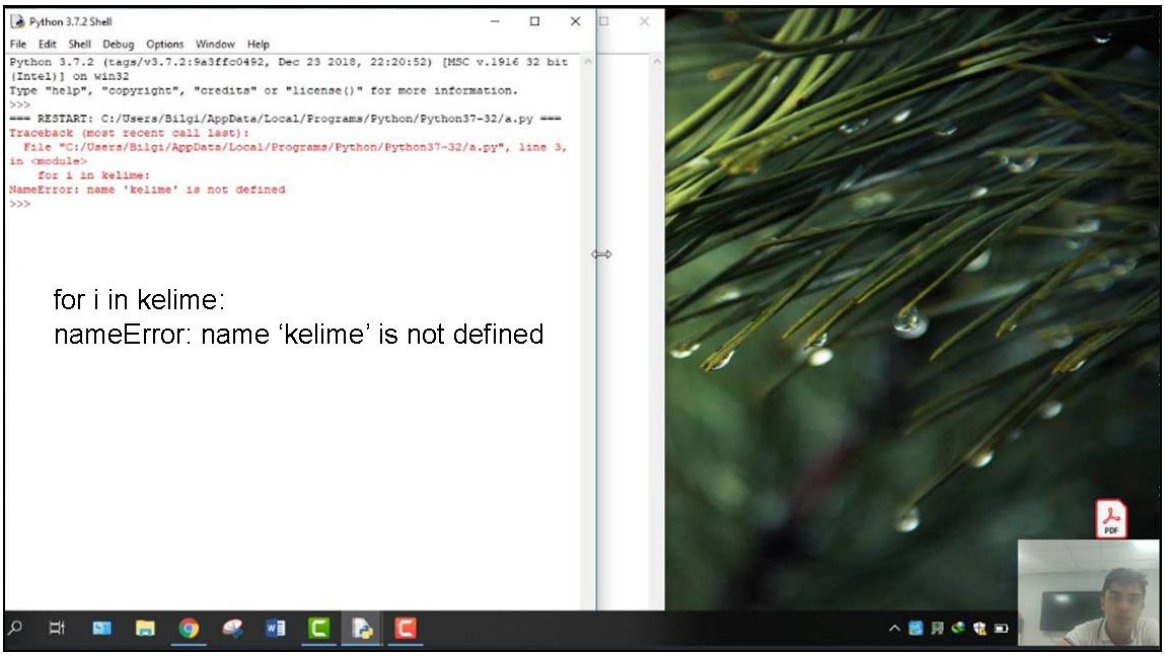
Şekil 35. Ö4 problem 7 çözümü

Yukarıda Ö4'ün 7. problemin çözümü esnasında çekilmiş olan ekran görüntüsü yer almaktadır. Ö4, *noktalama* hatası(yanlış karakter kullanımı) yaptığından dolayı "*geçersiz sözdizimi"* hatası almıştır. Hata aldıktan sonra hatanın nereden kaynaklandığını anlamaya

çalışan yüz ifadesiyle ekrana kilitlenmiş, elini alnına koymuş ve gözleriyle hatayı anlamaya çalışmıştır. Hatayı gidermek için break komutunu bir alt satıra almış ve programı tekrar çalıştırmıştır. Ancak yine aynı hatayı almıştır. Çok uğraşmış ve sonuç olarak hatayı gidermiştir(UÇ). Noktalama hatasını düzelttikten sonra programı çalıştırmıştır. Uygulamadan sonra yapılan mülakatta "*Çok hata veriyor. Verdiği hatayı düzeltiyorum yine veriyor. Hata vermesi beni zorluyor.*" şeklinde zorlandığını ifade etmiştir.

Problem çözümünde kullanılması gereken programlama yapılarına anlamsal boyutta bakıldığında değişken atama(34), koşul(52), döngü(50) ve fonksiyon(33) yapılarında zorlanmalar olduğu görülmektedir. Bu durumda anlamsal boyutta karşılaşılan zorluklar çoğunlukla koşul ve döngü yapılarında rastlanmıştır.

Karşılaşılan zorlanmaların bir kısmının(36) hemen çözüldüğü(Ç), bir kısmının(36) biraz düşünülüp çözüldüğü(DÇ), diğer kısmının(40) uğraşılıp çözüldüğü(UÇ) tespit edilmiştir. Karşılaşılan hataların bir kısmının(22) durup çözülemediği(D), bir kısmının(18) düşünülüp bulunamadığı(DB) diğer kısmının(17) uğraşılıp bulunamadığı(UB) durumlar olduğu görülmektedir. Bu durumda karşılaşılan zorlukların ve hataların yaklaşık %66'sı giderilirken %34'ü giderilememiştir. Bu zorlanmalar *yanlış kullanma, karar verememe, belirleyememe* ve *eksik kullanma* olmak üzere tüm hata türlerinde zorluk yaşandığını göstermektedir. Bu zorlanmanın öğrencilere yansıması şu şekildedir. Örneğin;



Şekil 36. Ö5 problem 7 çözümü esnasında aldığı hata

Yukarıda Ö5'in 7. Problemin çözümü esnasında çekilmiş olan ekran görüntüsü yer almaktadır. Ö5, fonksiyon içinde bir değişken tanımlamıştır. Bu değişkeni kodlar içerisinde

kullanılması gereken yerde kullanmadığından “Böyle bir değişken tanımlanmamıştır” hatası almıştır. Bu durum özellikle *eksik kullanım* hatası yapıldığını göstermektedir. Ö5 şaşkın bir yüz ifadesiyle hatayı anlamaya çalışmış ve yazmış olduğu kodları incelemeye başlamıştır. “Hata verdi… Niye böyle bir hata verdi?” şeklinde sesli düşünerek bir yandan da ekrana odaklanmıştır. Hatanın çözümü için ne yapması gerektiğini bulamayan Ö5 öylece durarak(D) hatayı çözemeyeceğini ifade etmiştir. Uygulamadan sonra yapılan mülakatta Ö5 bu problem çözümü sırasındaki davranışlarını değerlendirirken ”*Bir hata verdi. Bulamadık… Çok düşündüm ama “* demiş ve kafasını sağa sola sallayarak yaşadığı zorluğu ifade etmiştir.

Bu yapılar stratejik boyutta incelendiğinde değişken atama(78), koşul(47), döngü(47) ve fonksiyon(37) yapılarında zorlanmalar olduğu görülmektedir. Bu durumda stratejik boyutta karşılaşılan zorluklar çoğunlukla değişken atama yapısında rastlanmıştır.

Karşılaşılan zorlanmaların bir kısmının(46) hemen çözüldüğü(Ç), bir kısmının(37) biraz düşünülüp çözüldüğü(DÇ), diğer kısmının(41) uğraşılıp çözüldüğü(UÇ) tespit edilmiştir. Karşılaşılan hataların bir kısmının(30) durup çözülemediği(D), bir kısmının(28) düşünülüp bulunamadığı(DB) diğer kısmının(27) uğraşılıp bulunamadığı(UB) durumlar olduğu görülmektedir. Bu durumda karşılaşılan zorlukların ve hataların yaklaşık %59’u giderilirken %41’i giderilememiştir. Bu durum problem çözülürken hangi yolun kullanılması gerektiği konusunda *yanlış yerde kullanma*, *eksik kullanma*, *belirleyememe ve karar verememe* olmak üzere tüm hata türlerinde zorlanıldığını göstermektedir. Bu zorlanmanın öğrencilere yansıması şu şekildedir. Örneğin;

Ö2: Bu problemin cümlesini okuduğunda ilk aşamada anlamamış ve anlamak için defalarca okumuştur. Problemin çözümü için uzun uzun düşünmüş ve nasıl yapması gerektiğini *belirleyememiştir*. Bir fonksiyon tanımlamış ve tekrar problem cümlesini okumuştur. Kullanıcıdan veri almak için bir değişken tanımlamıştır. Ancak bu değişkeni bu aşamada tanımlamaması gerekmektedir. Elini yüzüne götürerek bir süre düşünmüş ve ekrana kilitlenmiştir. Daha sonra “*Bunun nasıl yapılacağını anlamadım, yapamayacağım*.” diyerek uygulamayı bitirmek istemiştir. Çok düşünmesine rağmen problemin çözümü yapamamıştır(DB). Uygulama sonrasında yapılan mülakatta Ö2 sıkılgan bir yüz ifadesiyle “*Ya problemi anladım da nasıl yapılacak onu bulamadım. Daha önce hiç böyle bir şey yapmadım. O yüzden zorlandım.*” şeklinde yaşadığı zorluğu ifade etmiştir.

Problem 8*: Bir sayının kendisi dışında bütün pozitif bölenlerinin toplamı kendisine eşit olan sayılara mükemmel sayı denir. Kullanıcının girdiği sayının mükemmel sayı olup olmadığını fonksiyon kullanarak yazınız.*

8. problemde karşılaşılan zorlanmalar ve hatalar Tablo 28’de gösterilmiştir.

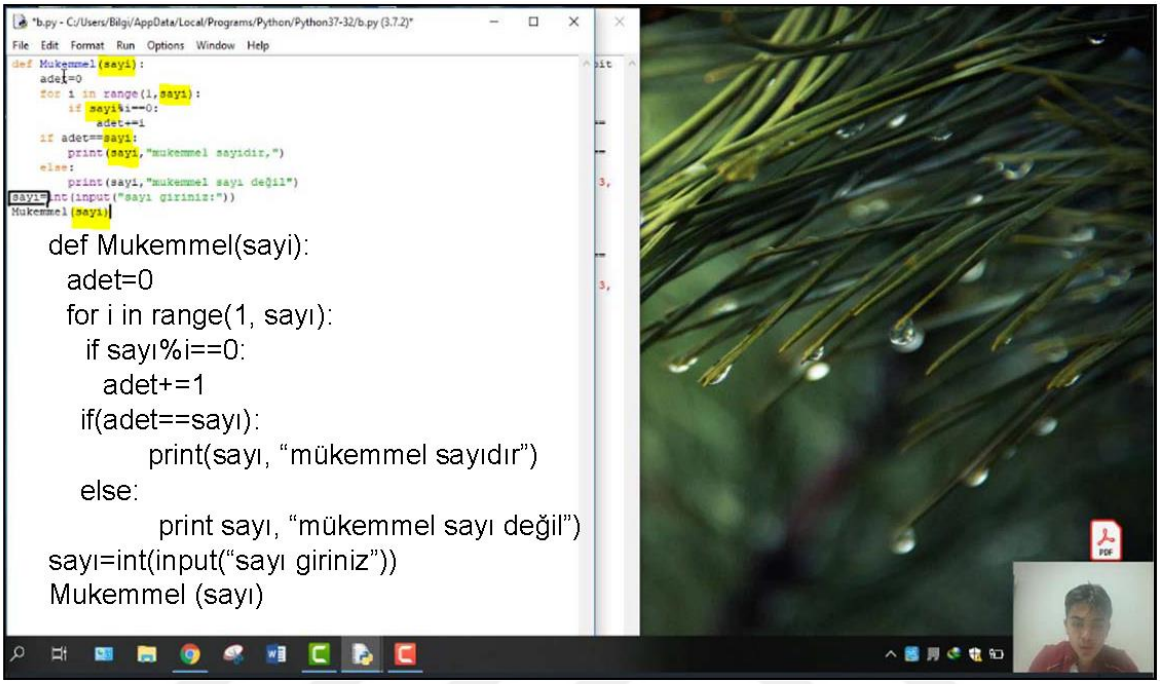
Tablo 28. Problem 8 Hata ve Zorluk Kaynakları

			Programlama Yapıları												
	Bilgi Türleri	Hata ve Zorluk Türleri	Değişken Atama			Koşul			Döngü			Fonksiyon			Toplam
Problem_8	Söz dizimsel	Girinti	4	3	2	7	8	7	4	3	1	1	2	2	44
			5	2	1	-	2	2	1	-	-		-		13
		Noktalama				2	-	-	-	1	-	3	3	1	10
							-			-			-		-
	Anlamsal	Yanlış Kullanma	2	4	2	3	5	6	2	2	3	3	1	1	34
			2	3	1	1	3	1	3	2	1	2	2	1	22
		Eksik Kullanma	2	-	3	3	2	5	3	5	3	2	1	-	29
			2	2	1		-		-	-	2	2	1	1	11
		Belirleyememe	-	2	4	4	4	1	1	1	2	4	3	2	28
			3	2	2	1	2	-	2	1	1	2	2	1	19
		Karar Verememe	2	2	3	5	2	3	2	3	3	3	2	1	31
			-	-	1	1	-	-	2	1	2	3	1	1	12
	Stratejik	Yanlış Kullanma	5	5	4	3	3	4	3	3	4	4	4	4	46
			6	5	7	-	-	3	2	2	1	4	1	3	34
		Eksik Kullanma	2	2	3	3	3	5	2	5	2	3	3	2	35
			3	-	-	2	2	2	2	3	2	3	2	2	23
		Belirleyememe	2	3	5	5	4	6	5	4	1	3	2	-	40
			3	4	4	1	1	-	2	1	3	3	3	2	27
		Karar Verememe	3	-	5	5	5	2	2	1	1				24
			4	-	-	2	3	1	1	-	2				13
	Toplam		137			145			111			102			495

Tablo 28 incelendiğinde sözdizimsel, anlamsal ve stratejik bilgi türlerinde zorluklarla karşılaşıldığı ve birçok hata yapıldığı görülmektedir. Bu zorlanmaların büyük bir kısmının stratejik bilgi türünde(242) olduğu belirlenmiştir. Bu bilgi türünü anlamsal bilgi türü(186) takip ederken son olarak sözdizimsel bilgi türünde(67) zorlanmalar olduğu görülmektedir. Bu problemin çözümü için ele alınan tüm yapıların(değişken atama, operatör, koşul, döngü ve fonksiyon) kullanılması gerekmektedir. Kullanılan bu yapılara sözdizimsel boyutta bakıldığında değişken atama(17), koşul(28), döngü(10) ve fonksiyon(12) yapılarında zorlanmalar ve hatalarla karşılaşıldığı görülmektedir. Bu durumda sözdizimsel boyutta karşılaşılan zorluklara çoğunlukla koşul yapılarında rastlanmıştır.

Karşılaşılan zorlanmaların bir kısmının(21) hemen çözüldüğü(Ç), bir kısmının(20) biraz düşünülüp çözüldüğü(DÇ), diğer kısmının(13) uğraşılıp çözüldüğü(UÇ) tespit edilmiştir. Karşılaşılan hataların bir kısmının(6) durup çözülemediği(D), bir kısmının(4) düşünülüp bulunamadığı(DB) diğer kısmının(3) uğraşılıp bulunamadığı(UB) durumlar olduğu görülmektedir. Bu durumda karşılaşılan zorlukların ve hataların yaklaşık %91'i

giderilirken %19'u giderilememiştir. Bu zorlanmalar özellikle *girinti* ve *noktalama* türünde zorluk yaşandığını göstermektedir. Bu zorlanmanın öğrencilere yansıması şu şekildedir. Örneğin;



Şekil 37. Ö5 problem 8 çözümü

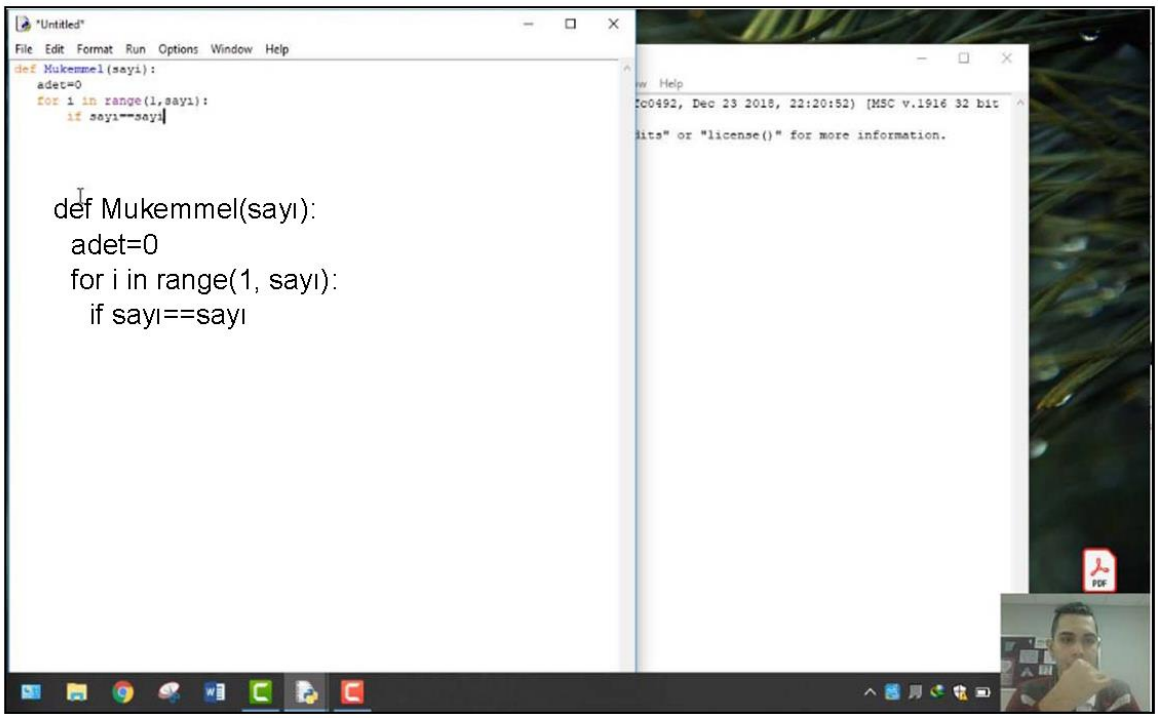
```
 in <module>
    Mukemmel(sayi)
NameError: name 'sayi' is not defined
>>>
```

Şekil 38. Ö5 problem çözümü esnasında aldığı hata mesajı

Yukarıda Ö5'in 8. Problemin çözümü esnasında çekilmiş olan ekran görüntüsü yer almaktadır. Ö5, Şekil 37'de görüldüğü gibi değişken atamada harf hatası yaptığından dolayı "*böyle bir değişken tanımlanmamıştır."* hatası almıştır. Bu esnada 'düşünceli' ve hatanın nerede olduğunu anlamaya çalışan bir yüz ifadesiyle ekrana kilitlenen öğrenci tepkisini "*Değişkende sözdizimi hatası verdi değiştirdim ama yine hata veriyor, vermemesi gerekir.*" sesli düşünme ifadesiyle ortaya koymuştur. Öğrenci programı tekrar çalıştırdıktan sonra "Doğru veriyor ama çoklu veriyor." şeklinde durumu ifade etmiş ancak çok uğraşmasına(UB) rağmen bu hata giderilememiştir. Uygulamadan sonra yapılan mülakatta "*Problem çözüm sırasında biraz zorlandım*. *Değişken satırında hata verdi, niye verdi anlamadım. Değiştirdim ama yine verdi "* şeklinde yaşadığı zorluğu ifade etmiştir.

Problem çözümünde kullanılması gereken programlama yapılarına anlamsal boyutta bakıldığında değişken atama(45), koşul(52), döngü(47) ve fonksiyon(42) yapılarında zorlanmalar olduğu görülmektedir. Bu durumda anlamsal boyutta karşılaşılan zorluklara tüm yapılarda rastlanmıştır.

Karşılaşılan zorlanmaların bir kısmının(41) hemen çözüldüğü(Ç), bir kısmının(39) biraz düşünülüp çözüldüğü(DÇ), diğer kısmının(42) uğraşılıp çözüldüğü(UÇ) tespit edilmiştir. Karşılaşılan hataların bir kısmının(26) durup çözülemediği(D), bir kısmının(22) düşünülüp bulunamadığı(DB) diğer kısmının(16) uğraşılıp bulunamadığı(UB) durumlar olduğu görülmektedir. Bu durumda karşılaşılan zorlukların ve hataların yaklaşık %66'sı giderilirken %34'ü giderilememiştir. Bu zorlanmalar *yanlış kullanma, belirleyememe, karar verememe* ve *eksik kullanma* olmak üzere tüm türlerde zorluk yaşandığını göstermektedir. Bu zorlanmanın öğrencilere yansıması şu şekildedir. Örneğin;



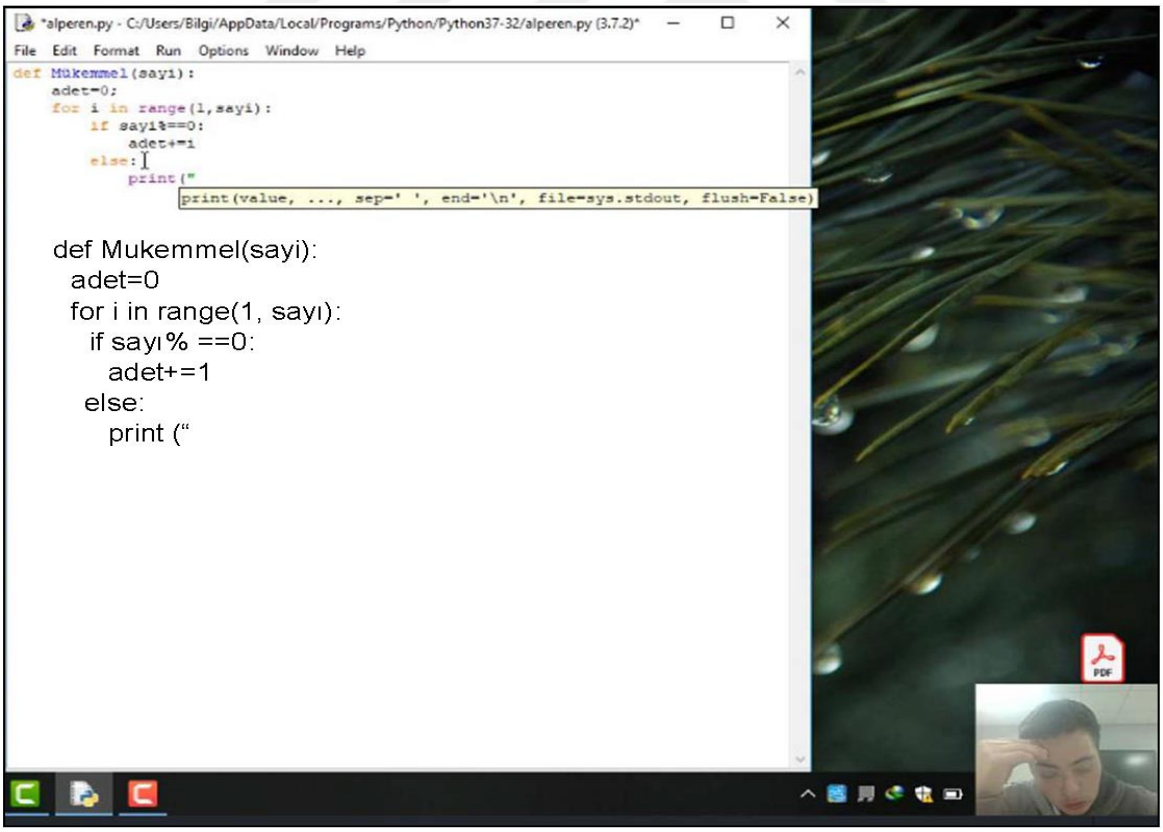
Şekil 39. Ö2 problem 8 çözümü

Yukarıda Ö2'nin 8. Problemin çözümü esnasında çekilmiş olan ekran görüntüsü yer almaktadır. Ö2, anlamsal olarak bir hata yapmıştır. Problemin doğru çözümü için yapılmaması gereken bir işlemi(koşul içerisinde bir değişkeni kendisine eşitleme şartı) yaparak doğru çözümden uzaklaşmıştır. Bu esnada stratejik olarak da ne yapacağını belirlememiş olmasından kaynaklı 'düşünceli' ve ne yapacağını bilemeyen bir yüz ifadesiyle elini ağzına götürerek bir yazmış olduğu kodlara bir problem cümlesine bakmıştır. Zorlandığı her halinden belli olan Ö2 "*Burası nasıl olacak*" diyerek kendi

kendine sesli düşünmüştür. Öğrenci problemi çözmeye çalışmış ancak doğru sonuca ulaşamamıştır. Uygulamadan sonra yapılan mülakatta ”*Bu problem beni çok zorladı ya…*“ şeklinde yaşadığı zorluğu ifade etmiştir.

Bu yapılar stratejik boyutta incelendiğinde değişken atama(75), koşul(65), döngü(54) ve fonksiyon(48) yapılarında zorlanmalar olduğu görülmektedir. Bu durumda stratejik boyutta karşılaşılan zorluklara tüm yapılarda rastlanmıştır.

Karşılaşılan zorlanmaların bir kısmının(50) hemen çözüldüğü(Ç), bir kısmının(47) biraz düşünülüp çözüldüğü(DÇ), diğer kısmının(48) uğraşılıp çözüldüğü(UÇ) tespit edilmiştir. Karşılaşılan hataların bir kısmının(38) durup çözülemediği(D), bir kısmının(27) düşünülüp bulunamadığı(DB) diğer kısmının(32) uğraşılıp bulunamadığı(UB) durumlar olduğu görülmektedir. Bu durumda karşılaşılan zorlukların ve hataların yaklaşık %59'u giderilirken %41'i giderilememiştir. Bu durum problem çözülürken hangi yolun kullanılması gerektiği konusunda *yanlış yerde kullanma*, *eksik kullanma*, *belirleyememe ve karar verememe* olmak üzere tüm türlerde zorlanıldığını göstermektedir. Bu durumun öğrencilere yansıması şu şekildedir. Örneğin;



Şekil 40. Ö3 problem 8 çözümü

Yukarıda Ö3'ün 8. Problemin çözümü esnasında çekilmiş olan ekran görüntüsü yer almaktadır. Ö3, koşul satırından sonraki print kodunda ne yazacağını bilemediğinde dolayı *belirleyememe* zorluğu yaşamıştır. Bu zorluğu gidermek için uzun uzun düşünmüş ekrana kilitlenerek kodları incelemiştir. Elini kafasını koyarak defalarca problemi okuyup anlamaya çalışarak durumu çözmeye çalışmıştır. Ö3 çok uğraşmasına(UB) rağmen bu zorluğu gidememiştir. Uygulamadan sonra yapılan mülakatta "*Ne yazacağımı bulamadım. Belli bir yere kadar geldim ama devamı gelmedi. O kısımda çok zorlandım.*" şeklinde yaşadığı zorluğu ifade etmiştir.

4. 2. 1. Sözdizimsel Bilgi Türündeki Bilişsel Yük Kaynakları

Elde edilen veriler değerlendirildiğinde Python programlama dili için sözdizimi bağlamında tüm programlama yapılarında, "girinti, iki nokta, parantez ve harf kullanımı hatası" söz konusu olduğu belirlenmişti. Bu durumlar, zorluk durumları olarak ele alınmış ve tekrarlanma durumları *Tablo 29* 'da gösterilmiştir. Karşılaşılan bu zorlukların ve hataların giderilme durumları öğrencilerin sarf etmiş oldukları çabalar ekran kayıtları, sesli düşünmeler, beden dili hareketleri ve klinik mülakat verileriyle çözdü(Ç), düşünüp çözdü(DÇ), uğraşıp çözdü(UÇ), durdu(D), düşünüp bulamadı(DB), uğraşıp bulamadı(UB) şeklinde oluşturulan kodlar çerçevesinde edilmiştir.

Tablo 29. Sözdizimsel Bilgi Türünde Karşılaşılan Zorluklar

Hata/Zorluk Türü		Problem																																				Toplam
		1						2						3						4						5						6						7						8			
Değişken Atama	Girinti	4	6	8	5	6	6	6	3	4	3	4	3	6	4	4	5	4	3	4	4	3	4	3	2															104
			-		5	7	9	4	5	4	6	4	3	5	3	2	-	2	2	1	1	-	5	2	1															71
	Parantez				4	2	-	1	1	3							1	1	-																						13
						-			-									-																						-
	Harf Hatası				1	1	3	2	1	-				2	1	-				1	1	1																			14
						-			-						-						-																			-
	Toplam	4	6	8	15	16	18	13	10	11	9	8	6	13	8	6	6	7	5	6	6	4	9	5	3															202
Koşul	Girinti				7	9	11	8	4	11	16	12	9	10	11	7	11	9	8	9	9	6	7	8	7															189
					7	4	3	4	5	1	2	3	2	2	2	3	-	-	1	2	3	-	-	2	2															48
	İki Nokta				8	9	5	5	6	4	6	4	5	6	6	4	-	4	3	-	-	5	2	-	-															82
						-		2	1	-	1	-	-		-			-			-				-															4
	Toplam				22	22	19	19	16	16	25	19	16	18	19	14	11	13	12	11	12	11	9	10	9															323
Döngü	Girinti				7	8	2	6	5	2	5	4	4	6	3	1	7	5	2	6	4	2	4	3	1															87
						-		2	1	-	2	1	-	1	-	1		-		1	1	-	1	-	-															11
	İki Nokta				8	7	3	5	4	3	4	4	-	5	-	-	3	1	1	-	2	1	-	1	-															52
						-		2	-	-	1	-	-		-			-			-				-															3
	Toplam				15	15	5	15	10	5	12	9	4	12	3	2	10	6	3	7	7	3	5	4	1															153
Fonksiyon	Girinti	5	6	8	8	6	3										4	7	4	2	1	1	1	2	2															60
		3	6	2	4	4	1										1	1	-	3	-	-		-															25
	İki Nokta	7	7	5	4	3	2										4	2	1	3	4	1	3	3	1															50
		1	1	-	-	-	1											-		1	-	-		-															4
	Harf Hatası	-	1	1	1	-	-													-	2	-																	5
			-			-															-																		-
	Toplam	16	21	16	17	13	7										9	10	5	9	7	2	4	5	3															144
Genel Top		71			184			115			108			95			97			85			67																	822

Tablo 29 incelendiğinde sözdizimsel bilgi türü ile ilgili birçok hata ve zorluk ile karşılaşıldığı görülmektedir. Karşılaşılan zorlukların kaynaklarından “girinti, noktalama, parantez, harf” olarak çıkarılan kodların eylemsel durumları oluşturulmuştur. Bu eylemler, kaynak oluşumu ve karşılaşma sıklığı durumu Tablo 30’da gösterilmiştir.

Tablo 30. Sözdizimsel Bilgi Temelli Hata ve Zorluk Kaynakları

	İçsel Bilişsel Yükü Oluşturan Kaynaklar	Bilişsel Yük Kaynağında Yük Oluşumu (Değişken atama, koşul, döngü ve fonksiyon yapısında)	Karşılaşma Sıklığı
Sözdizimsel	Girinti	• kodu yanlış yere koyma	Sıklıkla
	Noktalama	• yanlış noktalama kullanma • noktalama koymayı unutma	Sıklıkla
	Parantez	• parantez koymayı unutma • parantezi yanlış yerde açma/kapama	Düşük
	Harf	• kod kelimesini yanlış yazma • kod kelimesini eksik yazma	Düşük

Tablo 30’da bulunan içsel bilişsel yükü oluşturan kaynaklar arasında olan *girinti* yazılan kodun üst satırdaki koda ait olduğunu belirtiyor. Bu kod girintilemeden konulduğu durumda program hata veriyor ve doğru çıktı alınamıyor. Diğer bir kaynak olan *noktalama* özellikle iki nokta hatası olarak zorluk oluşturuyor. Program içerisinde fonksiyon, döngü ve koşul satırlarının sonuna konulması gereken iki nokta karakteri konulmadığında program sözdizimi hatası veriyor. *Parantez* kaynağı konulması gereken yerlere parantezin konulmaması ya da yanlış yere konulması sonucu alınan hatalar üzerine oluşan sözdizimsel bir kaynaktır. *Harf* hataları kodun yanlış ya da eksik yazılması sonucu alınan hatalar üzerine oluşan sözdizimsel bir kaynaktır. Tabloda bulunan karşılaşma sıklığı kategorisi, tüm uygulamalar sonucu yapılan hatalar ve karşılaşılan zorluk miktarları doğrultusunda değerlendirilmiştir. 1-50 değerleri arası düşük, 50-100 değerleri arası orta, 100 ve sonrası sıklıkla şeklinde derecelendirilmiştir. Bu durumdan hareketle g*irinti* ve *noktalama* hatalarının karşılaşma sıklığı fazla iken *parantez* ve *harf* hatalarının karşılaşma sıklığı düşük derecededir. Ele alınan temel öğelerin üç tanesinin sonuna konulması gereken iki nokta üst üste(:) karakterinin konulmaması noktalama hatalarının sıklıkla yapılmasına neden olmaktadır. *Noktalama* hataları genellikle girinti hatalarını da beraberinde getirdiğinden bu iki hata türüne sıklıkla rastlanmaktadır. Oysa *parantez* kullanımı kullanıcıdan alınan verilerin değişkene atanmasında kullanılır. Zorunlu

olmamakla birlikte döngü, fonksiyon ve koşul yapılarında da kullanılır. Kullanımı az olduğundan dolayı hata yapılma sıklığı düşük derecededir.

4. 2. 2. Anlamsal Bilgi Türündeki Bilişsel Yük Kaynakları

Elde edilen veriler değerlendirildiğinde Python programlama dili için anlamsal bilgi türü bağlamında tüm programlama yapılarında, “yanlış kullanma, eksik kullanma, belirleyememe, karar verememe” hataları olduğu belirlenmişti. Bu durumlar, zorluk durumları olarak ele alınmış ve tekrarlanma durumları *Tablo 31* ’de gösterilmiştir. Karşılaşılan bu zorlukların ve hataların giderilme durumları öğrencilerin sarf etmiş oldukları çabalar ekran kayıtları, sesli düşünmeler, beden dili hareketleri ve klinik mülakat verileriyle çözdü(Ç), düşünüp çözdü(DÇ), uğraşıp çözdü(UÇ), durdu(D), düşünüp bulamadı(DB), uğraşıp bulamadı(UB) şeklinde oluşturulan kodlar yardımıyla analiz edilmiştir.

Tablo 31. Anlamsal Bilgi Türünde Karşılaşılan Zorluklar

	Hata/Zorluk Türü	Problem																																		Toplam
		1				2			3			4			5			6			7			8												
Değişken Atama	Yanlış Kullanma	3	1	1	2	3	3	3	-	2	2	4	4	2	-	-	3	3	2	3	2	-	2	4	2											51
			-		-	1	2	2	2	1	2	-	-	2	1	-	1	-	1	1	1	2	2	3	1											25
	Eksik Kullanma	2	3	1	3	2	1	1	-	2	2	3	2	3	3	2	3	1	-	2	1	-	2	-	3											42
			-			-		1	-	-		-		3	2	-	3	3	1	3	2	1	2	2	1											24
	Belirleyememe				2	3	1				2	2	1	2	-	-	2	-	4	-	-	3	-	2	4											28
						-					-	1	1	1	1	2	2	-	-	-	-	3	3	2	2											18
	Karar Verememe				3	-	2	2	1	2	-	-	3	-	4	2	3	4	3	2	-	1	2	2	3											39
					4	3	2	3	3	1	-	-	1	2	2	-	2	1	-	3	3	1	-	-	1											32
	Toplam	5	4	2	14	12	11	12	6	8	8	10	12	15	13	6	19	12	11	14	9	11	13	15	17											259
Koşul	Yanlış Kullanma				5	5	4	2	5	-	3	2	-	2	3	3	3	1	1	3	4	6	3	5	6											66
					2	-	-	2	3	-	3	2	1	1	2	-	-	-	1	2	1	-	1	3	1											25
	Eksik Kullanma				1	4	2	2	2	3	3	4	2	2	1	2	3	2	1	4	-	6	3	2	5											54
						-		1	-	-	3	1	-		-		2	1	-		-			-											8
	Belirleyememe																3	-	-	3	5	2	4	4	1											22
																				1	-	-	1	2	-											4
	Karar Verememe				3	2	-	3	2	2	3	3	1	5	3	2	3	3	3	4	4	5	5	2	3											61
						-		3	2	2	3	4	1	4	4	2	2	2	1	1	1	-	1	-	-											33
	Toplam				11	11	6	13	14	7	18	16	5	14	13	9	16	9	7	18	15	19	18	18	16											273
Döngü	Yanlış Kullanma				4	3	3	4	4	2	3	4	4	3	3	4	4	4	2	3	2	3	2	2	3											66
					1	2	-	1	-	3	2	-	-	-	-	1	2	1	1	2	2	2	3	2	1											26
	Eksik Kullanma				2	3	2	4	2	3	2	4	4	3	3	4	4	6	3	2	4	4	3	5	3											70
						-		3	2	3	3	2	2	2	2	2	2	4	3	2	1	1	-	-	2											36
	Belirleyememe				2	1	1	1	1	2	1	1	1	3	2	2	-	-	1	1	1	2	1	1	2											27
						-			-			-		1	1	2	1	1	1	1	1	2	2	1	1											15
	Karar Verememe				3	3	4	4	3	5	4	4	5	3	3	2	4	3	4	4	5	3	2	3	3											74
					2	1	-	1	-	-	2	1	1	3	2	-	2	-	-	-	-	2	2	1	2											22
	Toplam				14	13	10	18	12	18	17	16	17	18	16	17	19	19	15	15	16	19	15	15	17											336

Tablo 31'in devamı

Hata/Zorluk Türü		Problem																		
		1			2			3	4	5	6			7			8			Toplam
Fonksiyon	Yanlış Kullanma	3	3	2	3	1	-				4	1	2	-	2	-	3	1	1	26
			-			-					1	1	4	1	2	3	2	2	1	17
	Eksik Kullanma	-	2	3	2	2	-				2	1	-	-	-	1	2	1	-	16
			-			-						-		1	1	-	2	1	1	6
	Belirleyememe										2	1	1	3	4	3	4	3	2	23
											2	-	-	2	1	-	2	2	1	10
	Karar Verememe													2	2	2	3	2	1	12
														2	2	-	3	1	1	9
	Toplam	3	5	5	5	3	0				11	4	7	11	14	9	21	13	8	119
Genel Toplam		24			110			108	119	121	149			170			186			987

4. 2. 3. Stratejik Bilgi Türündeki Bilişsel Yük Kaynakları

Elde edilen veriler değerlendirildiğinde Python programlama dili için stratejik bilgi türü bağlamında tüm programlama yapılarında, “yanlış yerde kullanma, eksik kullanma, belirleyememe, karar verememe” hataları olduğu belirlenmiştir. Bu durumlar, zorluk durumları olarak ele alınmış ve tekrarlanma durumları Tablo 32’de gösterilmiştir. Karşılaşılan bu zorlukların ve hataların giderilme durumları öğrencilerin sarf etmiş oldukları çabalar ekran kayıtları, sesli düşünmeler, beden dili hareketleri ve klinik mülakat verileriyle çözdü(Ç), düşünüp çözdü(DÇ), uğraşıp çözdü(UÇ), durdu(D), düşünüp bulamadı(DB), uğraşıp bulamadı(UB) şeklinde oluşturulan kodlar yardımıyla analiz edilmiştir.

Tablo 32. Stratejik Bilgi Türünde Karşılaşılan Zorluklar

Hata/Zorluk Türü		Problem																															
		1			2			3			4			5			6			7			8			Top							
Değişken Atama	yanlış yerde kullanma	4	4	6	4	3	6	5	5	4	5	6	7	4	6	4	6	5	5	4	4	6	5	5	4	117							
		5	5	4	5	5	5	6	4	5	7	5	3	5	6	6	7	5	6	6	6	5	6	5	7	129							
	eksik kullanma	2	1	1				3	4	4	2	5	5	4	3	3	2	3	5	4	4	-	2	2	3	62							
			-						-		1	2	-		-		1	2	1	2	1	3	3	-	-	16							
	Belirleyememe				5	5	1	4	3	4	2	3	4	5	6	3	2	3	4	4	4	5	2	3	5	77							
					3	2	2	3	-	4	-	2	1	-	3	3	2	2	-	4	3	2	3	4	4	47							
	karar verememe				2	-	2	-	-	3	3	2	4	1	2	2	3	4	-	2	-	4	3	-	5	42							
					4	5	-	5	6	1	-	4	3	2	2	2	3	-	-	2	3	-	4	-	-	46							
	Toplam	11	10	11	23	20	16	26	22	25	20	29	27	21	28	23	26	24	21	28	25	25	28	19	28	536							
Koşul	yanlış kullanma				2	3	3	2	2	2	5	5	2	3	2	-	4	4	4	2	2	1	3	3	4	58							
						-			-			-		2	1	1	-	-	1	1	2	-	-	-	3	11							
	eksik kullanma										4	3	4	3	3	2	-	-	2	3	2	1	3	3	5	38							
											2	2	1	1	1	-	1	2	-	2	3	1	2	2	2	22							
	Belirleyememe				2	2	3	3	3	3	2	4	3	3	1	3	3	2	4	4	2	3	5	4	6	65							
						-			-		2	1	1	2	2	3	2	-	1	1	2	3	1	1	-	22							
	karar verememe										3	2	2	3	4	2	3	2	1	2	2	4	5	5	2	42							
												-		2	1	-	2	1	1	1	1	2	2	3	1	17							
	Toplam				4	5	6	5	5	5	18	17	13	19	15	11	15	11	14	16	16	15	21	21	23	275							
Döngü	yanlış kullanma				4	3	3	2	4	4	5	2	3	4	2	1	3	2	5	2	4	2	3	3	4	65							
					2	2	1	3	2	2	3	2	2	2	2	1	1	1	-	2	-	1	2	2	1	34							
	eksik kullanma				3	2	3	2	3	4	4	4	2	4	4	2	3	2	3	3	3	4	2	5	2	64							
					3	2	1	3	2	2	2	1	1	-	1	1	1	2	3	1	-	2	2	3	2	35							
	Belirleyememe				3	2	1	1	1	2	3	3	2	3	4	4	5	4	1	4	2	3	5	4	1	58							
					1	1	-	2	1	1	2	2	1	-	2	1	2	1	1	1	1	2	2	1	3	28							
	karar verememe							1	1	2	3	2	1	1	-	-	2	1	1	2	1	2	2	1	1	24							
									-		1	-	-	-	2	2	1	1	-	2	1	2	1	-	2	15							
	Toplam				16	12	9	14	14	17	23	16	12	14	17	12	18	14	14	17	12	18	19	19	16	323							

Tablo 32'nin devamı

Hata/Zorluk Türü		Problem																
		1	2			3	4	5	6			7			8			Top
Fonksiyon	yanlış kullanma								2	3	3	4	3	2	4	4	4	29
										-		3	2	2	4	1	3	15
	eksik kullanma								4	2	2	3	2	2	3	3	2	23
										-		2	1	1	3	2	2	11
	Belirleyememe		2	1	1				2	2	2	3	2	2	3	2	-	22
			1	3	2				-	-	3	-	2	1	3	3	2	20
	Toplam		3	4	3				8	7	10	15	12	10	20	15	13	120
Genel Toplam		32	121			133	175	160	182			209			242			1254

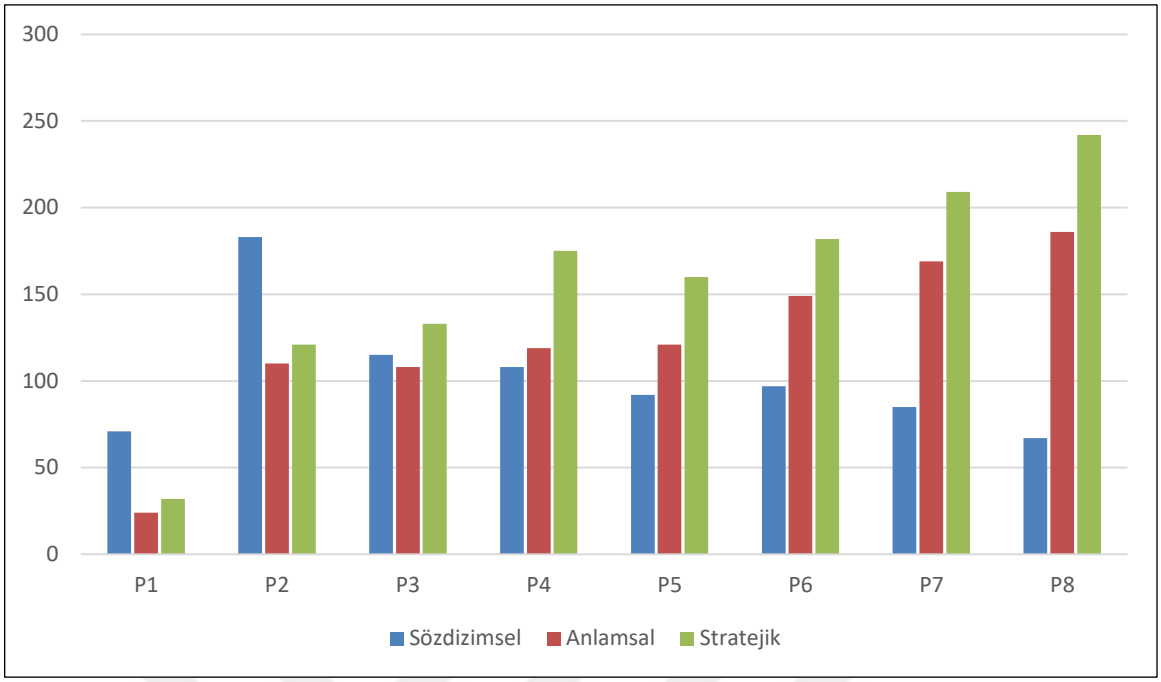
Tablo 31 ve Tablo 32 incelendiğinde anlamsal ve stratejik bilgi türü ile ilgili karşılaşılan zorlukların kaynaklarından “yanlış kullanma, eksik kullanma, belirleyememe ve karar verememe” olarak çıkarılan kodların eylemsel durumları oluşturulmuştur. Bu eylemler, kaynak oluşumu ve karşılaşma sıklığı durumu Tablo 33’te gösterilmiştir.

Tablo 33. Anlamsal ve Stratejik Bilgi Temelli Hata ve Zorluk Kaynakları

	İçsel Bilişsel Yükü Oluşturan Kaynaklar	Bilişsel Yük Kaynağında Yük Oluşumu (Değişken atama, koşul, döngü ve fonksiyon yapısında)	Karşılaşma Sıklığı
Anlamsal/Stratejik	Yanlış Kullanma	• kod satırını yanlış yazma • kod satırını yanlış yerde kullanma	Sıklıkla
	Eksik Kullanma	• kod satırını eksik yazma	Sıklıkla
	Belirleyememe	• ne yazacağını bilememe	Sıklıkla
	Karar Verememe	• ne yazacağına karar verememe	Sıklıkla

Tablo 33’te bulunan içsel bilişsel yükü oluşturan kaynaklar arasında olan kod satırında meydana gelen yük durumları *yanlış kullanma* yazılan kodun yanlış yazıldığını ya da yanlış yere yazıldığını, *eksik kullanma* durumu kodun eksik yazıldığını belirtmektedir. Diğer bir durum olan *belirleyememe* öğrencinin ne yazacağı ile ilgili hiçbir fikri olmamasını, ne yazacağını bilememesini içeriyor. *Karar verememe* ise öğrencinin ne yazacağına karar verememesini, aklına gelen tüm fikirleri denemesini ancak hangisinin doğru olduğunu bulamaması durumunu içermektedir. Tabloda bulunan karşılaşma sıklığı kategorisi, tüm uygulamalar sonucu yapılan hatalar ve karşılaşılan zorluk miktarları doğrultusunda değerlendirilmiştir. 1-50 değerleri arası düşük, 50-100 değerleri arası orta, 100 ve sonrası sıklıkla şeklinde derecelendirilmiştir. Bu durumdan hareketle anlamsal ve stratejik bilgi türü içerisinde sınıflandırılan hata ve zorluklarla (yanlış kullanma, eksik kullanma, belirleyememe, karar verememe) sıklıkla karşılaşılmıştır.

Sözdizimsel, anlamsal ve stratejik bilgi türlerine bakıldığında birçok hata ve zorluk ile karşılaşıldığı görülmektedir. Bu zorluklar problemler boyutunda incelendiğinde hata ve zorluk sayıları Şekil 41’ de gösterilmektedir.



Şekil 41. Problemlerin zorluk ve hata durumları

Şekil 41 incelendiğinde sözdizimsel, anlamsal ve stratejik bilgi türlerinin problemler üzerinde karşılaşılma durumları görülmektedir. Sözdizimsel bilgi türünde karşılaşılan zorlukların ilk olarak bir artış gösterdiği daha sonra azaldığı görülmektedir. Uygulamanın sözdizimsel hataları uyarıcı mesaj olarak vermesi bu durumu sağlayan etken olarak gösterilebilir. Öğrenciler o hatayla daha önce karşılaştığından bir daha uyarıcı mesaj almamak için daha dikkatli davranmışlardır. Problemlerin zorluğu arttıkça anlamsal bilgi türünde karşılaşılan zorlukların arttığı görülmektedir. En fazla artan bilgi türünün stratejik bilgi türü olduğu görülmektedir. Bu durum daha fazla öğe ve öğe etkileşimi içeren problemlerde programlama yapılarının nasıl işe koşulması ve nasıl bir yol izlenmesi gerektiği konusunda karar verilememesi olarak değerlendirilebilir.

4. 3. Problemlerin Zorluk Durumu ile Algılanan Bilişsel Yük Arasındaki İlişki

Bu çalışma kapsamında 4 hafta boyunca uygulanan problemlerin zorluk dereceleri ele alınan yapılar içerisinde temel öğeler ve öğe etkileşimleri bağlamında alan uzmanları tarafından belirlenmiştir. Zorluk durumları belirlenen bu problemlerin çözümünde öğrencilerin algılamış olduğu bilişsel yük durumları ekran kayıtları, gözlem formu, sesli düşünme, beden dili hareketleri ve klinik mülakat verileriyle analiz edilmiş ve bu veriler programlama bilgi türü boyutunda sunulmuştur. Bu iki durum arasındaki ilişki incelendiğinde aşağıdaki sonuç ortaya çıkmıştır. Problemlerin zorluk durumu içerdiği

öğeler ve bu öğelerin birbirleriyle etkileşim durumları değerlendirilerek belirlenmiştir. Bu anlamda problemler zorluk durumları göze alınarak basitten karmaşığa sıralanmıştır.

1. problem değişken atama, operatör ve fonksiyon yapılarının kullanımını gerektirmektedir. Bu bakımdan içerdiği temel öğe ve öğe etkileşimi sayısı en az olan problemdir. Kullanılan öğelerin kullanılma ve birbirleriyle etkileşim durumları yüksektir. Bu problemin zorluk derecesi yüksek olmakla birlikte en fazla sözdizimsel bilgi türünde zorluk yaşanırken en az anlamsal bilgi türünde zorlukla karşılaşılmıştır.

2. problem değişken atama, operatör, koşul, döngü ve fonksiyon olmak üzere ele alınan tüm yapıların kullanımını gerektirmektedir. Kullanılan öğelerin kullanılma ve birbirleriyle etkileşim durumları yüksektir. Bu problemin zorluk derecesi yüksek olmakla birlikte en fazla sözdizimsel bilgi türünde zorluk yaşanırken en az anlamsal bilgi türünde zorlukla karşılaşılmıştır.

3. problem değişken atama, operatör, koşul ve döngü yapılarının kullanımını gerektirmektedir. Kullanılan öğelerin kullanılma ve birbirleriyle etkileşim durumları yüksektir. Bu problemin zorluk derecesi yüksek olmakla birlikte en fazla stratejik bilgi türünde zorluk yaşanırken en az anlamsal bilgi türünde zorlukla karşılaşılmıştır.

4. problem değişken atama, operatör, koşul ve döngü yapılarının kullanımını gerektirmektedir. Kullanılan öğelerin kullanılma ve birbirleriyle etkileşim durumları yüksektir. Bu problemin zorluk derecesi yüksek olmakla birlikte en fazla stratejik bilgi türünde zorluk yaşanırken en az sözdizimsel bilgi türünde zorlukla karşılaşılmıştır.

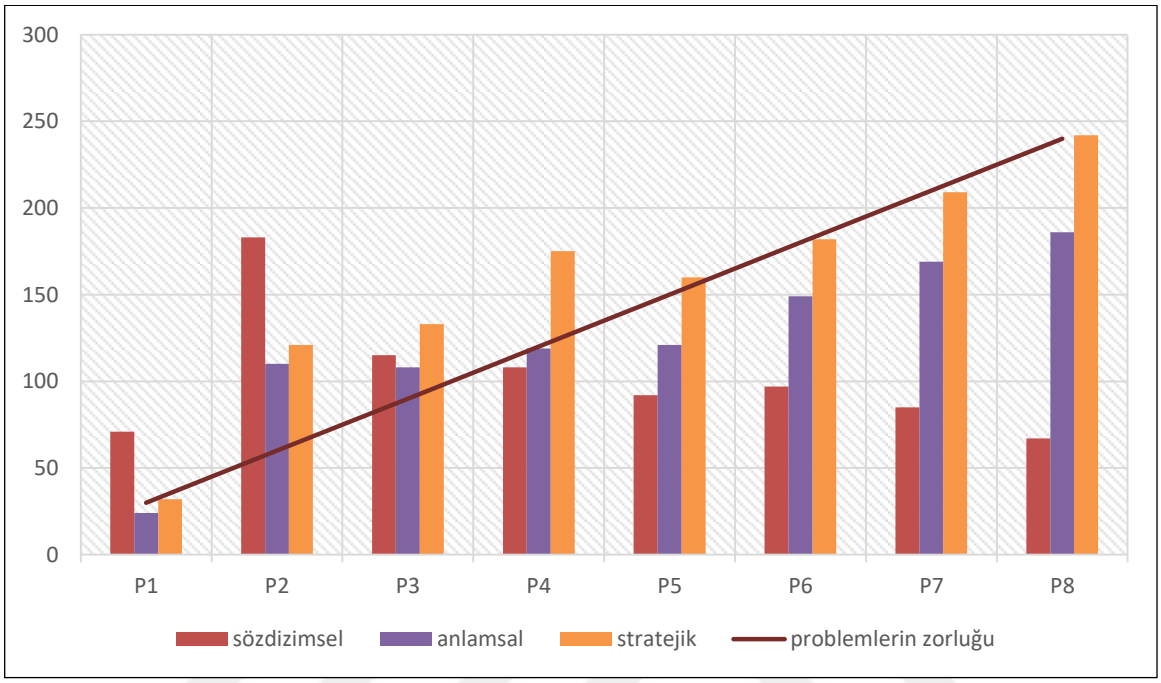
5. problem değişken atama, operatör, koşul ve döngü yapılarının kullanımını gerektirmektedir. Kullanılan öğelerin kullanılma ve birbirleriyle etkileşim durumları yüksektir. Bu problemin zorluk derecesi yüksek olmakla birlikte en fazla stratejik bilgi türünde zorluk yaşanırken en az sözdizimsel bilgi türünde zorlukla karşılaşılmıştır.

6. problem değişken atama, operatör, koşul, döngü ve fonksiyon olmak üzere ele alınan tüm yapıların kullanımını gerektirmektedir. Kullanılan öğelerin kullanılma ve birbirleriyle etkileşim durumları yüksektir. Bu problemin zorluk derecesi yüksek olmakla birlikte en fazla stratejik bilgi türünde zorluk yaşanırken en az sözdizimsel bilgi türünde zorlukla karşılaşılmıştır.

7. problem değişken atama, operatör, koşul, döngü ve fonksiyon olmak üzere ele alınan tüm yapıların kullanımını gerektirmektedir. Kullanılan öğelerin kullanılma ve birbirleriyle etkileşim durumları yüksektir. Bu problemin zorluk derecesi yüksek olmakla birlikte en fazla stratejik bilgi türünde zorluk yaşanırken en az sözdizimsel bilgi türünde zorlukla karşılaşılmıştır.

8. problem değişken atama, operatör, koşul, döngü ve fonksiyon olmak üzere ele alınan tüm yapıların kullanımını gerektirmektedir. Kullanılan öğelerin kullanılma ve

birbirleriyle etkileşim durumları en yüksek olan problemdir. Bu problemin zorluk derecesi yüksek olmakla birlikte en fazla stratejik bilgi türünde zorluk yaşanırken en az sözdizimsel bilgi türünde zorlukla karşılaşılmıştır.



Şekil 42. Uzmanlar tarafından belirlenen problemlerin zorluk durumu ile öğrenciler tarafından deneyimlenen zorluk ve hata kaynakları

Uygulama sürecinde çözülmesi istenen problemlerin zorluk durumları içerdikleri temel öğeler ve bu öğelerin etkileşim durumları değerlendirilerek belirlenmiştir. Şekil 42'de bu zorluk durumu çizgi ile gösterilmiştir. Problem 1'den 8'e kadar zorluk durumunun arttığı görülmektedir. Problemlerde karşılaşılan zorluklar ve hatalar programlama bilgi türleri boyutunda analiz edilmiştir. Programlama bilgi türleri sütunlarla gösterilmiş ve problemlerin zorluk durumu arttıkça karşılaşılan zorluklar ve hatalar sunulmuştur.

Problemlerin zorluk durumu arttıkça sözdizimsel bilgi türünde yapılan hatalar ve karşılaşılan zorluklar azalırken, anlamsal ve stratejik bilgi türünde yapılanlar artış göstermektedir. Bu durumdan hareketle problemlerin zorluk durumu belirlenirken ele alınan öğe etkileşimleri ile öğrencilerin yaptığı hata ve karşılaştıkları zorluklar programlama bilgi türleri boyutunda problemler üzerinde Tablo 34'te gösterilmiştir.

Tablo 34. Problemlerin Hata ve Zorluk Durumları ve Öğe Etkileşimleri

		Bilgi Türü			Öğe etkileşimi									
		S	A	Str	D.A X O	D.A X K	D.A X D	D.A X F	O X K	O X D	O X F	K X D	K X F	D X F
Problem	1	71	24	32	√			√ √			√			
	2	184	110	121	√√√√√	√ √	√ √ √	√	√	√ √ √				
	3	115	108	133			√ √		√ √	√ √ √		√ √ √		
	4	108	119	175	√	√ √	√ √ √							
	5	95	121	160	√ √ √ √	√ √ √	√		√ √	√ √ √		√ √ √		
	6	97	149	182	√	√	√	√	√	√		√		
	7	85	170	209	√	√			√ √	√		√	√	√
	8	67	186	242	√	√ √ √	√	√	√ √	√		√	√	√

Tablo 34 içerisinde görülen kategoriler; S:sözdizimsel bilgi türü, A: anlamsal bilgi türü, Str: stratejik bilgi türü, D.A X O: değişken atama-operatör etkileşimi, D.A X K: değişken atama-koşul etkileşimi, D.A X D: değişken atama-döngü etkileşimi, D.A X F: değişken atama-fonksiyon etkileşimi, O X K: operatör-koşul etkileşimi, O X D: operatör-döngü etkileşimi, O X F: operatör-fonksiyon etkileşimi, K X D: koşul-döngü etkileşimi, K X F: koşul-fonksiyon etkileşimi, D X F: döngü-fonksiyon etkileşimi şeklindedir.

5. TARTIŞMA

DuBoulay (1986), programlama sürecinde yaşanan zorlukları; programlama problemlerinin tam olarak ne istediğini anlama ve programlama dilinin sözdizimsel ve anlamsal olarak kullanılacak bilgileri açıklama, süreçte kullanılması gereken programlama yapılarını bilme gibi birtakım becerilere sahip olma olarak belirtmiştir. Programlama öğretimi için kullanılan programlama dillerinden bağımsız olarak, program yazma ve program anlama sırasında çeşitli düzeylerde bilişsel güçlükler olduğu görülmektedir(Renumol vd., 2009).

Bu çerçevede programlamada yaşanan bilişsel süreçlerdeki içsel bilişsel yük oluşturan kaynakları araştıran çalışmanın bu bölümünde; gözlemci verileri, ekran kaydı, sesli düşünme ve beden dili hareketlerinden elde edilen bulgulara yönelik yapılan tartışmalara yer verilmiştir. Programlama sürecindeki bilişsel yüklerin belirlenmesi için programlama yapıları ve ilgili bilgi türlerinin öğe olarak ele alınması şeklinde ortaya konulmuştur. Bu çerçevede hangi yapılarda hangi bilgi türlerinde zorlanıldığı, hangi ilişkilerin kurulamadığı ve bunların nedenlerinin ne olduğu gibi tartışmalara yer verilmiştir.

5. 1. Problemlerin Zorluklarının Belirlenmesi

Problemin zorluk durumu belirlenirken alan uzmanları tarafından muhtemel çözüm üzerinde değerlendirmeler yapılmıştır. Bu amaçla önerilen zorluk belirleme yolu, temel öğeler ve öğe etkileşimi olmak üzere iki kategoriden oluşmaktadır. Temel öğeler kategorisi için ilk olarak problemlerin doğru çözümünde hangi yapıların(değişken atama, koşul, operatör, döngü, fonksiyon) kullanılması gerektiği belirlenmiştir. Problem bu yapılar kullanılmadan da çözülebilir mi? sorusu üzerine, yapıların problem için kullanılma zorunluluğu değerlendirilerek derecelendirilmiştir. Bunun yanı sıra ikincil zorluk belirleme söz konusu yapının problemin çözümü için stratejik öneme sahip olmasıdır. Bu yapıyla problem daha kolay çözülebiliyorsa yapının problem için stratejik önemi olduğu değerlendirilmiştir. Bu yapıların problem içerisinde kaç kez kullanıldığı durumu dikkat edilen diğer bir etken olarak ele alınmıştır. Bu şekildeki frekanslama problemin içsel zorluk yapısı hakkında değerlendirme ve problemleri birbiriyle kıyaslama yapabilecek durumlar oluşturabildiği düşünülmektedir. Problemlerin zorluk durumları düşük/orta/yüksek olarak değerlendirilmiştir. Örneğin; yapının kullanılması problemin çözümü için zorunlu değilse ve stratejik önemi az ise düşük, zorunlu değil ancak stratejik önemi varsa orta, hem zorunlu hem de stratejik öneme sahipse yüksek değer almıştır. Problemlerin çözümü için temel

öğelerin birbirleriyle etkileşme durumlarının problemlerin zorluğu belirlenirken göz önüne alınmış olması da öğrencilerin programlama sürecinde yapacakları ilişkilendirmelerin karmaşıklığı üzerinden içsel zorlukların belirlenmesinin değerlendirilmesine katkı sağladığı söylenebilir. Bu öğeler birbirleriyle etkileşmeden de problem çözülebilir mi? sorusu üzerinden etkileşim durumları değerlendirilmiştir. Etkileşimin problem için zorunlu olmasının yanı sıra stratejik öneme sahip olması etkileşime verilecek değer için önemlidir. Bu değerlendirme öğe etkileşimi kategorisinin derecesini belirlemiştir. Dolayısıyla zorluk belirleme için önerilen yolda temel öğelerin birbirleriyle etkileşme zorunluluğu ve stratejik önemi bu problemin zorluğunu belirleyen unsurlar olarak ele alınabileceği ve bu yolun temel problemlerde işe koşulabileceği söylenebilir.

Bu çalışmada üzerinde çalışılan problemler, özel olarak sınırlı sayıda programlama yapısı ile çözülebilir problemler olarak belirlenmiştir. Bu durum, önerilen iki boyutlu yapının sınırlı düzeyde detaylandırılarak, doğruluğunun test edilmesini kolaylaştırmıştır. Bu şekilde kolaylıklar sözdizimsel, anlamsal ve stratejik bilgiler belirlenmiş, bunların hangi etkileşimler ilişkiler oluşturabildiği ortaya konulabilmiştir. Bu yöntemle problemler muhtemel çözüm süreleri de dikkate alınarak küçükten büyüğe ve basitten karmaşığa sıralanmış ve öğrencilere uygulama boyunca bu sıra ile verilmiştir.

5. 2. Öğrencilerin Deneyimledikleri İçsel Bilişsel Yük Kaynakları

5. 2. 1. Sözdizimsel Bilgi Türüne İlişkin Kaynaklar

İçsel bilişsel yükü oluşturan sözdizimsel kaynaklar arasında olan *girinti* yazılan kodun bir üst satırdaki koda ait olduğunu belirtmektedir. Program içerisinde fonksiyon, döngü ve koşul satırlarının sonuna konulması gereken iki nokta karakteri konulmadığında program *noktalama* hatası vermektedir. *Parantez* hataları konulması gereken yerlere parantezin konulmaması ya da yanlış yere konulması sonucu oluşmaktadır. *Harf* hataları kodun yanlış ya da eksik yazılması sonucu alınan hatalar üzerine oluşan sözdizimsel bir kaynaktır.

Bu çalışmada, programlama yapıları içerisinde ele alınan değişken atama, operatör, koşul, döngü ve fonksiyon yapıları uygulama esnasında çözülmesi istenen problemler içerisinde kullanılması gereken temel öğeler olarak belirlenmiştir. Elde edilen bulgular sözdizimsel bilgi türü boyutunda incelendiğinde bu bilgi türünün tüm temel öğelerde(değişken atama, operatör, koşul, döngü, fonksiyon) bilişsel yük oluşturduğu görülmektedir. Sözdizimsel olarak öğrencilerin en fazla zorluk yaşadığı ve hata ile karşılaştığı durumlar *girinti* ve *noktalama* hataları olarak belirlenmiştir.

Uygulama yapılan programın yapılan sözdizimsel hataları editör mesajı olarak vermesinden dolayı noktalama hatalarının nerede yapıldığını bilme ve giderme noktasında öğrencilere kolaylık sağladığı düşünülmektedir. *Noktalama* ve *girinti* hatalarıyla ilgili karşılaşılan zorluklara en fazla koşul yapısında rastlanırken bu yapıyı değişken atama, döngü ve fonksiyon yapıları takip etmiştir. Özellikle koşul ve döngü satırlarının sonunda konulması gereken iki nokta üst üste(:) karakterinin konulmaması üzerine karşılaşılan hatalar ve zorlukların ilk problemlerde sıklıkla olmakla birlikte giderek azalmıştır. Ancak azalmış olsa dahi çalışmanın sonuna doğru cevaplanan problemlerde her öğrenci tarafından süregiden bir şekilde karşılaşılması *noktalama* ve *girinti* hatalarının içsel bilişsel yük oluşturan kaynaklar arasında önemli bir yeri olduğuna işaret etmektedir.

Özellikle sözdizimsel hataların bazı durumlarda birbirilerini etkiler yapıda oluşması bu çalışmada ortaya çıkan önemli hata ve zorluk kaynakları arasında değerlendirilebilir. Uygulama yapılan programda fonksiyon, koşul ve döngü satırlarının sonuna konulması gereken iki nokta üst üste(:) karakterinin konulması bir alt satırda olması gereken girintileme işlemini de otomatik olarak yapması, girinti ve noktalama hatalarının birbiriyle bağlantılı olarak oluşmasının sebeplerinden birisi olarak değerlendirilebilir. Bu karakterin(:) konulmaması bir hata durumu oluştururken bir alt satırda olması gereken girintileme işlemini de unutturmaktadır. Dolayısıyla bir hatayla daha karşılaşılmasına neden olmaktadır. İki nokta üst üste(:) karakterinin üç yapının ardından da kullanılma durumu birçok hatayı beraberinde getirmektedir. *Noktalama* hatalarının *girinti* hatalarını beraberinde getirmesinde problemler içerisindeki döngü-koşul, döngü-değişken atama, koşul-değişken atama öğeleri arasındaki etkileşimler olduğu düşünülmektedir. Nitekim problemlerin doğru çözümü için olması gereken etkileşimlerin doğru kurulamaması öğrenciler için zorluklar ortaya çıkmakta, zaman zaman hata yapmalarına sebep olabilmektedir. Öğrencilerin problemlerin çözümünde döngü yapısı içerisinde koşul yapıları kullanılması gerektiğinde öğrencilerin noktalama hataları artmaktadır. Örneğin, koşul kullanılarak sağlanan komutun, döngü içerisinde kullanılması gerekiyorsa koşul, döngü satırının altında ve girintilenmiş şekilde yazılmalıdır. Dolayısıyla bu noktada döngü-koşul etkileşimi gerçekleşmelidir. Girintileme işleminin olmaması demek bu iki yapı arasında etkileşimin kurulamamış olması ve girinti işleminin sözdizimi hatası olarak öğrencinin karşısına çıkması demektir. Problemlerin doğru çözümü için bu yapıların birbirleriyle etkileşmesi zorunludur.

Sözdizimsel bilgi türünde en çok *girinti* ve *noktalama* hatalarının meydana gelmesi öğeler arasındaki etkileşimleri gerçekleştirme adına yapılması gereken birçok işlemin örneğin döngü-koşul etkileşiminde döngü komutunun sonuna iki nokta karakterinin konularak bir alt satıra yazılacak olan koşul komutunun döngü satırının girintisi şeklinde

yazılmasının gerekmesi olduğu değerlendirilebilir. İki nokta(:) karakterinin konulmaması girinti işlemini de otomatik oluşturmayacağından her iki satırda da sözdizimsel anlamda hata oluşmaktadır. Dolayısıyla bu öğeler içerisindeki koşul ve döngü yapılarının sonuna iki nokta konulmaması ve alt satıra inildiğinde girintileme işleminin olmaması bu öğeler arasındaki etkileşime engel olmaktadır.

Bu çalışma içerisinde *parantez* fonksiyon, değişken atama, koşul ve döngü yapılarında kullanılan bir karakterdir. Ancak *parantez* hatalarına ve karşılaşılan zorluklar incelendiğinde özellikle değişken atama yapısında rastlandığı belirlenmiştir. Kullanıcıdan istenen verinin atandığı değişkenlerde iki adet parantez kullanımı gerekmektedir. Öğrencilerin parantez kapama işlemini unutmaları hata yapmalarına neden olmaktadır. Dolayısıyla bu hatalar herhangi bir öğe etkileşimi nedeniyle ortaya çıkmadığı söylenebilir. Python programlama ortamının hatayı mesaj olarak vermesi, hatanın fark edilip giderilmesini sağlamaktadır. Bu anlamda parantez hatalarının kolaylıkla aşılabilen hata ve zorluk türlerinden olduğu düşünülebilir. Sözdizimsel zorluk ve hataların karşılaşıldığı öğe ve öğe etkileşimleri daha çok programlama dilinin o şekildeki tanımlamalarından kaynaklıdır. Örneğin; parantez hatalarının daha çok temel öğeler içerisinde değişken atama yapısında rastlanılması, sıklıkla bu yapıda kullanılmasının gerektiğinden olduğu düşünülmektedir.

Sözdizimsel olarak yapılan hatalar ve karşılaşılan zorluklardan bir diğeri ise harf hatalarıdır. Program içerisinde kullanılması gereken kodun yazımının yanlış yazılması, harflerinin eksik yazılması nedeniyle öğrenciler bu hatalarla karşılaşmışlardır. Harf hatası tüm yapılarda karşılaşılabilen bir hata ve zorluk türüdür. Bu hatalar, tüm temel öğelerde karşılaşılabildiği gibi tüm etkileşim durumlarından da kaynaklanabilmektedir. Harf hatalarının daha çok dilin komut yapısının yabancı dilde oluşundan, öğrencilerin basit değişken isimleri seçmeyişlerinden kaynaklandığı değerlendirilebilir. Programın hata yapılan satırı vurgulayarak hata mesajı vermesi hatanın fark edilmesi ve giderilmesi noktasında öğrencilere kolaylık sağlamıştır.

Sözdizimsel bilgi türünde karşılaşılan zorluklar ve hataların büyük ölçüde öğrenciler tarafından program yazımı sırasında giderilebildiği görülmüştür. Öğrencilerin sözdizimsel bilgi türünde karşılaştıkları hataları giderebilme durumları diğer bilgi türlerine göre daha kolay olmaktadır. Bu durumun nedenlerinden birisi olarak kullanılan editörün sözdizimsel hataları hata mesajı olarak bir pencerede kullanıcının karşısına çıkarması ve bu hatalar çözülmeden programı çalıştırmaması olduğu değerlendirilmektedir. Diğer taraftan katılımcıların ekran kayıtları, sözdizimsel hataların uyarıcı mesaj olarak verilmesinin hataların giderilmesinde kolaylaştırıcı rol oynadığını göstermiştir. Bu şekilde verilen hata mesajlarının(etkili bilişsel yük) yük kaynağı oluşturduğu düşünülebilir. Nitekim bu hata

mesajları öğrencilerin yaptığı hataları görmelerini sağlamış ve hataları giderme noktasında onlara yol göstermiştir. Bu çerçevede; Pollock vd.(2002) acemilerin, yeni karmaşık becerileri öğrenirken yüksek öğe etkileşimi nedeniyle zorlandıklarını savunmuştur. Uygulamanın verdiği hata mesajları sonraki problemler için katılımcıyı uzmanlaştırarak öğrenmeyi kolaylaştırmıştır.

Öğrencilerin birçok temel öğeyi program içerisinde kullanırken noktalama hataları yapıyor olması zaman zaman program yazma süreçlerini olumsuz etkilemektedir. Bu şekildeki hatalar öğrencilerin bazen anlamsal veya stratejik hata yapıp yapmadıkları noktasında kontroller yapmalarına, geri dönüp yazdıkları kodları kontrol etmelerine, zaman kaybetmelerine ve programlamaya karşı isteksizlik oluşmasına sebep olabilmektedir.

Sözdizimsel hatalar tümüyle değerlendirildiğinde; yanlış yere koyma(girintileme işlemi), unutma, harflerinin yerlerini karıştırma ve eksik harf kullanma gibi durumlardan kaynaklandığı değerlendirilmektedir. Karşılaşılan bu durumlar problemlerin zorluk derecesi arttıkça azalmıştır. Bu anlamda çözülen problem sayısı arttıkça öğrencilerin alışkanlıklarını şekillendirmek zorunda bırakan bir özellik taşıdığına işaret etmektedir. Dolayısıyla sözdizimsel hatalar, öğrencilerin yoğun zihinsel süreç gerektiren durumları değil, dilin kullanımını alışkanlık edinip edinememeleri üzerinden ortaya çıkan durumlar olarak değerlendirilebilir.

5. 2. 2. Anlamsal Bilgi Türüne İlişkin Kaynaklar

Anlamsal bilgi türü çerçevesinde karşılaşılan içsel bilişsel yük kaynakları daha çok programlama yapılarını doğru kullanamama temellidir. Bu kaynaklar arasında *yanlış kullanma* yazılan kodun yanlış yazıldığını ya da yanlış yere yazıldığını, *eksik kullanma* durumu kodun eksik yazıldığını belirtmektedir. Diğer bir durum olan *belirleyememe* öğrencinin ne yazacağı ile ilgili hiçbir fikri olmamasını, ne yazacağını bilememesini içermektedir. Bu kaynaklar arasında *karar verememe* öğrencinin ne yazacağına karar verememesini, aklına gelen tüm fikirleri denemesini ancak hangisinin doğru olduğunu bulamaması durumunu içermektedir.

Elde edilen bulgular anlamsal bilgi türü boyutunda incelendiğinde bu bilgi türünün tüm temel öğelerde (değişken atama, operatör, koşul, döngü, fonksiyon) bilişsel yük oluşturduğu görülmektedir. Anlamsal olarak öğrencilerin en fazla zorluk yaşadığı ve hata ile karşılaştığı durumlar *yanlış kullanım* ve *eksik kullanım* hataları olarak belirlenmiştir. Bu hataların nedeni öğrencilerin kullanılması gereken yapıları ve ne şekilde kullanılması gerektiğini bilememelerinden kaynaklandığı değerlendirilmiştir. Bu anlamda *yanlış kullanım* ve *eksik kullanım* hatalarının sıklıkla yapılması bir yük kaynağı olarak

belirlenmiştir. Bu zorluklara en fazla döngü yapısında rastlanırken bu yapıyı takip eden diğer yapılar koşul, değişken atama ve fonksiyon yapılarıdır. Bu yapılar belirli kural gerektiren yapılardır. Bunun yanı sıra bu yapıların içerisinde hangi komutların kullanılması gerektiği bilinmesi gereken önemli bilgilerdir. Döngü, koşul ve fonksiyon yapıları kullanılırken içerisinde hangi değişkenin kullanılması gerektiği bilinmelidir. Öğrenciler öğeler arasındaki etkileşimi kuramadıklarından hangi yapının hangi yapıyla işe koşulması gerektiğini belirleyemeyebiliyorlar. Dolayısıyla bu noktada uzun düşünmelere dalıp zorlanıyorlar. *Yanlış kullanım* hataları özellikle söz konusu yapılar içerisinde kullanılması gereken değişkenden, farklı bir değişken kullanıldığı zamanlarda ortaya çıkmaktadır. Döngü, koşul ve fonksiyon yapısının altında yazılması gereken komutların da yanlış yazılması bu hata türünü ortaya çıkaran etmenlerdendir. Bu anlamda öğrencilerin yapıların doğru kullanımını ve yapılarla birlikte kullanılması gereken komutları bilmeleri gereklidir. Öğrencilerden bazıları bazı programlama temel öğelerini kullanırken, gereken değişkenleri yanlış kullanarak, öğeler etkileşimlerden kaynaklı hatalar ile karşılaşmışlardır. Örneğin bir değişkenin başka bir değişkene eşitliğini koşul satırında sorgulaması gereken (if sayi==toplam:) durumunda öğrenciler kendisine eşitliğini(if sayi==sayi:) sorgulayarak *yanlış kullanım* hatası yapmıştır. Bu hatada anlamsal bilgide temel öğeler arası etkileşim kaynaklı hata olarak değerlendirilmiştir. Bu durum koşul yapılarında ve döngü yapılarında ve bunların kombinasyonel etkileşimlerinde sıklıkla karşılan bir durum olarak karşımıza çıkmaktadır. Bu öğelerin diğer öğelerle etkileşim durumu yüksek olduğundan ve genel olarak problemlerin çözümü için stratejik öneme sahip olduklarından bu hatalara sıklıkla rastlanmaktadır. *Eksik kullanım* hatalarına bakıldığında daha çok yapılarla birlikte kullanılması gereken komutların yazılmaması ya da eksik yazılmasından doğan zorluklar olduğu belirlenmiştir. Problemin çözümü için bir değişkene atama yapılacaksa öncelikle bu değişkenin tanımlanması gerekmektedir. Öğrenciler döngü, koşul gibi yapıların ardından değişken atamıştır. Ancak programın başında bu değişkeni tanımlamadığından dolayı program çalıştırıldığında “böyle bir değişken tanımlanmadı” hatası ile karşılaşılmıştır. Bu değişken tanımlanmadığından dolayı *eksik kullanım* hatası yapılmıştır.

Belirleyememe ve *karar verememe* durumları öğrencilerin sıklıkla karşılaştıkları zorluklardandır. Öğrenciler bu durumlarla karşılaştıklarında uzun süre düşünerek problemleri çözmeye çalışmışlardır. Bu zorlukların nedeni hangi yapının kullanılması gerektiğine *karar verilememesi* ve *belirlenememesinden* kaynaklıdır. Örneğin; öğrenciler, değişken atama-koşul ya da değişken atama-döngü etkileşiminde koşul ve döngü yapıları içerisinde tanımlanan hangi değişkeni kullanması gerektiğine karar verememektedir. Diğer yandan çözümün yapılıp sıra print ile değişken yazdırmaya geldiğinde hangi değişkenin yazdırılması gerektiğini belirleyememektedirler.

Öğrenciler, problemlerin zorluk durumu arttıkça ilk aşamada uzun bir düşünme süreci yaşamışlar ve çözüm için bazı adımlar atmışlardır. Örneğin; kullanıcının girdiği iki sayı arasındaki asal sayıların yazdırılmasını isteyen problemde, çözüm yapılırken hangi değişkenin yazdırılması gerektiğine karar verememişlerdir. Diğer bir durum ise o aşamada hangi değişkenin kullanılması gerektiği ya da hangi yapının olması gerektiği belirlenememiştir. Örneğin; öğrenciler fonksiyon kullanılarak yapılması gereken problemlerde ilk olarak fonksiyonu tanımlamışlardır. Ancak fonksiyondan sonra hangi yapıları ne şekilde işleme koymaları gerektiğini belirleyememişlerdir. Benzer biçimde Özmen ve Altun(2014), öğrencilerin programlamada zorluk çekmesine neden olan bir diğer konunun, programlama dilleri ile ilgili fonksiyonları hatırlama ve bu fonksiyonlara ilişkin parametreler olarak belirlemiştir. Bu durum öğrencilerin bazılarının komutların nasıl kullanıldığını ve öğelerin birbirleriyle nasıl etkileşmesi gerektiğini bilme noktasında oldukça zorlandıklarını göstermektedir. Özellikle döngü-koşul, döngü-değişken atama, koşul-değişken atama öğeleri arasındaki etkileşimler bu zorlanmaya neden olmuştur. Atanan değişkenlerin döngü ve koşul içerisinde nasıl kullanılması gerektiği, döngü ve koşul yapılarının birlikte nasıl işe koşulacağı öğrencileri anlamsal boyutta oldukça zorlamıştır. Bu durum Leahy vd. (2015) tarafından da belirtildiği üzere yüksek öğe etkileşimli bir görevin öğelerinin eşzamanlı olarak özümlenmesi gerekir ve bu nedenle çok yüksek çalışan bir bellek yükü yükler. Bu tür görevler, düşük öğeli bir etkileşim görevinden çok farklı nedenlerle zordur.

Bu çalışma her ne kadar yükseköğretim öğrencileriyle yapılan bir çalışma olsa da anlamsal bilgi noktasındaki zorlukların programlama süreci için önemli bir yük kaynağı olduğuna dair işaretler farklı çalışmalarla da ortaya konulmaktadır. Bu çalışmalardan birisinde Özmen ve Altun (2014), lisans öğrencilerinin programlama derslerinde başarısızlığın nedenleri ve programlamada karşılaştıkları problemler hakkındaki görüşlerini incelemiştir. Araştırmacıların öğrencilerin süreçte yaşadıkları zorlukların temel olarak programlama bilgisi, programlama becerileri, programın anlamını anlama ve hata ayıklama ile ilgili olduğuna ilişkin bulguları bu araştırma ile benzerlik göstermektedir.

Anlamsal bilgi türünde karşılaşılan hatalar ve zorluklar tümüyle değerlendirildiğinde yapının doğru kullanımını bilememe, hangi aşamalarda kullanması gerektiğine karar verememe ve çözüm için hangi yapıların kullanılması gerektiğini belirleyememe gibi durumlardan kaynaklandığı değerlendirilmektedir. Karşılaşılan bu durumlar problemlerin zorluk derecesi arttıkça artmıştır. Problemlerin zorluk derecesi arttıkça problem içerisindeki temel öğe ve öğe etkileşimi sayısı arttığından hangi yapıların ne şekilde kullanılması gerektiği kararının verilememesi öğrencilerde yoğun zihinsel süreçler geçirmelerine neden olmuştur. Bu çerçevede anlamsal bilgi türündeki hatalar ve zorluklar,

yoğun zihinsel süreç gerektiren durumlar üzerinden ortaya çıkmakta olduğu değerlendirilebilir.

5. 2. 3. Stratejik Bilgi Türüne İlişkin Kaynaklar

Programlama sürecinde stratejik bilgi türündeki içsel bilişsel yük kaynakları arasında olan *yanlış kullanma* problemin çözümü için yazılan kodun yanlış yerde kullanıldığına, *eksik kullanma* durumu eksik kod yazıldığına işaret etmektedir. Bu çerçevede karşılaşılan *karar verememe* durumu ise öğrencinin problemin çözümü için hangi yolun kullanılması gerektiği noktasında öğrencinin ne yazacağına karar verememesini, aklına gelen tüm fikirleri denemesini ancak hangisinin doğru olduğunu bulamaması durumunu içerirken diğer bir durum olan *belirleyememe* öğrencinin ne yazacağı ile ilgili hiçbir fikri olmamasını, ne yazacağını bilememesini içermektedir. Bu kaynaklar arasında olan *yanlış yerde kullanım* ve *eksik kullanım* hata yapılması sonucu ortaya çıkarken *karar verememe* ve *belirleyememe* bir zorluk durumu olarak ele alınmaktadır.

Programlama sürecindeki hata ve zorluklar stratejik bilgi türü boyutunda incelendiğinde bu bilgi türünün değişken atama, koşul, döngü ve fonksiyon olmak üzere ele alınan tüm yapılarda bilişsel yük oluşturduğu görülmektedir. Oluşturulan kodlar içerisinde *yanlış yerde kullanım* hatalarının sıklıkla yapılması bir yük kaynağı olarak belirlenmiştir. Öğrenciler gerekli programlama bilgilerini birleştirip programı nasıl tasarlayacakları konusunda zorluk çekmişlerdir. Özellikle problemlerin zorluk dereceleri arttıkça stratejik anlamda yaşanan zorluklar artış göstermiştir. Örneğin; 2.problemin çözümü için kullanıcıdan 5 adet isim, vize notu ve final notu verilerinin değişkenlere atanması gerekmektedir. Öğrenciler bu işlem için döngü kullanmıştır. Daha sonra alınan vize ve final notlarının problem cümlesi içerisinde verilen yüzdeliklerinin alınıp döngü dışında işleme koyulması gerekmektedir. Bu işlem için tanımlanan değişken de döngü içerisinde kullanılarak *yanlış yerde kullanım* hatası yapılmıştır. Program çalıştırıldığında çıktı vermediği görülmüştür. Ancak hatanın nereden kaynaklandığı anlaşılamamıştır. Benzer biçimde Kinnunen ve Malmi (2008), programlama dersinde en fazla zorlanılan konuları tespit etmek amacıyla yaptıkları çalışmada öğrencilerin en fazla hata bulma, problemin çözümü için algoritma oluşturma ve kod yazma konularında zorluk yaşadıklarını belirtmişlerdir.

Yanlış (yerde) kullanma stratejik bilgi olarak problem çözümünde daha çok öğe etkileşimlerinin söz konusu olduğu noktalarda öğrencilerin yapmış oldukları hata olarak ortaya çıkmaktadır. Örneğin; en başta tanımlanması gereken değişkenlerin kodlar arasında tanımlanması, değişkenlerin yanlış yerde kullanıldığını göstermektedir. Bu durum yapıların hangi aşamalarda işe koşulması gerektiğini bilememekten kaynaklandığı

düşünülmektedir. Öğrenciler problemlerin çözümünde kullanmaları gereken yapının, kullanması gereken yapıyı kullanmamalarından kaynaklı *eksik kullanım* hatası yapmaktadırlar. *Karar verme* ve *belirleyememe* zorlukları çoğunlukla hangi yapıların kullanılması gerektiğinin bilinmemesi ve yapılar arasındaki etkileşimlerin kurulamamasından kaynaklanmaktadır. Bu durum tüm etkileşimler için geçerlidir.

Stratejik bilgi türü hangi komutların, nasıl işe koşulması gerektiği noktasında doğru karar verme ve en uygun yolu bulma gerektirdiğinden hataları düzeltme ve çözüme ulaşma konusunda öğrencilerin en fazla zorluk yaşadığı ve daha çok uğraştıkları bilgi türü olduğunu göstermektedir. Benzer biçimde Tan, Ting ve Yang (2011), öğrencilerin zorluk yaşadığı konulardan birinin programda hata bulmak olduğunu vurgulamıştır. Benzer biçimde Pollock vd. (2002), acemilerin, yeni karmaşık becerileri öğrenmeyi yüksek öğe etkileşimi nedeniyle zor bulduklarını savunmuştur.

Öğrencilerin zamansal olarak da en çok stratejik hataları anlama ve çözme noktasında sıkıntı yaşadıkları görülmüştür. Stratejik olarak karşılaştıkları zorlukları gidermek için harcadıkları süre diğer bilgi türlerine göre çok daha fazladır. Bu çerçevede hatayı fark edememe ya da doğru sonuca ulaşmak adına izlenmesi gereken yolu bulamama durumunun öğrencilerde bilişsel yük oluşturduğu değerlendirilmektedir. Lahtinen, Ala-Mutka ve Jarvinen (2005), yapmış oldukları çalışmada programlamada karşılaşılan zorlukların programlama yapısının anlaşılması ve bir programın nasıl tasarlanacağının anlaşılması gibi konulardan kaynaklı olduğunu belirtmişlerdir. Bu türde yapılan hatalar ve karşılaşılan zorlukların arkasında çocukların problem çözme ve algoritma oluşturma noktasında yaşadığı sorunlar yatmaktadır. Bu konuda Eryılmaz (2003), programlamaya yeni başlayanların (acemi programcıların) ön koşul olarak problem çözme becerisine sahip olmaları gerektiğini belirtmiştir.

Stratejik bilgi türündeki hatalar ve zorluklar tümüyle değerlendirildiğinde öğrencilerin problemin çözümü için nasıl bir yol izlemeleri gerektiğini belirleyememe, öğe etkileşimlerini kuramama gibi durumlardan kaynaklandığı değerlendirilmektedir. Karşılaşılan bu durumlar problemlerin zorluk derecesi arttıkça artmıştır. Problemlerin zorluk derecesi arttıkça problem içerisindeki temel öğe ve öğe etkileşimi sayısı arttığından yapılar arasındaki etkileşimlerin nasıl kurulacağına karar verilememesinin öğrencilerde bilişsel yük oluşturduğu değerlendirilmektedir. Dolayısıyla stratejik hatalar ve zorluklar, yoğun zihinsel süreç gerektiren durumlar üzerinden ortaya çıkmaktadır.

5. 3. Programlama Problemlerinin Zorluk Durumu ile Öğrencilerin Deneyimlediği İçsel Bilişsel Yük İlişkisi

Sweller, Chen ve Kalyuga (2014) bilişsel süreçlerle ilişkili olarak verilen görevde öğelerin tanımlanabileceğini ve öğe etkileşiminin, bilişsel yük seviyelerinin değerlendirilmesinde etkili olduğunu ifade etmişlerdir. Literatür incelendiğinde öğe etkileşim düzeylerinin, öğrenme materyallerinde etkileşen öğelerin sayısının tahmin edilerek belirlenebildiğine dair birçok araştırma mevcuttur (Sweller 1994; Sweller ve Chandler 1994; Tindall Ford vd., 1997). Bu çalışmada problemlerin muhtemel çözümleri üzerinden içerdiği temel öğeler(programlama yapıları) ve öğe etkileşimleri(yapıların birbirleriyle kullanılma durumu) göz önüne alınarak zorluk durumları belirlenmiştir. Bir problemin çözümü içerisinde kaç adet temel öğe bulunduğu, temel öğelerin kullanılma zorunluluğu, kaç adet öğe etkileşimi olduğu, bu öğelerin birbirleriyle etkileşme durumları analiz edilmiştir. Bu sayılar, kullanılma zorunlulukları ve öğenin problem çözümü için stratejik öneme sahip olma durumu alan uzmanları ile birlikte belirlenerek problemler zorluk durumlarına göre derecelendirilmiştir. Benzer şekilde Sweller, Chen ve Kalyuga (2017) yapmış oldukları araştırmada birçok öğe ve öğe etkileşimi içeren bir görevin zor olduğunu belirtmişlerdir.

Problemlerin zorluk durumu arttıkça öğrenciler uzmanlaşmış ve karşılaştıkları hata ve zorlukları giderme noktasında daha başarılı olmuşlardır. Uygulama sürecinin ilk haftalarında sözdizimsel bilgi türü boyutunda çok fazla hata ile karşılaşılmıştır. Editör mesajları yardımıyla öğrenciler hatanın nereden kaynaklandığını anlama ve hatayı giderme noktasında çaba sarf etmişlerdir. Son problemlerin çözümü diğer problemlere nazaran daha fazla temel öğe ve öğe etkileşimi içermektedir. Problem içerisinde temel öğeler ve öğe etkileşim durumları fazla olduğunda daha fazla kod yazımı söz konusu olacağından sözdizimsel hatalara daha fazla rastlanacağı düşünülse de, haftalar ilerledikçe problemler zorlaşsa dahi sözdizimsel hatalar azalmıştır. Uygulamanın gerçekleştirildiği programlama dili (Python) sözdizimsel olarak yapılan hataları editör mesajı olarak vermektedir. Bu anlamda öğrenciler nerede hata yaptıklarını görebilmişlerdir. Bu durum hatayı anlama ve giderme noktasında öğrencilere kolaylık sağlamıştır. Bu çerçevede öğrencilerin Python dilinde sıklıkla yaptıkları hatalardan olan noktalama ve girinti hatalarını fark edip bunları yapmamayı öğrenmelerinden kaynaklandığı düşünülmektedir. Bu duruma editör mesajlarının da olumlu yönde katkısı olduğu değerlendirilebilir. Bu mesajların İngilizce olması, öğrencilere başlangıçta ekstra bir zorluk yüklediği düşünülebilir. Ancak ilerleyen problemlerde öğrenciler mesajların anlamlarını da içselleştirerek gerekli düzenlemeleri yapmaya alışmışlardır. Mesajları aldıkları sırada hemen kapatıp vurguyla gösterilen satıra odaklanan öğrenciler hatayı fark

edip giderme işlemine yönelmişlerdir. Sürekli editör mesajıyla karşılaşılan öğrenciler sonraki haftalarda yapılan uygulamada hata mesajı almamak için daha dikkatli davranmış ve eksik ya da yanlış işlem yapmamaya gayret göstermiştir. Benzer şekilde Pollock vd. (2002), yapmış oldukları çalışmada öğrencilere verilen görevlerde uzmanlık derecelerinin artmasının öğrencilerin anlayışında da bir artışa yol açabileceğini belirmiştir.

Öğrenciler ilk problemlerde anlamsal ve stratejik bilgi türlerinde daha az hata ve zorlukla karşılaşırken, problemlerin zorluk derecesi arttıkça daha fazla hata ve zorlukla karşılaşmışlardır. Bu durum temel öğe sayısının artmasının yanında problemin çözümü için bu öğeler arasında önemli etkileşimler kurulması gerektiğinden kaynaklanmaktadır. Öğrenciler kullanması gereken yapıların doğru kullanımını bilemediklerinden ve bu yapıları problemin doğru çözümü için nasıl işe koşmaları gerektiğini belirleyememelerinden kaynaklandığı değerlendirilmektedir. Diğer yandan hata yapıldığında hatanın nereden kaynaklandığı fark edilemediğinden hatayı giderme yoluna gidilememiştir. Bundan dolayı öğrencilerin 5. problemden itibaren öğrenciler problemleri çözmek için çok zaman harcadıkları düşünülebilir. Benzer biçimde Leahy vd. (2015), yapmış oldukları çalışmada yüksek öğe etkileşimli bir görevde, öğrencilerin, sonraki problemleri çözmede yardımcı olacak uzun süreli bellekteki bilgiyi elde etmek için materyali birçok kez incelemeleri gerekebileceğini belirtmişlerdir.

Bu çalışmada kullanılan zorluk belirleme yöntemi ile değerlendirildiğinde sorulan problemlerin haftalar ilerledikçe zorlaştığı söylenebilir. Bu çerçevede öğrencilerin özellikle stratejik bilgi gerektiren durumlardaki hatalarının haftalar geçtikçe artış gösterdiği görülmektedir. Bu hataların özellikle 5. problemden sonra sorulan problemlerde öğrencilerin yapmaları gereken ilişkilendirmelerin artışı ve karmaşıklaşmasından kaynaklandığı düşünülmektedir. Diğer yandan sorulan problemlerin doğasında yer alan matematiksel kavramların artış göstermesi bunun nedenlerden birisi olabilir. Nitekim bazı problemlerde asal sayı, mükemmel sayı gibi tanımlamalar için öğrencilerin öncelikle bu tanımlamaları hatırlamaları gereklidir. Daha sonra bu tanımlamalar çerçevesinde kullanılması gereken programlama yapılarını belirleyip uygun biçimde kullanmaları beklenmektedir. Hatırlama, uygun yapıları bulma ve bu yapıları ilişkilendirme gibi davranışların kısa sürede doğru biçimde yapılması öğrencilerin yoğun bir zihinsel süreç yaşamalarını gerektirmekte ve bu durum, problemlerde stratejik hataları yoğun biçimde yapmalarının sebeplerinden birisi olduğu düşünülmektedir.

Bu çalışmada problemlerin zorluk durumlarının belirlenmesi temel öğeler ve öğe etkileşimi üzerinden gerçekleştirilmiştir. Bu iki boyutlu yapı ile problemlerin zorluk durumu açıklanabilmiş ve bu zorluk durumu öğrencilerin bu problemleri çözerken algıladıkları içsel bilişsel yük hakkında anlamlı ve doğru bilgiler sunabilmiştir. Nitekim problemlerin

belirlenen yönteme göre zorluk dereceleri arttıkça yapılan hatalar ve karşılaşılan zorluk durumları programlama bilgi türleri boyutunda sonuçlandırılmıştır.

5. 4. Çalışmayı Benzerlerinden Farklılaştıran Bazı Noktalar

Bu çalışma 5 öğrenci ile Python programı üzerinden yürütülmüştür. Bu anlamda çalışmanın az sayıda öğrenci ile 8 problem üzerinden yürütülmesi, python programının özgün yapısı ve veri toplama sürecinin ekran kayıtları, sesli düşünmeler, beden dili hareketleri ve araştırmacının tutmuş olduğu gözlem raporlarıyla gerçekleştirilmesi ile elde edilen verilerin genellenebilirliği yönüyle sınırlı kaldığı söylenebilir. Örneklem sayısının ve problem sayısının arttırılması ile daha fazla veri elde edileceğinden çıkan sonuçların daha net ifade edilebileceği düşünülmektedir. Bunun yanı sıra python programının sözdizimsel olarak yapılan hataları editör mesajı yardımıyla göstermesi öğrencilerin bu bilgi türünde yaptığı hataların azalmasını sağlamıştır. Uygulama sürecinde başka bir programın kullanılmasıyla ulaşılacak sonuçların daha farklı çıkabileceği değerlendirilmektedir. Uygulama sürecinde araştırmacı dışında başka bir gözlemcinin sürece katılıp gözlem formunu doldurmasıyla gözden kaçan durumların ortaya çıkarabileceği ve elde edilen sonuçların güvenilirliğinin desteklenebileceği düşünülmektedir.

Literatür incelendiğinde birçok problem tabanlı programlama sürecinde yaşanan hataları belirleme ve bu hataları sınıflandırma çalışmaları olduğu görülmektedir. Bu çalışma programlama sürecinde içsel bilişsel yük oluşturan kaynakları programlama bilgi türleri boyutunda araştırmıştır. Uygulama esnasında kullanılan problemlerin önerilen iki boyutlu yöntemle zorluklarının belirlenmesi ve öğrencilerin algıladığı içsel bilişsel yük ile ilişkilendirilmesi araştırmayı benzerlerinden ayırmaktadır. Aynı zamanda yalnızca öğrenci çözümleri üzerinden değil, öğrencilerin beden dili hareketleri, sesli düşünmeleri ve uygulama sonrasında onlarla yapılan klinik mülakatlar sonucu elde edilen verilerle değerlendirme yapılmıştır. Tüm veriler öğrencilerin hataları ve zorlukları giderip giderememe ve bu süreçte sarf ettikleri çaba göz önüne alınarak analiz edilmiştir. Bu durum her hata ve zorluk türü için 6 hücreli oluşturulan tablo üzerinde açıklanmıştır. Bu tablo yapısı bir görev için çalışan öğrencilerin karşılaştıkları zorluk ve hataları belirleme amacıyla analiz edilebilir anlamlı bilgi yapıları oluşturmaktadır. Bu çalışma ile elde edilen sonuçlar ile programlama sürecinde yapılan hataların ve karşılaşılan zorlukların programlama bilgi türleri üzerinden nasıl şekillendiğini ve hangi zihinsel süreçler sırasında gerçekleştiğini ortaya koymaktadır.

Özetle; bu çalışmada programlama problemlerinin zorluk durumlarının belirlenmesi, süreçte karşılaşılan hata ve zorluklar üzerinden içsel bilişsel yük kaynaklarının belirlenmesi ve problemlerin zorluğu ile algılanan bilişsel yük durumlarının ilişkilendirilmesi

üzerine odaklanılmıştır. Elde edilen bulguların programlama problemlerinin çözümü sürecinde öğrencilerin algıladıkları yüklerin temel kaynaklarının programlama dilinin yapısı, problemlerinin çözümlerinin içerdikleri programlama yapıları, öğrencilerin çözüm yaklaşımları gibi unsurlar çerçevesinde gerçekleştiği değerlendirilmektedir. Bu noktada önerilen zorluk belirleme yolunun, öğrencilerin deneyimleyecekleri zorlukları ve karşılaşacakları muhtemel hataları kestirmede başarılı sonuçlar oluşturabilmektedir.

6. SONUÇLAR VE ÖNERİLER

Bu bölümde araştırmadan elde edilen bulgulara analiz edilmiş ve araştırma için belirlenen problemler çerçevesinde özetlenmiştir.

6. 1. Sonuçlar

Bu çalışmada programlama öğreniminde içsel bilişsel yük oluşturan kaynakları belirlenmiştir. Uygulama öncesinde uygulamada kullanılan problemler belirlenmiş ve alan uzmanları yardımıyla problemlerin zorluk durumları analiz edilmiştir. Zorluk durumlarına göre basitten karmaşığa sıralanan problemler uygulama sürecinde öğrencilere çözmeleri için sunulmuştur. Bu süreçten elde edilen bulgular çalışma için belirlenen problemler çerçevesinde ortaya konulmuştur.

Çalışmanın sonuçları, programlama problemlerine yönelik önerilen zorluk belirleme yönteminin işe koşulabilirliği, öğrencilerin programlama sürecinde karşılaştıkları bilişsel yük kaynakları ve problemlerin zorluğu ile karşılaşılan bilişsel yük arasındaki ilişkiler bağlamında sunulmuştur. Bu çerçevede araştırmadan;

1. Programlama problemleri, problemde yer alan temel öğeler, öğe etkileşimleri ve sözdizimsel, anlamsal ve stratejik bilgi türleri gibi bilgi türlerini içeren iki boyutlu yapıda ortaya konulabilir.
2. Problemlerin zorluk durumlarının belirlenmesinde problemin çözümünde kullanılacak yapıların kullanım zorunluluğu en önemli unsur olmak üzere; ilgili problem çözümü için kullanılacak yapıların stratejik önemi gibi durumlar belirleyici rol oynamaktadır.
3. Alan uzmanlarının farklı çözümlerde uygun ortak önemli yapıları belirleme ve problemlerdeki gerekli etkileşimleri ortaya koyma noktasındaki değerlendirmeleri problemlerin zorluklarının belirlenmesinde önem arz etmektedir.
4. Sözdizimsel olarak karşılaşılan içsel bilişsel yük kaynakları girinti, noktalama, parantez ve harf hataları olarak ortaya çıkmıştır.
5. Anlamsal olarak karşılaşılan içsel bilişsel yük kaynakları yanlış kullanma, eksik kullanma, belirleyememe ve karar verememe hataları ve zorluklarıdır.
6. Stratejik olarak karşılaşılan içsel bilişsel yük kaynakları yanlış (yerde) kullanma, eksik kullanma, belirleyememe ve karar verememe olarak belirlenmiştir.

7. Problemlerin zorluğunun belirlenmesi için önerilen iki boyutlu gösterim yolu ile belirlenen zorluk durumu, öğrencilerin karşılaştıkları zorluk ve hatalar ile doğrusal bir ilişki göstermektedir. Problemler karmaşıklaştıkça ve zorlaştıkça anlamsal ve stratejik hata miktarı artmakta ve bu hata kaynakları çeşitlenmektedir.
8. Sözdizimsel hata kaynakları süreç boyunca azalmaktadır.
9. Öğrencilerin zorluk ve hata temelinde yaşadıkları olumsuz durumlar ile bilgi türlerinden en çok stratejik bilgi gerektiği durumlarda karşılaşılmaktadır.
10. Programlama dilinin yapısı, problemlerin doğası ve öğretim yöntemi gibi durumlar öğrencilerin karşılaştıkları bilişsel yük kaynaklarının şekillenmesinden rol oynamaktadır.

6. 2. Öneriler

Çalışmanın bu bölümünde elde edilen sonuçlar doğrultusunda programlama sürecinde içsel bilişsel yük oluşturan kaynakların belirlenmesine yönelik araştırma yapacak olan akademisyenlere, öğretmenlere ve öğrencilere yol gösterici bazı önerilere yer verilmiştir.

6. 2. 1. Araştırma Sonuçlarına Dayalı Öneriler

1. Bu çalışmada elde edilen sonuçlar arasında en sık yapılan hataların ve karşılaşılan zorlukların stratejik hatalar olduğu bilinmektedir. Stratejik bilgi türü, doğrudan problemin çözümüyle alakalıdır. Gerekli olan yapıları doğru yerde işe koşabilmeyi gerektirir. Bu türde yapılan hatalar ve karşılaşılan zorlukların arkasında öğrencilerin problem çözme ve algoritma oluşturma noktasında yaşadığı sorunlar yatmaktadır. Bu anlamda öğrencilere problem çözme örnekleri sunmak ve bu alanda deneyim kazandırmak bu hata kaynaklarının azalmasında rol oynayabilir.
2. Bilgi işlemsel düşünme becerisinin programlama ile kazandırılmasında programlamanın hata kaynaklarının göz önünde bulundurulması, bu düşünce sisteminin kazandırılmasında önemli rol oynayacağı gözden kaçırılmamalıdır.
3. Öğrencilere programlama öğretimi yapılırken öncesinde problemlerin zorluk durumu belirlenmesi, öğrencilerin basitten zora doğru örneklerle karşılaşmaları ve bu şekilde hata kaynaklarından az etkilenmeleri mümkün olabilir.
4. Programlama editörü tasarımcıları sıklıkla rastlanan sözdizimsel, anlamsal ve stratejik olarak yapılan hataların farkına vararak bunlar için çözüm üreten

mekanizmaları editörlere eklemeleri programlama öğrenenler için faydalı olabilir. Bunun yanında programların sadece sözdizimsel anlamda şeklen hataları göstermeyip, anlamsal ve stratejik olarak da yardımcı olabilecek yapıların ortamlara eklenmesi problemlerin çözümünde kolaylaştırıcı rol oynayabilir.

6. 2. 2. İleride Yapılabilecek Araştırmalara Yönelik Öneriler

1. Bu çalışmada problemlerin zorluk durumunun belirlenmesi ile ilgili önerilen yol, bilgi türü ve programlama yapıları üzerine kurulmuştur. Bu yol programlama süreci, programlama için temel problemlerin çözümünde uygun biçimde işe koşulabilmiştir. Benzer biçimde bu yola kullanılan editör bağlamında farklı yapıların eklenmesiyle ayrıntılandırılarak tümüyle programlama sürecine genellenebilen daha geniş bir çerçeve oluşturulabilir.
2. Bu çalışma 5 öğrenci ile yürütülmüştür. Öğrenci sayısının arttırılmasıyla önerilen yöntemin genellenebilirliği ile ilgili daha kesin yargılara varılabilir.
3. Çalışmada veri toplama araçları olarak ekran kayıtları, sesli düşünmeler, beden dili hareketleri ve gözlem formu kullanılmıştır. Elde edilen verilerin analizinde bu veri toplama araçlarından elde edilen veriler birlikte analiz edilmiştir. Bu analiz sırasında ekran kayıtları merkeze alınmış, diğer araçlar destekleyici rol oynamıştır. Farklı çalışmalarda özellikle beden dili, sesli düşünmeler gibi doğrudan davranışları yansıtan durumlar merkeze alınarak bilişsel yükün davranışlar bağlamındaki göstergeleri üzerinden analizler yapılabilir.
4. Bu çalışmanın uygulama süreci 8 problemle yürütülmüştür. Problem sayısının arttırılmasıyla belirlenen hata ve zorlukların kaynaklarının çeşitlenmesi sağlanabilir.
5. Örneklem daha uzun süre gözlenerek süreçteki temel aşamaları şekillendiren davranışların genellenebilirliği ile ilgili çalışmalar yapılabilir.

7. KAYNAKLAR

Anvari, F., Tran, H. M. T., and Kavakli, M. (2013). Using cognitive load measurement and spatial ability test to identify talented students in three-dimensional computer graphics programming. *International Journal of Information and Education Technology*, *3*(1), 94-99.

Ayres, P. (2006). Impact of reducing intrinsic cognitive load on learning in a mathematical domain. *Applied Cognitive Psychology: The Official Journal of the Society for Applied Research in Memory and Cognition*, *20*(3), 287-298.

Aytaçlı, B. (2012). Durum çalışmasına ayrıntılı bir bakış. *Adnan Menderes Üniversitesi Eğitim Fakültesi Eğitim Bilimleri Dergisi*, *3*(1), 1-9.

Barut, E., Tuğtekin, U., ve Kuzu, A. (2016). Programlama eğitiminin bilgi işlemsel düşünme becerileri bağlamında incelenmesi. *4. Uluslararası Öğretim Teknolojileri ve Öğretmen Eğitimi Sempozyumunda sunulan bildiri,* (210-214). Elazığ: Fırat Üniversitesi.

Başkale, H. (2016). Nitel araştırmalarda geçerlik, güvenirlik ve örneklem büyüklüğünün belirlenmesi. *Dokuz Eylül Üniversitesi Hemşirelik Fakültesi Elektronik Dergisi*, *9*(1), 23-28.

Bayman, P., and Mayer, R. E. (1983). A diagnosis of beginning programmers' misconceptions of basic programming statements. *Communications of the ACM, 26*(9), 677-679.

Berland, M., and Wilensky, U. (2015). Comparing virtual and physical robotics environments for supporting complex systems and computational thinking. *Journal of Science Education and Technology*, *24*(5), 628-647.

Bundy, A. (2007). Computational thinking is pervasive. *Journal of Scientific and Practical Computing*, *1*(2), 67-69.

Bonar, J., and Soloway, E. (1983). Uncovering principles of novice programming. In *Proceedings of the 10th ACM SIGACT-SIGPLAN Symposium on Principles of Programming Languages* (10-13).

Cevahir, H., ve Özdemir, M. (2017, Mayıs). *Programlama öğretiminde karşılaşılan zorluklara yönelik öğretmen görüşleri ve çözüm önerileri*. Uluslararası Bilgisayar ve Öğretim Teknolojileri Sempozyumunda sunulan bildiri, İnönü Üniversitesi Bilgisayar ve Öğretim Teknolojileri Bölümü, Malatya.

Chao, P. Y. (2016). Exploring students' computational practice, design and performance of problem-solving through a visual programming environment. *Computers & Education*, *95*, 202-215.

Chen, O., Kalyuga, S., and Sweller, J. (2017). The expertise reversal effect is a variant of the more general element interactivity effect. *Educational Psychology Review*, *29*(2), 393-405.

Cook, M. P. (2006). Visual representations in science education: The influence of prior knowledge and cognitive load theory on instructional design principles. *Science Education*, *90*(6), 1073-1091.

Cooper, G. (1998). Research into cognitive load theory and instructional design at UNSW. Retrieved from http://education.arts.unsw.edu.au/CLT_NET_Aug_97.HTML October 12, 2006.

Coull, N. J., and Duncan, I. M. (2011). Emergent requirements for supporting introductory programming. *Innovation in Teaching and Learning in Information and Computer Sciences*, *10*(1), 78-85.

Çakiroğlu, Ü., Suiçmez, S. S., Kurtoğlu, Y. B., Sari, A., Yildiz, S., and Öztürk, M. (2018). Exploring perceived cognitive load in learning programming via Scratch. *Research in Learning Technology*, 26, *1-19*.

Djambong, T., and Freiman, V. (2016). Task-Based Assessment of Students' Computational Thinking Skills Developed through Visual Programming or Tangible Coding Environments. *International Association for Development of the Information Society,* 41-51.

Eryılmaz, S. (2003). *Algoritma tasarlama ve programlamaya giriş*. Ankara: Detay Yayıncılık.

Gerjets, P., Scheiter, K., and Catrambone, R. (2004). Designing instructional examples to reduce intrinsic cognitive load: Molar versus modular presentation of solution procedures. *Instructional Science*, *32*(1-2), 33-58.

Gomes, A., and Mendes, A. J. (2007, September). *Learning to program-difficulties and solutions*. In International Conference on Engineering Education–ICEE, Coimbra: University of Coimbra.

Gökçek, T., ve Davey, L. (2009). Durum çalışması değerlendirmelerinin uygulaması. *İlköğretim Online*, *8*(2), 1-3.

Gülbahar, Y. (2018). *Bilgi işlemsel düşünme ve programlama konusunda değişim ve dönüşümler.* Ankara*:* Pegem Atıf İndeksi.

Hsu, J. M., Chang, T. W., and Yu, P. T. (2012). Learning Effectiveness and cognitive loads in ınstructional materials of programming language on single and dual screens. *Turkish Online Journal of Educational Technology-TOJET*, *11*(2), 156-166.

International Society for Technology in Education. (2016). *2016 ISTE Standards for Students*. *International Society for Technology in Education*.

De Jong, T. (2010). Cognitive load theory, educational research, and instructional design: Some food for thought. *Instructional Science*, *38*(2), 105-134.

Joppe, M. (2000), The Research Process. Retrieved from http://www.ryerson.ca/~mjoppe/rp.htm February 25, 1998.

Kalelioglu, F., Gülbahar, Y., and Kukul, V. (2016). A framework for computational thinking based on a systematic research review. *Baltic Journal of Modern Computing*, *4*(3), 583-596.

Kazimoglu, C., Kiernan, M., Bacon, L., and MacKinnon, L. (2012). Learning programming at the computational thinking level via digital game-play. *Procedia Computer Science*, *9*, 522-531.

Kester, L., Kirschner, P. A., and Van Merriënboer, J. J. (2005). The management of cognitive load during complex cognitive skill acquisition by means of computer-simulated problem solving. *British Journal of Education Psychology, 75*(1), 71-85.

Kinnunen, P., and Malmi, L. (2008). CS minors in a CS1 course. In *Proceedings of the Fourth international Workshop on Computing Education Research.* Sidney.

Kolfschoten, G., Lukosch, S., Verbraeck, A., Valentin, E., and de Vreede, G. J. (2010). Cognitive learning efficiency through the use of design patterns in teaching. *Computers & Education*, *54*(3), 652-660.

Lahtinen, E., Ala-Mutka, K., and Järvinen, H. M. (2005). A study of the difficulties of novice programmers. *Acm Sigcse Bulletin*, *37*(3), 14-18.

Leahy, W., and Sweller, J. (2008). The imagination effect increases with an increased intrinsic cognitive load. *Applied Cognitive Psychology: The Official Journal of the Society for Applied Research in Memory and Cognition*, *22*(2), 273-283.

Leymun, Ş. O., Odabaşı, F., ve Yurdakul, I. K. (2017). Eğitim ortamlarında durum çalışmasının önemi. *Eğitimde Nitel Araştırmalar Dergisi*, *5*(3), 367-385.

Leppink, J., Paas, F., Van der Vleuten, C. P., Van Gog, T., and Van Merriënboer, J. J. (2013). Development of an instrument for measuring different types of cognitive load. *Behavior Research Methods*, *45*(4), 1058-1072.

Lincoln, Y. S., and Guba, E. G. (1982). Establishing dependability and confirmability in naturalistic ınquiry through an audit. *Paper presented at the Annual Meeting of the American Educational Research Association.* New York.

Lye, S. Y., and Koh, J. H. L. (2014). Review on teaching and learning of computational thinking through programming: What is next for K-12?. *Computers in Human Behavior*, 41, 51-61.

Malan, K., and Halland, K. (2004). Examples that can do harm in learning programming. In *Companion to the 19th Annual ACM SIGPLAN Conference on Object-oriented Programming Systems, Languages, and Applications*. Vancover.

Milne, I., and Rowe, G. (2002). Difficulties in learning and teaching programming—views of students and tutors. *Education and Information Technologies*, *7*(1), 55-66.

Moons, J., and De Backer, C. (2013). The design and pilot evaluation of an interactive learning environment for introductory programming influenced by cognitive load theory and constructivism. *Computers & Education*, *60*(1), 368-384.

Moreno, R. (2006). When worked examples don't work: Is cognitive load theory at an impasse?. *Learning and Instruction*, *16*(2), 170-181.

Moreno, R., and Valdez, A. (2005). Cognitive load and learning effects of having students organize pictures and words in multimedia environments: The role of student interactivity and feedback. *Educational Technology Research and Development*, *53*(3), 35-45.

Mow, I. C. (2008). Issues and difficulties in teaching novice computer programming. In *Innovative Techniques in Instruction Technology, E-Learning, E-Assessment, and Education* (pp. 199-204). Springer, Dordrecht.

Özdinç, F., and Altun, A. (2014). Factors effecting information technology teacher trainees' programming process. *Elementary Education Online*, *13*(4), 1531-1541.

Özmen, B., and Altun, A. (2014). Undergraduate students' experiences in programming: Difficulties and obstacles. *Turkish Online Journal of Qualitative Inquiry*, *5*(3), 1-27.

Paas, F., Tuovinen, J. E., Tabbers, H., and Van Gerven, P. W. (2003). Cognitive load measurement as a means to advance cognitive load theory. *Educational Psychologist*, *38*(1), 63-71.

Patton, M. Q. (2014). *Nitel araştırma ve değerlendirme yöntemleri*. Ankara: Pegem Akademi.

Pinto-Llorente, A. M., Casillas-Martín, S., Cabezas-González, M., and García-Peñalvo, F. J. (2018). Building, coding and programming 3D models via a visual programming environment. *Quality & Quantity*, *52*(6), 2455-2468.

Pollock, E., Chandler, P., and Sweller, J. (2002). Assimilating complex information. *Learning and Instruction*, *12*(1), 61-86.

Renumol, V., Jayaprakash, S., and Janakiram, D. (2009). Classification of cognitive difficulties of students to learn computer programming. *Indian Institute of Technology, India*, 1-*12*.

Saygıner, Ş., ve Tüzün, H. (2017, Mayıs). *Programlama eğitiminde yaşanan zorluklar ve çözüm önerileri*. Uluslararası Bilgisayar ve Öğretim Teknolojileri Sempozyumunda sunulan bildiri, İnönü Üniversitesi Bilgisayar ve Öğretim Teknolojileri Bölümü, Malatya.

Seufert, T., Jänen, I., and Brünken, R. (2007). The impact of intrinsic cognitive load on the effectiveness of graphical help for coherence formation. *Computers in Human Behavior*, *23*(3), 1055-1071.

Sezgin, M.E. (2009). *Çok ortamlı öğrenmede bilişsel kuram ilkelerine göre hazırlanan öğretim yazılımının bilişsel yüke, öğrenme düzeylerine ve kalıcılığa etkisi.*

(Yayımlanmamış doktora tezi). Çukurova Üniversitesi, Sosyal Bilimler Enstitüsü, Adana.

Schmeck, A., Opfermann, M., van Gog, T., Paas, F., and Leutner, D. (2015). Measuring cognitive load with subjective rating scales during problem solving: differences between immediate and delayed ratings. *Instructional Science*, *43*(1), 93-114.

Stachel, J., Marghitu, D., Brahim, T. B., Sims, R., Reynolds, L., and Czelusniak, V. (2013). Managing cognitive load in introductory programming courses: A cognitive aware scaffolding tool. *Journal of Integrated Design and Process Science*, *17*(1), 37-54.

Sweller, J., van Merri¨enboer, J. J. G., and Paas, F. (1998). Cognitive architecture and instructional design. *Educational Psychology*. Review, 10(3), 251–296.

Sweller, J. (1994). Cognitive load theory, learning difficulty, and instructional design. *Learning and Instruction*, *4*(4), 295-312.

Sweller, J., and Chandler, P. (1994). Why some material is difficult to learn. *Cognition and Instruction*, 12(3), 185–233.

Tepgeç, M. (2017). *Algoritma Öğretiminde Çözümlü Örnek Kullanımının Öğrenci Başarısına ve Bilişsel Yüke Etkileri* (Yayımlanmamış yüksek lisans tezi). Hacettepe Üniversitesi, Eğitim Bilimleri Enstitüsü, Ankara.

Tindall-Ford, S., Chandler, P., and Sweller, J. (1997). When two sensory modes are better than one. *Journal of Experimental Psychology*, 3(4), 257–287.

Van Merrienboer, J. J., and Sweller, J. (2005). Cognitive load theory and complex learning: Recent developments and future directions. *Educational Psychology Review*, *17*(2), 147-177.

Van Merrienboer, J. J., Kester, L., and Paas, F. (2006). Teaching complex rather than simple tasks: Balancing intrinsic and germane load to enhance transfer of learning. *Applied Cognitive Psychology: The Official Journal of the Society for Applied Research in Memory and Cognition*, *20*(3), 343-352.

Vogel-Walcutt, J. J., Gebrim, J. B., Bowers, C., Carper, T. M., and Nicholson, D. (2011). Cognitive load theory vs. constructivist approaches: which best leads to efficient, deep learning?. *Journal of Computer Assisted Learning*, *27*(2), 133-145.

Yıldırım, A. ve Şimşek, H. (2013). Sosyal bilimlerde nitel araştırma yöntemleri. (9. Baskı). Ankara:SeçkinYayıncılık

Yılmaz, G. K. (2014). Developing a belief scale according to using computer technology in mathematics teaching. *Procedia-Social and Behavioral Sciences*, *152*, 613-618.

Yousoof, M., Sapiyan, M., and Kamaluddin, K. (2006). Reducing cognitive load in learning computer programming. *Proceedings of the World Academy of Science, Engineering and Technology*, *12*, 259-262.

Yurdugül, H., and Aşkar, P. (2013). Learning programming, problem solving and gender: A longitudinal study. *Procedia-Social and Behavioral Sciences*, *83*, 605-610.

Zainal, Z. (2007). Case study as a research method. *Jurnal Kemanusiaan*, *5*(1), 1-6.

Weintrop, D., and Wilensky, U. (2015). Using Commutative Assessments to Compare Conceptual Understanding in Blocks-based and Text-based Programs. In *International Computing Education Research* (Vol. 15, pp. 101-110). Chicago: Northwestern University.

Wing, J. (2006). Computational Thinking. *Communications of the Association for Computing Machinery*, 49(3), 33-35.

8. ÖZ GEÇMİŞ VE İLETİŞİM BİLGİLERİ

15.06.1995 tarihinde Gaziantep'te doğdu. İlkokul, ortaokul ve lise eğitimini Gaziantep'te tamamlayan araştırmacı, 2013 yılında Karadeniz Teknik Üniversitesi Eğitim Bilimleri Fakültesi Bilgisayar ve Öğretim Teknolojileri Öğretmenliği bölümünü kazandı. 2017 yılında lisans öğrenimini tamamlayan araştırmacı aynı yıl Karadeniz Teknik Üniversitesi Eğitim Bilimleri Enstitüsü Bilgisayar ve Öğretim Teknolojileri Eğitimi Ana Bilim Dalı Tezli Yüksek Lisans programına kabul edildi. Mezun olduktan 1 yıl sonra özel bir okulda Bilişim Teknolojileri öğretmeni olarak görev yaptı. Şu an Tepebaşı Anadolu Lisesi'nde Bilişim Teknolojileri öğretmeni olarak görev yapmaktadır.

İLETİŞİM BİLGİLERİ

Adres: İbni Sina Mahallesi. 400 Nolu Cadde No:82 Şahinbey/Gaziantep

E-Posta : sevvalbilgi61@gmail.com

Tel : 0543 442 52 80