ÇOKLUORTAMIN BİLGİSAYAR PROGRAMLAMA BAŞARISI ÜZERİNE ETKİSİ

THE EFFECTS OF MULTIMEDIA ON COMPUTER PROGRAMMING ACHIEVEMENT

KEREM GÜLTEKİN

Hacettepe Üniversitesi

Lisansüstü Eğitim-Öğretim ve Sınav Yönetmeliğinin

Bilgisayar ve Öğretim Teknolojileri Eğitimi Anabilim Dalı için Öngördüğü

YÜKSEK LİSANS TEZİ

olarak hazırlanmıştır

2006

Fen Bilimleri Enstitüsü Müdürlüğü'ne

Bu çalışma jürimiz tarafından **BİLGİSAYAR VE ÖĞRETİM TEKNOLOJİLERİ EĞİTİMİ ANABİLİM DALI'NDA YÜKSEK LİSANS TEZİ** olarak kabul edilmiştir.

Başkan .....
(Prof. Dr. Petek AŞKAR)

Üye .....
(Prof. Dr. Buket AKKOYUNLU)

Üye .....
(Doç. Dr. Arif ALTUN)

Üye .....
(Yrd. Doç. Dr. Deniz DERYAKULU)

Üye (Danışman) .....
(Dr. Halil YURDUGÜL)

ONAY

Bu tez ...../ ...../..... tarihinde Enstitü Yönetim Kurulunca belirlenen yukarıdaki jüri üyeleri tarafından kabul edilmiştir.

...../ ...../.....

Prof. Dr. Ahmet R. ÖZDURAL
FEN BİLİMLERİ ENSTİTÜ MÜDÜRÜ

ÇOKLUORTAMIN BİLGİSAYAR PROGRAMLAMA BAŞARISI ÜZERİNE ETKİSİ

Kerem GÜLTEKİN

Öz

Bu araştırmada Çokluortama Dayalı Eğitim Yazılımının Programlama Eğitiminde öğrenci başarısı üzerine etkisi araştırılmıştır.

Araştırma deseni olarak öntest – uygulama – sontest deseni kullanılmış, kalıcılık için ise sontest uygulamasından 20 gün sonra bir kalıcılık testi uygulanmıştır. Kontrol gurubuna (18 kişi) alışılagelmiş öğretim yöntemleri kullanılarak ders içeriği verilmiştir. Deney grubuna (16 kişi) ise araştırmacı ve konu alanı uzmanı tarafından geliştirilmiş Çokluortama Dayalı Eğitim Yazılımı uygulanmıştır. Bu yazılım, bilgisayar programlama dersi kapsamındaki temel kavramlar olan "Değişken", "Bellek", "Operatör", "Program" gibi kavramları ve kavramlar arası ilişkileri bir problem durumu yardımıyla, çokluortam teknolojileri kullanarak verilmesini amaçlar.

Araştırma sonucunda programlama öğretiminde Çokluortama Dayalı Eğitim Yazılımının, alışılagelmiş öğretim yöntemine kıyasla öğrenci başarısı üzerinde istatistiksel olarak anlamlı bir etkisi bulunduğu saptanmıştır. Yine bu araştırma göstermiştir ki, ilk defa programlama eğitimi alan bireyler ile daha önceden formal veya informal olarak programlama eğitimi almış bireylerin ön test başarı puanları arasında fark bulunmuşken, son testte başarı puanları arasında bir fark gözlenmemiştir.

Programlama eğitiminde alışılagelmiş eğitim yöntemlerine alternatif olarak kullanılabilecek Çokluortama Dayalı Eğitim Yazılımları, eğitim kalitesinin yükseltilmesi ve anlamlı öğrenme sağlanması için, bilgisayar programlama derslerinde kullanılabilecek bir materyaldir. Programlama eğitimi sürecinde geliştirilecek olan materyal, araştırmada verilen özelliklere göre düzenlenir ve geliştirilirse sonuç, kazanımlar doğrultusunda olumlu olacaktır.

Anahtar Kelimeler : Eğitim Yazılımı, Çokluortama Dayalı Eğitim Yazılımı, Programlama Dilleri, Programlama Dilleri Öğretimi, Programlama Dilleri Öğrenimi

Danışman : Dr. Halil YURDUGÜL, Hacettepe Üniversitesi, Bilgisayar ve Öğretim Teknolojileri Eğitimi Anabilim Dalı

THE EFFECTS OF MULTIMEDIA ON COMPUTER PROGRAMMING ACHIEVEMENT

Kerem GÜLTEKİN

Abstract

In this study, “The Effects of Multimedia Based Instructional Software on Achievement of Using The Teaching Programming” has been searched.

Pretest-process-posttest model has been used as a research model. 20 days after the posttest, a permanence test has been given to the students. The content of computer programming concepts has been given to Control group (18 person) with using traditional instruction methods. The Multimedia Story Based Instructional Software which is developed by researcher and content expert has been applied to Experiment group(16 person). The aim of The Software is to teach basic concepts of computer programming lessons such as “Variables”, “Memory”, “Operator” and “Program” etc. using technics of multimedia technologies with based on the story.

The research results show that, Multimedia Based Instructional Software is more effective than traditional instruction methods on students achievement in programming education. Another result from the research proved that there was a difference between pretest results of the students who had programming education for the first time and those who had programming education formally and informally before but no difference was observed between achievement scores from the post test results.

Multimedia Based Instructional Software can be used as an alternative material to the traditional instruction methods which will increase instruction quality and provide meaningful learning. During the programming education process, if the material will be developed and arranged according to the given features, the outcome will be positive through gainings.

Keywords : Instructional Software, Multimedia Based Instructional Software, Programming Language, Teaching Programming Language, Learning Programming Language

Advisor : Dr. Halil YURDUGÜL, Hacettepe University, Computer Education and Instructional Technologies M. Sc.

Teşekkür

Araştırma sürecinde hem fikrime yön çizen hem de her basamakta yardımlarını hiç bir şekilde esirgemeyen, sadece bilgi olarak değil manevi olarak da çok büyük desteğini gördüğüm, lisans eğitimimde de tüm bilgi birikimini sonsuzca bana sunan ve aynı zamanda araştırmanın danışmanlığını üstlenen sayın Dr. Halil YURDUGÜL'e sonsuz saygı ve teşekkürlerimi sunarım.

Çalışmanın ilk şekillenmesinden son haline kadar, üstün bilgi birikimi ile bana destek veren, gerek lisans gerekse lisansüstü eğitimimde sonsuz minnettar olduğum sayın Prof. Dr. Petek AŞKAR'a sonsuz teşekkür ederim.

Çokluortama Dayalı Eğitim Yazılımı'nın geliştirilme basamağında eleştirileriyle yol gösteren ve uygulama basamağında büyük yardım aldığım sayın Dr. Alev ÖZKÖK'e teşekkür ederim.

Şu ana kadar olan akademik yaşantımda kazandığım bilgi birikimimi borçlu olduğum sayın Prof. Dr. Buket AKKOYUNLU, Yrd. Doç. Dr. Mukaddes ERDEM, Yrd. Doç. Dr. Yasemin USLUEL ve Yrd. Doç. Dr. S. Sadi SEFEROĞLU'na sonsuz teşekkürler.

Bu çalışmanın kağıda dökülmesi sürecinde her türlü bilgi birikiminden yararlandığım, manevi olarak desteğini her zaman hissettiğim, bana her türlü imkanı sunan Ömer Emin DEDE'ye sonsuz teşekkürlerimi sunarım

İçindekiler Dizini

Çizelgeler Dizini

Şekiller Dizini

Simge ve Kısaltmalar

ÇDEY : Çokluortama Dayalı Eğitim Yazılımı

BDÖ : Bilgisayar Destekli Öğretim

OOP : Object Oriented Programming (Nesne Yönelimli Programlama)

P : I. tür hata olasılığı

1. Giriş

1.1. Problem Durumu

Günümüzde bilim ve teknolojideki gelişmelerin eğitimde kullanılmasıyla birlikte eğitimsel kavramlar ve öğrenme-öğretme süreci de bu gelişmelerden etkilenmiştir. Artık okur-yazarlık kavramı yerini bilgi ve/veya bilgisayar okur-yazarlığına, alışılagelmiş öğretim ise yerini bilgisayara dayalı öğretime bırakmaya başlamıştır. Bununla birlikte iletişim teknolojileri sayesinde yoğun bir şekilde uzaktan öğretim çalışmaları hız kazanmıştır.

Bilgi ve iletişim teknolojilerindeki bu gelişmişlik eğitimsel amaçlara yönelik olarak özellikle; (a) öğrenmenin niteliğini artırır, (b) öğrenme ve öğretimle ilgili hedeflere ulaşmadaki zamanı kısaltır, (c) öğretmenin etkinliğini artırır, (d) maliyeti azaltır ve (e) öğrenciyi ortamda etkili kılar (Akkoyunlu, 1994).

Aşkar (1990) ise bilgisayarların öğretimde kullanılmalarını dört ana başlıkta toplamaktadır: (a) Alıştırma ve tekrarlar, (b) bire-bir öğretim, (c) problem çözme, (d) laboratuar çalışmaları ve deneyler. Aşağıda bunlarla ilgili detaylara yer verilmiştir.

a) Alıştırma ve Tekrarlar

Aşkar (1990)'a göre, bilgisayarların en yaygın uygulama alanlarından biri, işlenmiş konularla ilgili alıştırma ve tekrar yaptırma amacı ile kullanılmasıdır. Bilindiği gibi alıştırma ve tekrar yaptırmanın amaçları;

- Bilgi ve becerilerin pekiştirilmesi,
- Öğrenmenin kalıcılığının sağlanması,
- Üst seviyedeki davranışların öğrenilmesine zemin hazırlanması şeklinde sıralanabilir.

Üst düzey davranışların (analiz, sentez vb.) kazanılması için alt düzey davranışların yerleşmesi gerekir. Bir beceri ile ilgili ne kadar tekrar yapılırsa o

beceri o kadar kalıcı olur. Bilgisayar destekli öğretimde yukarıda belirtilen amaçların yerine getirilmesi bilgisayarlar yolu ile sağlanmaktadır (Aşkar, 1990).

b) Bire-bir Öğretim

Birçok araştırma göstermektedir ki her ideal öğrenme, bir öğretmenin yalnızca bir öğrenci ile çalışması sonucu gerçekleşen öğrenmedir. Günümüz koşullarında böyle bir eğitim sisteminin mümkün olamayacağı açıktır. Öte yandan bilgisayarların okullarda kullanılması ile bire-bir öğretim uygulamalarına yavaş yavaş başlanmıştır. Bu uygulamalar, bir konu ile ilgili olgu, yöntem, kavram, ilke, genelleme ve kanunların bilgisayardan öğrenilmesini amaçlamaktadır (Aşkar, 1990).

Yine Aşkar (1990)'a göre bire-bir öğretimde öğretim materyali bilgisayar tarafından verilir. Öğretimde bulunması gereken öğeler:

- Öğrencinin dikkatini çekme,
- Öğrenciyi hedeften haberdar etme,
- Ön bilgileri hatırlatma,
- Uyarıcıyı sunma ve rehberlik sağlama,
- Davranışı ortaya çıkartma,
- Davranışı değerlendirme,

olduğuna göre, bilgisayar programlarının geliştirilmesinde bunlara dikkat edilmelidir. Ayrıca yazılımlar kitap gibi olmamalı, bilgisayarın hareket ve ses gibi avantajlarından yararlanılmalıdır.

Bilgisayarla bire-bir öğretimde öğrenci kendi hızına göre çalışır. Ayrıca öğrenci istediği kadar tekrar yapma olanağına da sahiptir (Aşkar, 1990).

Öğretmenin tahtaya bir grafiği veya şekli çizmesi oldukça fazla vakit alır. Oysa bilgisayarla öğretim zamanı kısaltmakta ya da bu zaman içinde daha fazla uygulama yapma şansı yaratmaktadır (Aşkar, 1990).

Aşkar (1990)'a göre, iyi ve emek verilerek hazırlanılmış bir program öğrenci gereksinimlerine cevap verebilen bir programdır. Bazı öğrenciler çok hızlı öğrenirler ve öğretmenin vermeyi hedeflediğinin çok ötesine gidebilirler. Öte yandan bazı öğrenciler yavaş öğrenirler ve öğrenme gereksinimleri diğerlerine göre daha değişiktir. Bir program bu tür ihtiyaçları karşılayabilirse uygulamaların başarılı olma olasılığı artar.

Bire-bir öğretim için hazırlanan programlar, bazı sebeplerle dersi kaçıran öğrencilerin derse yetişmelerini sağlamak için de kullanılmaktadır. Böylece, kaçırdığı konularla ilgili yazılımlarla çalışan öğrenci bir sonraki derse hazır gelebilmektedir (Aşkar, 1990).

Günümüzde birçok ders içeriğine ilişkin öğretim yazılımları geliştirilmiştir. Aşkar (1991)'a göre, temel becerilerin öğretimi, pekiştirilmesi ve kalıcılığının sağlanmasından başlayarak problem çözme, model geliştirme, kritik düşünme gibi üst düzey hedeflerinin gerçekleştirilmesinde bilgisayarların tartışılmaz bir yeri vardır. Yine Aşkar (1992), farklı bilgi, beceri ve tutum düzeyindeki bireylerden oluşan bir sınıfta, bilgisayar aracılığıyla her bireye kendi yeteneğinde gelişmelerine olanak sağlanmakta, çeşitli beklentiler karşılanabilmektedir.

Bire-bir öğretim kapsamında programlı öğretime dayalı olarak çok sayıda ders için çeşitli eğitim yazılımları geliştirilmiştir. Bununla birlikte Aşkar (1991)'ın da belirttiği gibi daha üst düzey beceri gerektiren dersler için de bilgisayarla öğretim gereklidir ve bu yazılımların geliştirilme çalışmaları devam etmektedir. Bu derslere örnek olarak bilgisayar programlama dilleri gösterilebilir.

c) Problem Çözme

Aşkar (1990)'a göre eğitimin en önemli görevlerinden biri, öğrencilerde, karşılaştıkları problemleri çözme becerisini geliştirmektir. Problem çözümüyle ile ilgili yapılan çalışmalar bunun genellikle beş basamaktan oluştuğunu göstermektedir. Bunlar;

1) Problemi belirleme,

2) Problemi tanımlama,

3) Problemi çözme için strateji geliştirme,

4) Stratejileri uygulamaya koyma,

5) Sonuçları değerlendirme,

dir. Öğrencilerde bu tür becerilerin kazandırılması öğrenme-öğretme bütünlüğünün belli başlı amacı durumundadır (Aşkar, 1990).

Bilginin kalıcılığı açısından bakıldığında ise problem çözme adımı, öğrencilerin bilginin güncel hayatta hangi problemi çözdüğünü görüp anlamlandırmasına olanak verir. Bunun için Brown, Collins ve Duguid (1989) tüm öğretim sürecini gerçek yaşamdan alınma problem çözme durumlarına dayandırmaya çalışmışlardır. Örneğin yeni teoremler geliştirmek veya kanıtlar bulmak için, matematik öğretiminde, bankaya olan mesafanin uzunluğu, marketten yapılan alışverişin tutarı gibi güncel hayattan problemler kullanmışlardır.

Wilson ve Cole(1991)'a göre öğrenciler yeni bilgiler üzerinde akıl yürütmeye çalışırlar. Burada genel olarak karşılaşılan soru, sınıftaki öğrenilenlerle gerçek hayattaki ilişkinin ne olduğu ve bu bilginin hangi işlere yarayacağıdır. Şayet bilgi gerçek bir problemin çözümünden elde edilmiş ise, problemin çözümüne yardımcı olmuş ise bu soruların büyük bir kısmı cevaplanmış olur (Wilson & Cole, 1991).

Bilgi, bir problemin çözümüne yardımcı olmuş ise anımsanabilir şekilde beyinde saklanır. Bir bilgi problem çözme aşamasında kazanılmış ise yine problem çözme durumunda kolayca hatırlanabilir (Wilson & Cole, 1991).

d) Laboratuar Çalışmaları ve Deneyler

Yine Aşkar (1990)'a göre laboratuar çalışmaları, genellikle ilke, kural ve kanunları gerçek ya da yapay durumlar yaratarak öğrenme etkinlikleridir. Bu tür çalışmaların özellikle fen öğretiminde önemi açıktır. Deneyler genellikle iki tür amaca hizmet etmektedir:

1) Öğrencinin yaparak öğrenmesi sağlanarak, öğretilen temel ilkeler arasındaki ilişkiyi anlaması,

2) Öğrencinin ilke ve kanunlara varma yollarını öğrenmesi veya keşfetmesi.

1.2. Programlama Dilleri Becerileri

Bilgisayar programlama dilleri, içerisinde üst düzey becerilerin yer alması nedeniyle öğretimi ve öğrenimi zor bir kapsamdır. Programlama dillerinin zorluğu, aynı zamanda öğrenme içeriğinin geniş bir yelpazede ele alınması ile de açıklanabilir. Bu dersin içeriğini kısaca özetlemek gerekirse;

a) Bilgisayar okur-yazarlık bilgisi
b) Donanım bilgisi
c) Programlamaya ilişkin temel kavramlar bilgisi
d) Programlama dilinin “dil yapısı” bilgisi
e) Problem çözme becerisi

Burada sıralanan bilgilerin kısa açılımları aşağıda verilmiştir.

a) Bilgisayar okur-yazarlık bilgisi:

Programlama dillerinin amacı, herhangi bir platformda çalışabilecek programlar oluşturmaktır. Bu nedenle programlama için öncelikle öğrenenlerde işletim sistemine ilişkin temel bilgilerin olması/verilmesi gerekmektedir. Ancak bu bilgi programlama dilleri dersi kapsamından daha çok bu derse ilişkin ön öğrenmeler kapsamında ele alınabilir.

b) Donanım bilgisi:

Programlama dillerinin her bir komutu, bilgisayarın bir donanımına iş yaptırmaya ve/veya donanımı kullandırmaya yöneliktir. Bu nedenle, öğrenenlerin

bilgisayarın donanım bilgisine sahip olması ya da öncelikle bu bilginin öğretilmesi gerekir. Bu bilgi de, bilgisayar okur-yazarlık bilgisinde olduğu gibi ön öğrenme olarak kabul edilmelidir.

c) Programlamaya ilişkin temel kavramlar bilgisi:

Programlama becerilerindeki yaygın inanışın aksine belki de en önemli bilgi olarak ortaya çıkmaktadır. Günümüzde informal programlama dilleri eğitimine sahip çoğu kişi "komut" kavramını bilmeden komutlar yazmakta, "değişken" kavramını bilmeden değişkenleri kullanmaktadır. Programlama dilleri birçok kavramı içerisinde barındıran bir öğrenme alanıdır ve formal bir eğitim için ön ve öncelikli öğrenmeler temel kavramlar ile başlamalıdır.

d) Programlama dilinin "dil yapısı" bilgisi:

Programlama dillerinin anlamlı en küçük parçası "atom" (token) olarak bilinir. Atomlar bir araya gelerek sözcükleri (lexical), sözcükler kurallı bir şekilde bir araya gelerek sözdizilimini (syntax) oluştururlar. Sözdizilimleri anlamlı olduğu sürece bilgisayara bir iş yaptırabildikleri için aynı zamanda anlamlı (semantic) olmak zorundadır. Buna göre bilgisayar programlama dilleri günlük yaşamda kullanılan diller ile büyük benzerlik gösterir. Günümüzde sözcüksel ve sözdizilimsel olarak birbirinden farklılık gösteren çok sayıda programlama dili geliştirilmiş olmasına karşın tüm bu dillerin başvuru kaynağı (referans) olarak Backus-Naur gramatik yapısı kullanılır.

e) Problem çözme becerisi:

Programlar, komutların (deyimlerin) bir araya getirdiği bir bütündür. Komutlar bir araya gelerek programları, programlar bir araya gelerek yazılımları oluşturur. Ancak komutların sıralanması gelişigüzel değil, aksine bir problemi çözmek üzere planlı şekilde tasarlanıp yapılmalıdır. Bu nedenle, öğrenenlerin bir problemi çözmek için bir "program tasarımı" yapmaları gerekmektedir. Bu durum ise problem çözme becerisi ile açıklanabilir.

1.3. Programlama Becerileri ve Eğitim Yazılımları

Yakın zamana kadar bilgisayar okur-yazarlığı; temel bilgisayar kullanma becerileri ile açıklanırken günümüzde bilgisayar okur-yazarlığı kapsamına "programlama dilleri becerileri"de alınmaya başlanmıştır. Bu durum okullarda programlama dilleri dersinin önemini artırmaktadır. Alışılagelmiş programlama dilleri öğretimi, sınıf ya da laboratuar ortamlarında alışılagelmiş öğretim yöntemleri ile gerçekleşmektedir.

1.3.1. Programlama Öğretimi ve Tarihsel Gelişimi

Programlama dilleri eğitimine yönelik eğitimsel araştırmaların çoğu öğrencilerin programlamaya giriş dersleri boyunca kazandıkları programlama bilgisi ve programlama öğrenirken öğrencilerin gerçekleştirdikleri bilişsel işlemlerle ilgilenmişlerdir. Buna göre programlama öğrenimi süresince birbirleriyle ilişkili üç tip programlama bilgisinden söz edilir (Bayman & Mayer, 1988).

- Yazımsal (Syntactic) Bilgi: Belirli bir programlama diline ait kullanım kurallarının bilgisi.
- Kavramsal (Conceptual) Bilgi: Programlama kavramlarının ve prensiplerinin bilgisi
- Problem Çözme – Stratejik (Strategic) Bilgi: Programlama ile ilgili problem çözme becerisi

Bayman ve Mayer'in bu modeli Shneiderman (1977), Shneiderman ve Mayer (1979) ve Linn (1985)'in önceki araştırmalarının geliştirilmiş halidir.

Shneiderman ve Mayer (1979) programlama öğrenimi sırasında kazanılan bilgi çeşitlerinden yazımsal-mantıksal model üzerinde durmuşlardır. Onların modelinde yazımsal bilgi, bir programlama dilinin belirli bir döngü yapısının yazım kuralı bilgisi gibi kuralları ve işlemsel bilgileri içerir. Mantıksal bilgi ise, programlama dilinden bağımsız olarak programlamanın yapılarını ve prensiplerini anlamaktır.

Bayman ve Mayer (1988) ise daha önceden yapılmış bu çalışmanın gelişmişi olarak kendi çalışmalarında üç tip bilgi türünden söz eder. Bunlar, (a) Yazımsal bilgi, (b) Kavramsal bilgi, (c) Stratejik bilgi. Yazımsal bilgi dil özellikleri veya gramer bilgisi olarak tanımlanır. Örnek olarak Pascal programlama dilinde "Repeat-Until" döngüsünün yazımı veya her komutun sonuna noktalı virgül konulması gibi durumlar ele alınırsa; bu teknik bilgi derlenecek olan programın yazımı için gereklidir, fakat bir probleme çözüm üretmek ya da tasarım yapmak için yeterli değildir.

Bayman ve Mayer (1988)'e göre kavramsal bilgi, program içindeki dil özelliklerinin kombinasyonları içerisine gömülü olan kavramları anlama olarak tanımlanmıştır. Yazımsal bilgi belirli bir dilin yapısal kurallarının bilgisidir. Fakat kavramsal bilgi, bir probleme çözüm üretebilmek için bu yapıların mantıklarının ve ne iş yaptıklarının tam olarak anlaşılması bilgisidir.

Yine Bayman ve Mayer (1988)'in tanımı olan stratejik bilgi ise yazımsal ve kavramsal bilgileri de kullanarak ilk defa karşılaşılan bir probleme çözüm getirebilme yeteneğidir. Stratejik bilgi karşılaşılan problemi fark etme, tanımlama ve çözümü için teknik yollar geliştirebilmedir.

Bayman ve Mayer (1988) tarafından yapılan deneysel çalışmada, kavramsal ve stratejik bilginin ilişkili olup olmadığı denenmiştir. Bu çalışma, BASIC programlama dilini öğrenen ve programcılığa yeni başlayan bireyler üzerinde yapılmıştır. Bu çalışma göstermiştir ki, kavramsal bilgi ile problem çözme performansı (stratejik bilgi) birbirleriyle güçlü bir ilişki içerisindedir.

Daha öncede değinildiği gibi; bilgisayar programlama dillerinin öğretimi karmaşık bir süreçtir. Bunun temel nedenlerden bazıları; programlama dili öğretiminin programlama dillerinin kavramsal öğrenme ürünlerini, dil yapısını ve problem çözme gibi zihinsel becerileri kapsaması ve aynı zamanda programlama öğreniminin geniş bir yelpazede ele alınması olarak görülebilir. Benzer şekilde programlama öğretimi de karmaşık bir süreç gerektirir. McGill ve Volet (1997) bu süreci öğrenim ve öğretim boyutuyla bir tabloda şematikleştirmiştir (Çizelge 1).

	Bildirimsel Bilgi (Declarative)	İşlemsel Bilgi (Procedural)
Kavramsal Bilgi (Conceptual)	Program çalıştırıldığında hangi mantıksal işlemlerin yapıldığını anlama ve açıklama becerisi **Örn** : Yalancı (Pseudo) kodun ne iş yaptığını açıklayabilme	Bir programlama problemine yönelik çözüm oluşturabilme. **Örn** : Bazı dataların ortalamalarını bulan bir "Procedure" dizayn edebilme.
Yazımsal Bilgi (Syntactical)	Programlama dilinin yazım kuralları bilgisi **Örn** : Java programlama dilinde her komutun sonuna noktalı virgül işaretinin konulması gerektiğini bilme.	Program yazarken yazım kurallarına uyabilme **Örn** : Java programlama dilindeki While komutunu yazım kuralı olarak doğru yazabilme
Stratejik Bilgi (Problem Solving)	Alışılmışın dışında bir problemle karşılaşıldığında buna ait bir çözüm üretmek için program dizayn, kodlama ve test edebilme becerisi.	

Çizelge 1 : McGill ve Volet (1997)'in Programlama Dilleri Öğrenim ve Öğretim Tablosu

Bildirimsel bilgi, öğrencilerin bir programlama dilinin kurallarını bilmesi olarak adlandırılır.

İşlemsel bilgi ise öğrencilerin kullandıkları programlama dilinin kurallarına uyarak kod yazabilmesi olarak ifade edilebilir.

Yazılımsal bilgi bir programlama dilinin işaretsel kurallarını belirtir. Bu tip bilgi öğrenenlerin kendi problemlerini çözmeleri için bir programı doğru biçimde kodlayıp bu programı derlemeleri ve program geliştirmeleri için gerekli/şart bir bilgidir (Bayman & Mayer, 1988). Diğer bilgi türü *Kavramsal Bilgi* ise programlama işlemi sırasında kullanılan prensipler, yapılar ve mantık bilgisi olarak adlandırılır. Yine Bayman ve Mayer (1988)'e göre en karmaşık bilgi türü Stratejik bilgidir. *Sratejik bilgi* daha önce karşılaşılmamış bir probleme çözüm üretebilme bilgisi olarak tanımlanır.

Programlama bilişsel birçok kavramın yanında fazlaca beceri gerektirir. McGill ve Volet (1997) ise programlamada, öğrenenlerin kazanmaları gereken birbiri ile ilişkili üç bilgi tipinden söz eder. Birincisi yazımsal, ikincisi kavramsal,

üçüncüsü ise stratejik veya problem çözmedir. Yazımsal bilgi bir programlama diline ait belirli kuralların bilgisi ve bu kuralları kullanabilme olarak tanımlanmıştır. Kavramsal bilgi bilgisayar programcılığının yapıları ve prensipleri olarak tanımlanabilir. Yazımsal ve kavramsal bilgiler öğrencilerin basit ya da daha önce sınıfta karşılaştıklarına yakın problemlere çözüm ve tasarım üretebilmeleri için gereklidir. Stratejik bilgi ise genel problem çözme yeteneğine verilen ad olarak tanımlanmıştır (Baldwin & Kuljis, 2001).

Programlama öğretiminin karmaşık yapısı günümüzde bu alanda yeni gelişmeleri de beraberinde getirmiştir. Geçmişte programlama dillerinin öğretimi alışılagelmiş yöntemlerle kullanılarak yapılmaya çalışılmıştır. Alışılagelmiş öğretim yöntemi olan sınıf ortamında sunum yöntemi ile bu beceriler öğrencilere kazandırılmaya çalışılmıştır. Alışılagelmiş öğretim yöntemlerinin ön plana çıkan olumsuzlukları; öğretenden öğrenene doğru bilgi akışı olması ve derslerin anlatım yoluyla iletilmesi, öğrenende hem bir motivasyon düşüklüğü hem de tam bir "anlamlı öğrenme" gerçekleştirilememesi olarak belirtilebilir. Konuların benzetim yoluyla aktarımı ve böylece öğrenende bir anlam oluşturulması gerekmektedir.

Bir sonraki bölümde programlama dillerinin öğretiminde kullanılan görsel sistemler ve bu sistemlerin bazı özellikleri konusuna değinilecektir.

1.3.2. Programlama Dillerinin Öğretiminde Görsel Sistemler ve Bu Sistemlerin Özellikleri

Görsel sistemler çeşitli metaforlar kullanarak programlama yapmayı sağlarlar. Olsen ve diğerlerine (1990) göre görsel nesneler şablon setleri, formlar, modüller, komutlar, program veya alt programlar olarak tanımlanır. Hirakawa ve diğerlerine (1990) göre nesneler, algoritmaların ve veri yapılarının simgeler halinde görselleştirilmesi ve buna benzer veri akış grafikleri olarak adlandırılır (Baldwin, Kuljis, 2000).

Reader (1985)'ın yaptığı araştırmada, insan zihninin metin tabanlı kaynakları okumaktan çok, görsel bilgileri daha iyi anlamlandırıp onlar arasındaki ilişkiyi daha sağlam kurduğu ortaya çıkmıştır.

- Bergin ve Martinez (1996)'e göre görselleştirme araçları, eğiticilere yeni eğitim yöntemleri sunduğundan ve öğrencilerin ilgilerini çektiğinden bilgisayar bilimleri eğitmenlerine örnek yaratma açısından oldukça yardımcıdır. Bu da hem eğitmenlerin hem de öğrencilerin motivasyonlarını daha da arttırıcı bir etkidir. Motivasyon açısından aşağıdaki önermeler göz önünde tutulmalıdır (Bergin & Martinez, 1996):

- Karmaşık kavramların daha sadeleştirilmesi ve anlamlandırılması için resimler ve görsel materyaller kullanılmalıdır.
- Ayrıca görselleştirme araçları eğitmenlere daha az zamanda daha etkili ve içerikli materyaller geliştirmek için de yardımcı olur.
- İyi tasarlanmış benzetimler öğrencilerin anlama düzeylerini yükseltir (Bergin & Martinez, 1996).

Kullanılacak olan görselleştirme tipi konunun ne olduğuna bağlı olarak resim veya animasyon olabilir. Animasyon olarak geliştirilen görselleştirme aracı, sunulacak verinin oldukça büyük veya karışık olması, hareketlendirilmesi gerekliliği ve kavramların birbirleriyle olan ilişkilerini sunma durumlarında kullanılmalıdır. Görselleştirme aracı sınıf sunumları, açık veya kapalı laboratuar uygulamaları ve alışılagelmiş ödevlerde kullanılabilir. Görselleştirme aracının asıl gücü soyut kavramları şekillerle desteklenmesidir. Konu başında, sonunda veya konu içerisinde tartışma ortamı yaratmak için kullanılabilir. Öğrencilere farklı bakış açıları kazandırmaya ve çok boyutlu soyut kavramları anlamlandırabilmeye olanak sağlar (Bergin & Martinez, 1996).

Bergin ve Martinez (1996)'e göre programlama eğitiminde üç farklı türde görselleştirme tipi vardır. Bunlar:

1. Program Görselleştirme: Bu tip görselleştirme aracı, bir programın ve onun verilerinin nasıl çalıştığını grafiksel olarak sunmak için kullanılır. (Bergin & Martinez, 1996)

2. Algoritma Animasyonları: Kod ve datalara sahip temel algoritmaların hangi işlemleri tamamladığını göstermek amacıyla kullanılır. (Bergin & Martinez, 1996)

3. Veri Görselleştirme: Bu tip görselleştirme araçları ise program kodu içerisinde veri tiplerinin neler olduğu ve hangi işlemlerden sonra son durumlarının ne olduğu hakkında bilgi verir. (Bergin & Martinez, 1996)

1.3.3. Eğitim Yazılımları, Metaforlar ve Analojiler

Geçen on yıllık süreçte bilgisayarlar ve İnternet dünya çapında çok popüler hale gelmiştir. Bu da insanların çoğunun bilgisayarı anlama, yazılım geliştirme ve büyüyen interneti yönetme gereği duymalarına yönlendirmiştir. Birçok öğrenci bilgisayar bilimlerinde okumayı tercih etmiştir fakat öğretim yöntemleri bu sürece adapte olamamıştır. (Miyadera et al, 2000) Bu nedenden ötürü eğitim-öğretim sürecinde yeni yöntemler geliştirilmeye başlanmıştır. Bilgisayarların gelişmesi ve eğitim sürecindeki yeni arayışlar sonucunda ortaya eğitim yazılımı kavramı çıkmıştır.

Öğretim sürecinde kullanılan eğitim yazılımlarının diğer öğretim materyallerine göre en önemli avantajlarının başında öğrenmeyi kolaylaştırıcı görsel öğrenme nesnelerini kullanması ve anlamlı öğrenme tekniklerinden olan *metafor* ve *analojileri* etkin bir şekilde kullanmaya olanak tanıması gelmektedir. Görsel sistemler kullanmak ve tasarlamak, bilgisayar programlama ilkeleri ve kavramlar üzerinde eğitim gören öğrencilerin bu kavramları öğrenmelerine fırsat tanır.

Metaforlar ve analojiler karmaşık kavramları açıklamak veya işlemleri tanımlamak için kullanılır. Programlama eğiticileri genellikle kendi sınıflarında yapılan etkinliklerde sözlü veya görsel metaforların çeşitlerini sık sık kullanırlar. Görsel metaforlar öğrencilerin bir problemi anlamasına yardımcı olmak amacıyla kullanılır ve öğrencilerin bilişsel transformasyon yapmalarını gerektirir. Görsel metaforlar, sözlü metaforlara göre çok daha kuvvetli olduklarından bilgisayar bilimleri kitapları genellikle betimleyici şekillerle doludur. Lakoff ve Johnson (1980) metaforları bir kavramı başka bir kavramın üzerinden anlatma ve deneme olarak açıklar. *Metaforlar* daha soyut kavramların güncel modellenmesini ve benzer objeler veya deneyimler ile ilişkilendirilmesini sağlar. Güzel seçilmiş bir metafor kullanıcılara aslında ne olduğunu gerçeklik beklentisi içerisinde anlama fırsatı sağlar. *Analoji* ise var olan bilginin esnek bir şekilde yeni duruma uyarlanmasına olanak sağlar. Analojinin standart modeli çok iyi anlaşılmış bir örnek, bir teori ya da bir problem çözümünün tam anlamıyla anlaşılmamış bir konuya kaynak olması olarak tanımlanabilir. (Baldwin & Kuljis, 2001)

Glyn ve diğerleri (1989)'ne göre ise analoji; iki tanım, kural veya formül arasında haritalama işlemi olarak tanımlanmıştır. Biyoloji dalındaki alyuvarları bir yerden başka bir yere madde taşıyan kamyonlara benzetme örneği analojiye iyi bir örnek olarak gösterilebilir. (Newby et al, 1995).

Diğer taraftan analojiler somut veya yeni kavramın gerçek hayattaki benzerlerinin karşılaştırılması olarak bilinir ve öğrencilere yardımcı olduğu düşünülür (Gentner & Gentner, 1983; Reigelut, 1983; Treagust, 1993).

Eğitim sürecinde analoji kullanma, öğrenciler üzerinde anlamlı öğrenme kurmak için en uygun eğitim yöntemlerinden biridir. Anlamlı öğrenme ve öğretme ise, yeni içeriği sunmadan önce, öğrencilerin ön bilgilerini anımsatarak ve bunlar arasında bağ kurarak gerçekleştirilir (Ausubel, 1960; Gagne et al, 1988). Bu anımsatma sürecinin bir basamağı da, öğrencilerin ön bilgileri ile sunulan içerik arasında anlamlı bir bağ yaratma yani "Analoji" (Analogy) kurmadır. (Treagust, 1993).

1.4. Çalışmanın Önemi

Günümüzde programlama dillerinin etkili öğretimine ilişkin çalışmalar devam etmektedir. Bu çalışmaların çoğu; program animasyonları, algoritma animasyonları vb. metin tabanlı ya da görsel eğitim yazılımları olarak ele alınmaktadır. Ancak bu çalışmayla birlikte ilk defa *problem durumuna dayalı çokluortam tabanlı* bir eğitim yazılımı hazırlanmıştır. Bu çalışmada öğrencilerin programlama dilleri öğrenimini günlük yaşamla ilişkilendiren ve anlamlı öğrenme ilkelerine göre düzenlenmiş bu eğitim yazılımını ile programlama dilleri öğretimi gerçekleştirilmiştir. Bu çalışma için geliştirilen "Çokluortama Dayalı Eğitim Yazılımı - ÇDEY" bu alandaki eğitim yazılımlarına bir ilk oluşturabilir.

Bununla birlikte, bugüne kadar geliştirilen yazılımlar özellikle programlama dilleri kapsamında "yazımsal" kazanımlara yönelik olduğu gözlenmiştir. Ancak programlama becerileri McGill ve Volet (1997)'in belirttiği gibi yalnızca yazımsal bilgiden değil, aynı zamanda kavramsal ve stratejik bilgiden de oluşmaktadır. ÇDEY bu yaklaşım ile oluşturulmuştur. Ancak çalışmanın sınırlılığı gereği bu araştırmada ÇDEY yalnızca kavramsal bilgiler modülü ile yapılandırılmıştır. Programlama öğretiminde kavramsal bilgilere dayalı olarak yapılan araştırmanın ilgili araştırmalar incelendiğinde bir benzerinin bulunmadığı gözlenmiştir.

1.5. Problem Cümlesi

Çokluortama Dayalı Öğrenme Ortamları'nın programlama öğretiminde başarı üzerine etkisi var mıdır?

1.5.1. Alt Problemler

Bildirimsel (declarative) konuda hazırlanmış Çokluortama Dayalı Eğitim Yazılımı (ÇDEY) ile alışılagelmiş öğretim yönteminin başarı yönünden farklılıkları var mıdır?

Bildirimsel konuda hazırlanmış ÇDEY'nın programlama başarısındaki etkisi cinsiyete göre farklılık gösterir mi?

Bildirimsel konuda hazırlanmış ÇDEY'nın programlama başarısındaki etkisi daha önce programlama dersi almış ve almamış bireylere göre farklılık gösterir mi?

Bildirimsel (declarative) konuda hazırlanmış ÇDEY ile alışılagelmiş öğretim yöntemi arasında kazanılan bilginin kalıcılığı yönünden fark var mıdır?

1.6. Varsayımlar

Bu araştırmanın varsayımları;

- Öğrenciler kendilerine yöneltilen soruları içtenlikle yanıtlamışladır.

1.7. Sınırlılıklar

Hazırlanmış olan ÇDEY yalnızca bildirimsel (declarative) bilgi basamağında geliştirilmiştir.

Hazırlanmış olan ÇDEY yalnızca programlama dillerindeki kavramsal yapı ile sınırlıdır.

Araştırmanın kalıcılık uygulamaları yalnızca Hacettepe Üniversitesi Bilgisayar ve Öğretim Teknolojileri Eğitimi Bölümü 1. Sınıf öğrencileri ile sınırlıdır.

2. Programlama Dilleri Öğretiminde Eğitim Yazılımı Kullanımı Üzerine Yapılan Araştırmalar

Programlama öğretiminde görsel sistemler ve eğitim yazılımı kullanma yöntemi yeni sayılabilecek bir yöntemdir. Günümüzde global çapta, bu konuda yapılan araştırma sayısı oldukça azdır. Bu bölümde programlama eğitimi süreci boyunca bir eğitim materyali olarak hazırlanan eğitim yazılımları hakkında bazı araştırmalara yer verilmiştir.

Miyadera ve diğerleri (2000)'nin yapmış olduğu bir araştırmada öğrencilere bir anket uygulanmıştır. C programlama dersi alan öğrencilerin doldurduğu ankette öğrencilerin deneyimlerine dayanarak en çok zorluk çektikleri konular saptanmıştır. Anket, Tokyo Üniversitesi Bilgi Teknolojileri Bölümünde okuyan 3 ayrı gruba uygulanmıştır. Öğrencilerin en çok zorladıkları bölümler, hata ayıklama ve bu hataları düzeltmektir. Öğrenciler, hata ayıklama sürecinde, programın ne yapacağını ve nasıl çalıştığını bilmek zorundadırlar. Sonuçlara göre, var olan sistemin, programın nasıl çalıştığını öğrencilere göstermek zorunluluğu olduğu görülmüştür.

Yine aynı çalışma için her öğrenci tarafından rahatça kullanılabilecek Java tabanlı bir program görselleştirme aracı tasarlanmıştır. Bu yazılım programcılıkta yeni olan öğrenciler üzerinde denenmiştir. Bu yazılımla gerçekleştirilen öğretim süreci öğrencilerin çeşitli sıralama algoritmalarını öğrenmelerinde yardımcı olmak amacıyla geliştirilmiştir. Her satır işlemci tarafından işlenirken farklı bir renk alır. Öğrenciler üzerinde yapılan deneysel çalışma sonucunda programlama gösterimi ve animasyon teknikleri içeren bu program öğrencilere programın nasıl çalıştığını göstererek programlama becerisi üzerinde olumlu bir etki göstermiştir.

Lai ve Repman (1996)'ın yapmış oldukları araştırmanın amacı programlama kavramlarının öğretiminde benzeşimin etkilerini araştırmaktır. İlk olarak katılımcılara programlama dilleri hakkında ne bildikleri, cinsiyet ve yaşları hakkında bir anket uygulanmıştır. Bir hafta sonra hazırlanan dersler bilgisayar laboratuarında uygulanmıştır. Daha sonra metotları sınayıcı bir uygulama

yapılmıştır. Veri analizinden anlaşıldığı üzere bilgisayar programlama dili öğreniminde kavramsal bilgiler söz konusu olduğunda benzeşim anlamlı derecede bir başarı artışı sağlamıştır.

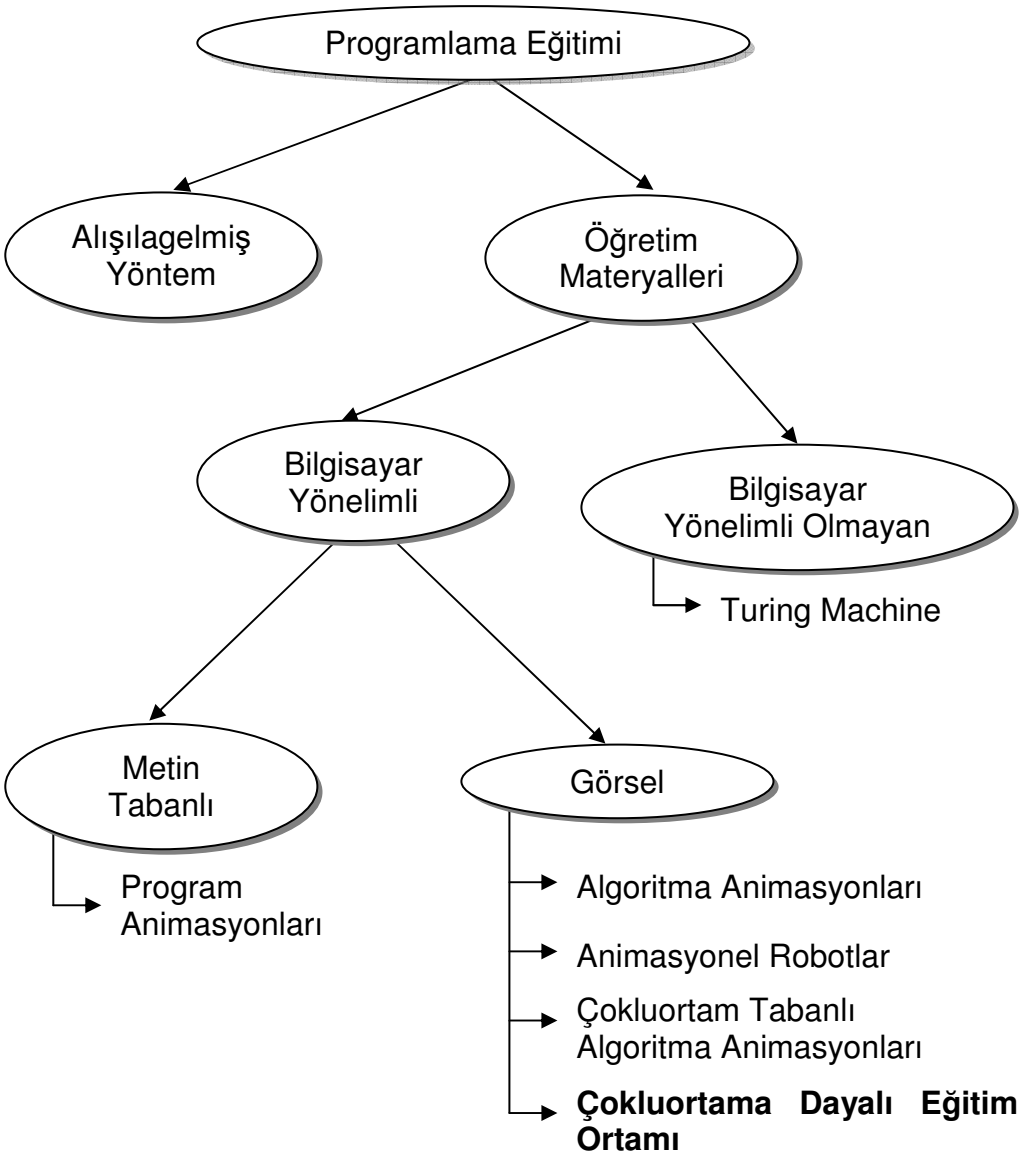
2.1.1. Programlama Öğretimi ve Eğitim Yazılımları

Üstünde incelemeler yapılarak öğrenilmesi gereken olgu, olay ve varlıkların benzeşimi bilgisayar aracılığı ile gerçekleştirilebilir. Tehlikeli ve karmaşık fizik, kimya deneyleri, mühendislik alanlarına ilişkin öğrenme-öğretme konuları gerçeğe son derece yakın biçimde bilgisayarla şematize edilebilir. Örneğin bir hidrolik veya elektronik devresi bilgisayar terminalinde izlenilebilir. Bu uygulamada öğrenci olası yanlışları kolayca görebilir. (Odabaşı, 1998)

Bilgisayarın benzeşim etkinliklerinde kullanımında öğretmen anlatacağı konuya ilişkin gerçek ve idealize durumları öğrenciler için hazırlama olanağına kavuşmaktadır. Bu kullanımda, karmaşık olgu ve olaylar bilgisayar yardımıyla sınıfa veya evlere getirilebilmektedir. Bu uygulama, bilgisayarı şimdiye kadar bilinen en etkili eğitim aracı yapacak güçtedir. Bu tür kullanımda bilgisayar, öğretilmesi söz konusu durumları daha somutlaştırma, ilişkilere hareket unsuru katma rolü oynayıp sonuçları açık biçimde ilgililerin yararına sunmaktadır. Kısaca belirtmek gerekirse gerçek yaşantıdaki olgu ve olayların çok iyi düzenlenmiş benzerlerini yaratma bilgisayar yardımıyla olanaklı hale gelmektedir. (Odabaşı, 1998)

Programlama sanatı, programlama araçlarının ve programlama dilinin bilgisi, problem çözme yeteneği, program tasarlayabilmek ve bunu uygulayabilmek için gerekli stratejileri içerir. Programlama eğitiminde yaygın yaklaşım, öncelikle programlama dilinin temel kavramlarını öğretmek ve bunun ardından öğrencilere programlama işlemi için etkili stratejiler sunmaktır. Üst düzey becerilerin kazandırılması için temel kavramların öğretimi oldukça önemlidir.

Alışılagelmiş öğretimin olumsuzluklarından dolayı, daha etkin programlama dilleri öğretimi için çeşitli öğretim materyalleri geliştirilmiştir (Şekil 1).



Şekil 1 : Geliştirilen Eğitim Materyallerinin Sınıflandırılması

Günümüzde programlama öğretiminde çokluortama dayalı öğrenme ortamları en çok çalışılan konuların başında gelmektedir. Bu nedenle eğitim materyali olarak kullanılmak üzere birçok eğitsel araç geliştirilmiştir.

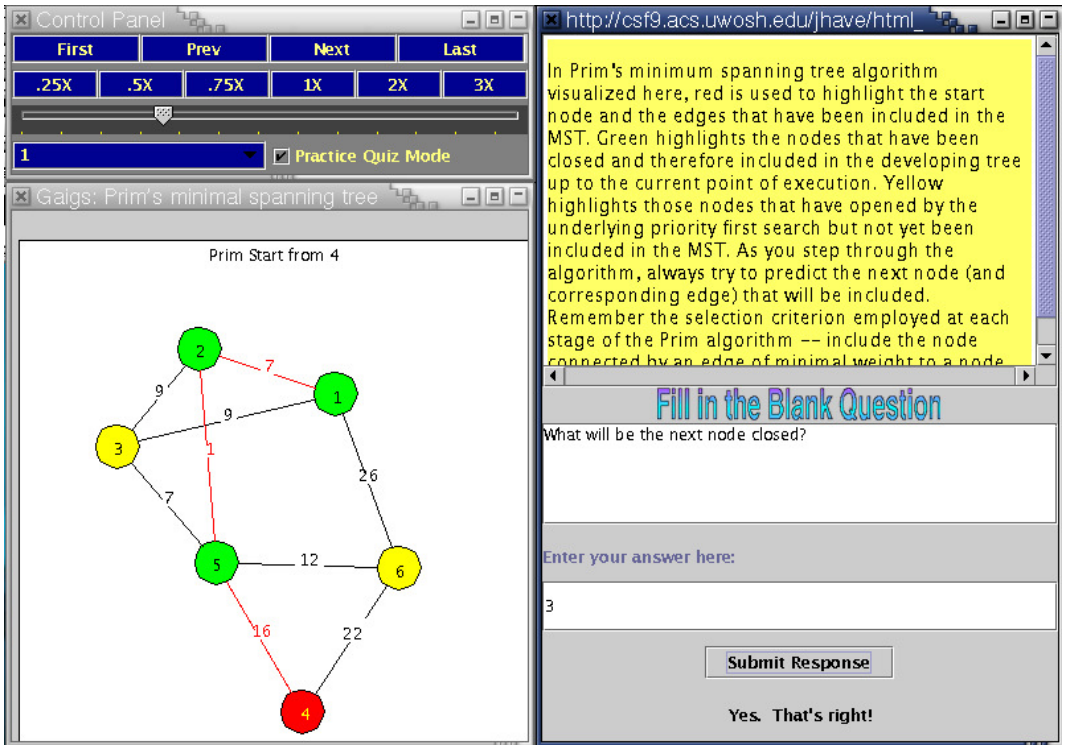
Bilgisayar algoritmalarını görselleştiren ve yeni animasyonlar yaratmaya yarayan birçok araç mevcuttur. Bazıları üç boyutlu etkiler içerir, bazıları ses kullanır. Bu sistemlerin asıl amacı programların veya algoritmaların dinamik görsel sunumlarını hazırlamak veya sunmaktır.

Bu araçların çoğu bilgisayar algoritmalarını görselleştirerek anlamayı kolaylaştırdıkları için eğitimsel bir araç olarak da kullanılırlar.

Bu araçlar genellikle belirli bir algoritma üzerinde sadece rol tabanlı simgeler ve animasyonlardan ibarettir. “Samba” adı verilen yazılım bu amaçla yazılmış programlardan bir tanesidir. Stasko 1996 yılında öğrencilerin etkileşimli bir şekilde basit komutlardan oluşan bir dil sayesinde kendi algoritmalarının animasyonlarını görmek isteyeceklerini ve bu yöntemin algoritma eğitimde önemli bir rol oynayacağını bildirmiştir. Geliştiriciler de “algoritma animasyon” aracı olan Samba’yı geliştirmişlerdir. Samba etkileşimli biçimde komutları yorumlayarak animasyonları oluşturan bir araçtır.

Yine bir “algoritma animasyon” aracı olan “The Sort Animator” isimli araç Dershem & Brummund (1998) tarafından tasarlanmıştır. Java Platformunun applet teknolojisi kullanılarak geliştirilen bu araçta hem algoritma kodunu hem de bu kodun animasyonunu aynı anda görebilmek için iki adet pencere bulunur. Eşzamanlı olarak işlem sırası hangi komutta ise o komut renklendirilip bu kodun sonucu olan animasyon da görüntülenir.

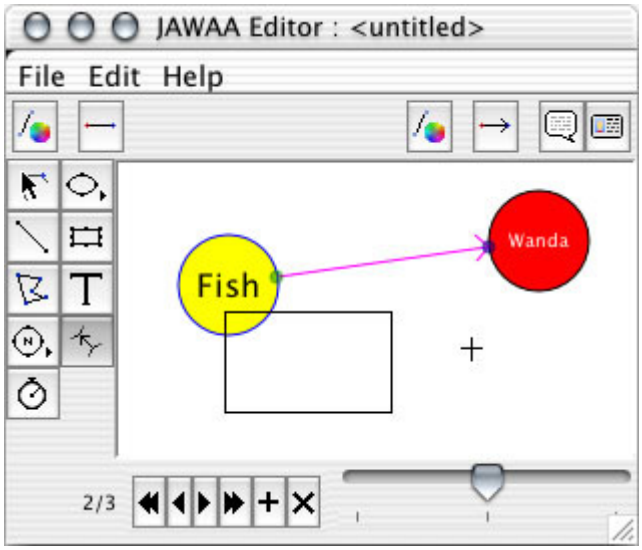
Naps (2000) ve arkadaşları tarafından geliştirilen JHAVE (Şekil 2) isimli istemci-sunucu mimarisine dayanan “algoritma animasyon” aracı internet üzerinde çalışmaktadır.



Şekil 2 : JHAVE Yazılımı Ekran Görüntüsü

Daha önceden programlama bilgisi olmayan bireyler üzerinde "Yapılandırmacılık" yaklaşımı da göz önünde bulundurulduğunda fazla anlamı olmayan bu materyaller günümüz eğitim sisteminde pek işe yaramamaktadır.

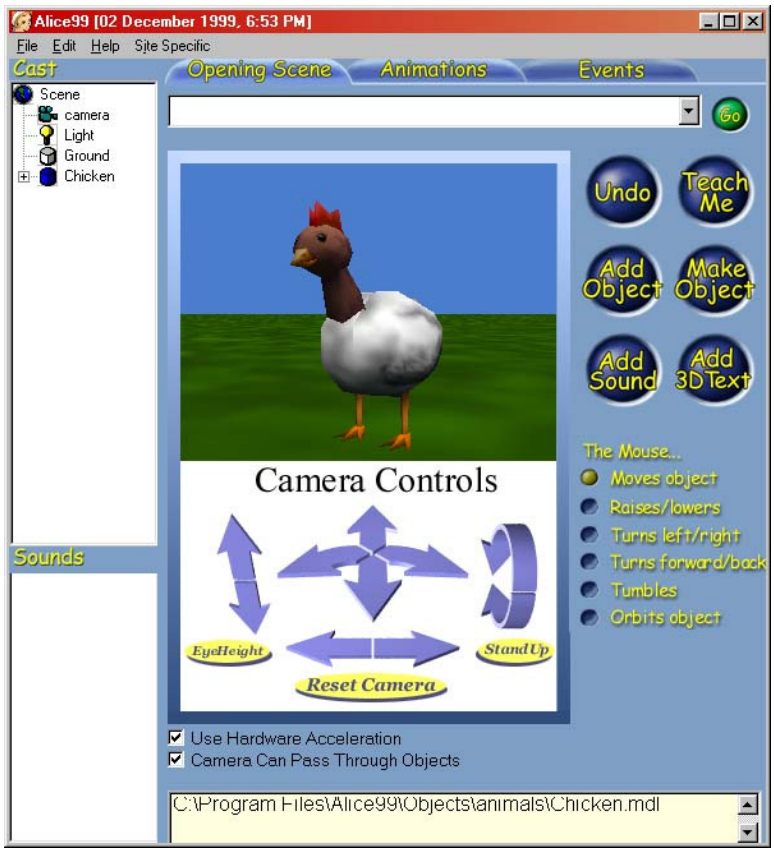
Algoritma animasyonlarının yanında programlama mantığını görsel halde aktarabilmek için çeşitli kahramanlarla ve araçlarla desteklenmiş yazılımlar da mevcuttur. Bu yazılımlar genellikle alıştırma yapmak ve belirli bir problemin çözümünü kendilerine has basit programlama dilleriyle sağlamak amacıyla yazılmışlardır. Durham'daki Duke Üniversitesi'nde bilgisayar bilimleri derslerinde öğrencilerin görsel yollardan kavramları öğrenebilmesi için kullanılan araçlardan bir tanesi de JAWAA'dır (Şekil 3). Yazılan programın algoritma animasyonunu çıkarabilen bu araç yazılan komutları HTML dilinde animasyonlara çevirmekte ve inernet üzerinde çalıştırılabilmektedir (Rodger, 2002).



Şekil 3 : JAWAA Yazılımının Ekran Görüntüsü

George (2000) tarafından geliştirilen EROSI isimli araç ise alt programların çağırılmasını sesler, renkler ve animasyonlar yardımıyla görselleştiren bir öğretim aracıdır. Bu araç yardımıyla programın akışını izlemek, alt programların program içerisinde çağırıldığı noktaları görmek ve programdaki aktif kontrol akışını görselleştirmek mümkündür.

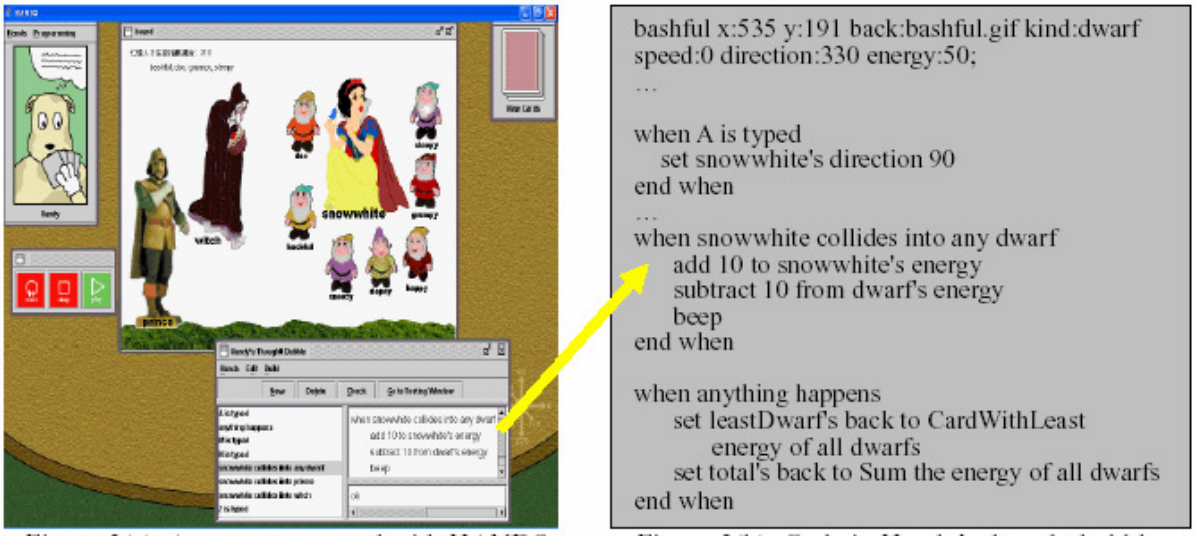
Alice (Şekil 4) isimli üç boyutlu etkileşimli animasyon aracı ise üç aşamada Carnegie Mellon Üniversitesi bünyesinde ve Randy Pausch yönetiminde geliştirilmiştir. Alice'in geliştirilme amacı "bilgisayar programlamasına yeni başlayanlar için, ilgi çekici üç boyutlu grafik animasyonlar geliştirebilmeyi kolaylaştırmak"tır.(Dann et al, 2000). Yapılan araştırmalarda, Saint Joseph Üniversitesi öğrencilerinden oluşan 21 kişilik grupta Alice ile CS1 dersindeki eğitim gören öğrencilerin başarılarında alışılagelmiş yöntemle eğitim görmüş diğer gruba göre anlamlı bir artış görülmüştür. (Cooper et al, 2003)



Şekil 4 : Alice İsimli Yazılımın Ekran Görüntüsü

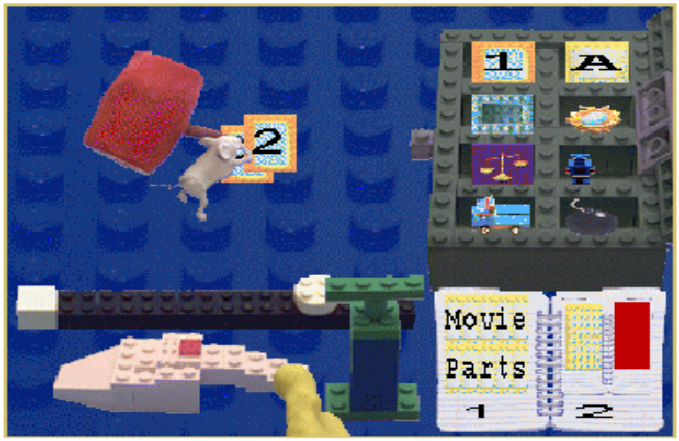
Backer (2001) ve arkadaşları Ontario'daki Waterloo Üniversitesi'nde Java tabanlı programlamaya giriş dersi için hazırlanmış metin tabanlı kaynakların tutum açısından zorluk yarattığını fark etmiş ve "Karel the Robot" isimli yazılımı geliştirmişlerdir (Şekil 11).

Stagecast Creator (Smith et al, 2000) yeni başlayan programcılar için görsel bir sistemdir (Şekil 5). Bu programın amacı hem öğretmen hem de öğrenciler için programlama simülasyonlarını yaratmak ve modifiye etmektir. Stagecast Creator programlama işlemlerini direk olarak değiştirmeye yarayan kuralları içerir ve benzeşimsel sunumlar için kullanılır.



Şekil 5 : StageCast Creator Yazılımının Ekran Görüntüsü

Bir diğer sistem ise yine çocuklar için yaratılmış ToonTalk'dur (Şekil 6). Amaç programlama dilinin yazım kurallarını eleyerek hareketlendirilmiş programlama yapmaktır. Programcı sanal bir dünya içerisinde karakterleri programlamanın soyut kavramları yerine anlaşılabilir kelimeler kullanarak hareketlendirebilir. ToonTalk'daki her şey görülebilir, tutulabilir ve değiştirilebilir. Çocuklar kuşlara çeşitli mesajlar vererek onların bir şeyler yapmasını programlayabilir, robotların kutular üzerinde iş yapmasını sağlayabilir, kamyonları yükleyebilir ve hareketlendirilmiş araçları kullanarak bunları kopyalayıp, silip, uzatabilir. Programlaya yeni başlayan her yaştaki öğrencilerin öğrenme tipleri hemen hemen aynıdır. Yetişkinler ve çocuklar benzer yollarla öğrenir. Bu yüzden ToolTalk sadece çocuklar için değil yetişkin eğitimlerinde de kullanılabilir(Baldwin & Kuljis, 2001)



Şekil 6 : ToonTalk'da Farenin 2+2 İşlemini Yapması

Esteves ve Mendes (2004)' göre günümüzde programlama derslerine nesne yönelimli (Object Oriented) diller ile başlama yöntemi oldukça yaygındır. İlk başlarda C ile başlanılan bu dersler artık Java ile anlatılmaktadır. Fakat çoğu öğrenci OOP dizaynı hakkında hala algoritmik ve programlama açısından zorluklar yaşamaktadır.

Bunun sebeplerinden bazıları şunlardır:

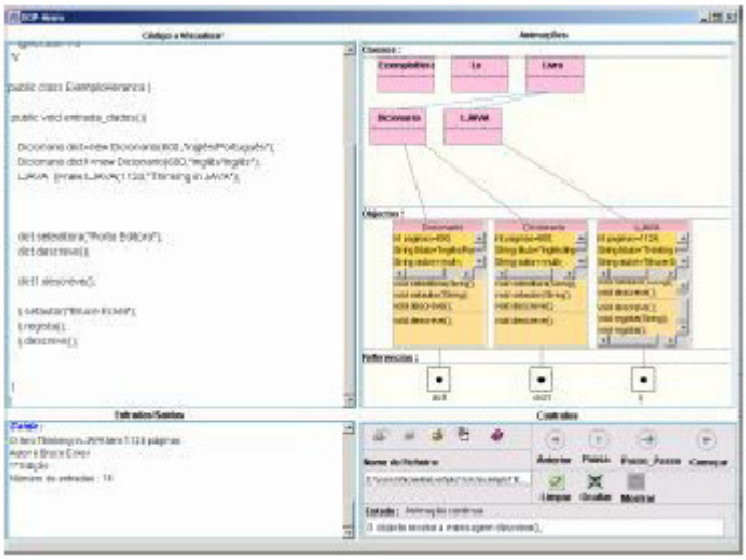
Programlama dinamik bir doğaya sahiptir, fakat çoğu öğrenme materyali (metinsel kitaplar) durağan formattadır ve dolayısıyla bu materyaller üzerinden programın dinamik davranışlarını anlamak oldukça güçtür.

Programlamanın soyut doğası çoğu öğrencinin program yapısının nasıl çalıştığını ve problemlerin nasıl çözüme kavuştuğunu görmelerini ve anlamalarını olumsuz yönde etkilemektedir

Programlama öğrenimi diğer derslere nazaran daha pratik yaklaşımları olan ve sadece teorik değil bir çok pratik bilgiye de bağımlı olan bir konudur.

Öğrencilerin bu gibi zorlukları aşabilmeleri için animasyon tabanlı simülasyonlar geliştirilmiş ve kullanılmıştır. Bu uygulamalar yardımıyla program dinamiklerini görsel hale getirmek ve öğrencinin kendi öğrenme ritmine göre pratik görevler vermek çok daha kolay olmuştur. Yine bu simülasyonlar sayesinde öğrencilerin programları anlamaları, var olan bir programı sınamaları ve yeni bir program geliştirmeleri sağlanabilir.

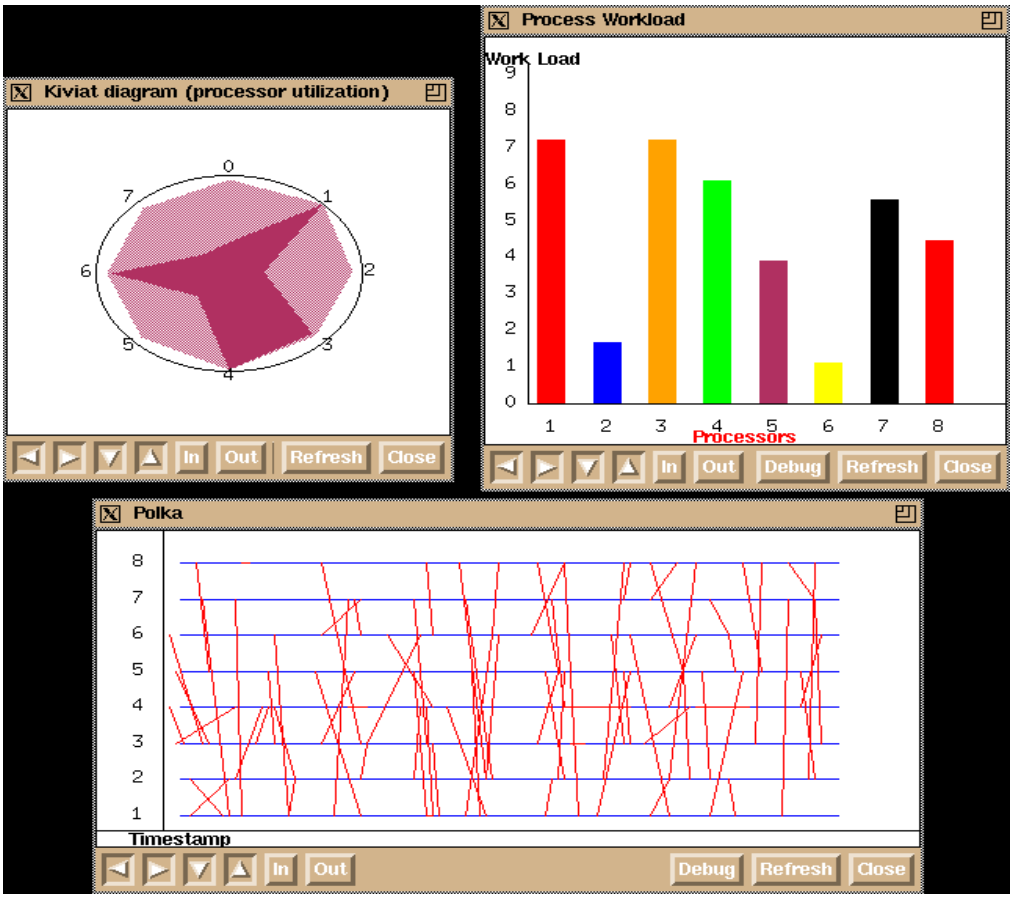
Bu yüzden araştırmacılar OOP-Anim (Şekil 7) isimli basit Java programları hareketlerle görselleştiren bir ortam geliştirmişler. Bu araç sayesinde öğrenciler var olan örnekleri sınayarak daha iyi öğrenme gerçekleştirebilirler ve kendi yazdıkları programları simüle ederek hatalarını düzeltebilirler. Ayrıca bu program öğrencilerin evde kendi başına kullanabilecekleri ve sınıf içinde grup olarak da çalışabilecekleri bir ortamdır.



Şekil 7 : OOP-Anim Yazılımının Ekran Görüntüsü

Bunun yanında şu anda piyasada problem çözme becerisinin geliştirilebilmesi için çeşitli eğitsel oyunlar da mevcuttur. Bu oyunlar kullanıcının yazılımla direk zihinsel etkileşimde bulunmasına olanak verir. Kullanıcının daha önce karşılaşmadığı problemleri çözmelerine dayalı bu yazılımlar oldukça iş görür hale gelmiştir.

Polka Algorithm animasyonu (Şekil 8) adı verilen bir araç üzerinde bir ağaç (Tree) algoritması animasyon haline getirilmiş ve öğretmen tarafından Georgia Institute of Technolgy öğrencilerinden toplam 62 öğrencilik bir gruba izlettirilmiştir. Daha sonra bir başarı testi uygulanmıştır. Bu araştırmanın sonucunda, bu uygulamaya katılan öğrencilerin başarılarının anlamlı ölçüde arttığı gözlemlenmiştir.



Şekil 8 : POLKA Yazılımın Ekran Görüntüsü

Öğrenme kaynakları, öğrencilerin programlama hakkındaki yapıları ve bilgileri öğrenmeleri için içerik oluşturur. Alışılagelmiş olarak bu kaynaklar, kitaplar ve öğretmenlerin hazırladığı notlardır. Çoğu programlama kursu internet üzerinden verilen dersler, sınavlar ve simülasyonlar yerine hala baslı materyalleri kullanır. (Garner, 2003)

Garner (2003) programlama öğretiminde bir yazılımın yaşam döngüsünü şu şekilde tanımlar:

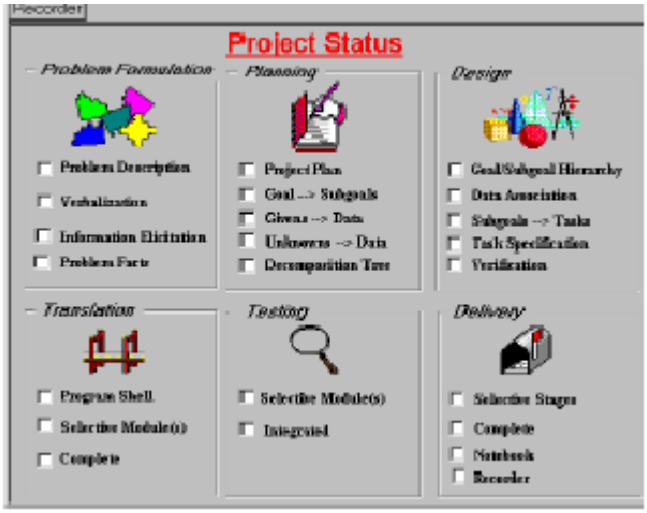
Adım 1 : Problemin analizi
Adım 2 : Çözüm veya algoritma dizayn etmek ve geliştirmek
Adım 3 : Algoritmayı uygulamak

Bu aşamalar detayları ile şu şekilde açıklanabilir :

Adım 1 : Problemin analizi

Programlama öğrenimi geçen yıllara rağmen çok fazla değişime uğramadı. Genellikle öğretmenler, yeni kontrol yapılarını veya veri yapılarını sunar, öğrencilere bir kaç örnek gösterir ve onlardan karşılaştıkları problemi çözmelerini beklerler. Bu konuda öğrencilerin yakındıkları konu hep öğretmenin anlattıklarını izleyip anladıkları fakat kendi problemlerine çözüm geliştiremedikleri olmuştur. (Garner, 2003)

Sınırlı olan bu alanda öğretmenlere bu konuda yardımcı olabilecek bir araç olan SolveIt (Şekil 9) video klipler ve mini dünyalar içerir. Bu program öğrencilerin programlama öğrenmesinde onlara destek olacak bütünleştirilmiş çevresel bir prototiptir. İçerisinde problem formülleştirici ile birlikte yardımcı olarak problem çözme, planlama, tasarım ve test aşamalarını içeren bu program Deek ve McHugh tarafından 2000 yılında yapılmıştır.



Şekil 9 : SolveIt Yazılımının Ekran Görüntüsü

"Multimedia Command Centre" (Şekil 10) adı verilen bir diğer program Garner tarafından 1997 yılında Visual Basic programlama dili için öğretici olarak geliştirilmiştir.

İçerisinde programlama problemlerinin setleri bulunan ve internet üzerinden verilen çeşitli ipuçları ve filmler sayesinde öğrencilerin problemi analiz etmelerini sağlayan bu yazılım problemlerin nasıl üstesinden gelineceğini de öğretir.

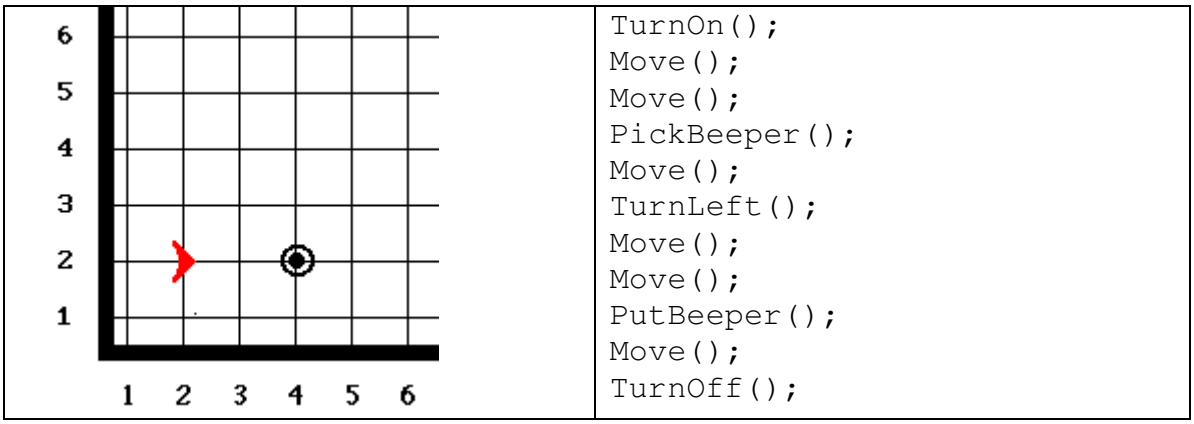
Şekil 10 : Multimedia Command Centre (VB Tutor) Yazılımının Ekran Görüntüsü

1980'lerden beri çeşitli formatları yapılmış olan Karel minidünya programlama öğreticilerinin en popüleridir. Robotlar tarafından yapılması sağlanan çeşitli problem setlerinin tanımlandığı bu yazılım, öğrenciler tarafından yeni problemlerin tanımlanmasına da olanak sağlar. Program içerisindeki dünyalarda hareket ettirilebilen caddeler, bulvarlar, duvarlar, bulunmaktadır ve robot ise bir ok şeklinde gösterilmiştir. Robot'a sağa dön, sola dön gibi çeşitli komutlar verilebilir. Avusturalya'daki bir kaç lisede Karel ile geliştirilmiş ve internet üzerinden yayınlanan çeşitli dersler ve ödevler mevcuttur. Bu sitede öğrencilerin problemleri analiz edebilmeleri için çeşitli ipuçları ve kullanışlı açıklamalar da yer almaktadır.

Adım 2 : Tasarım ve Çözüm-Algoritma Geliştirme

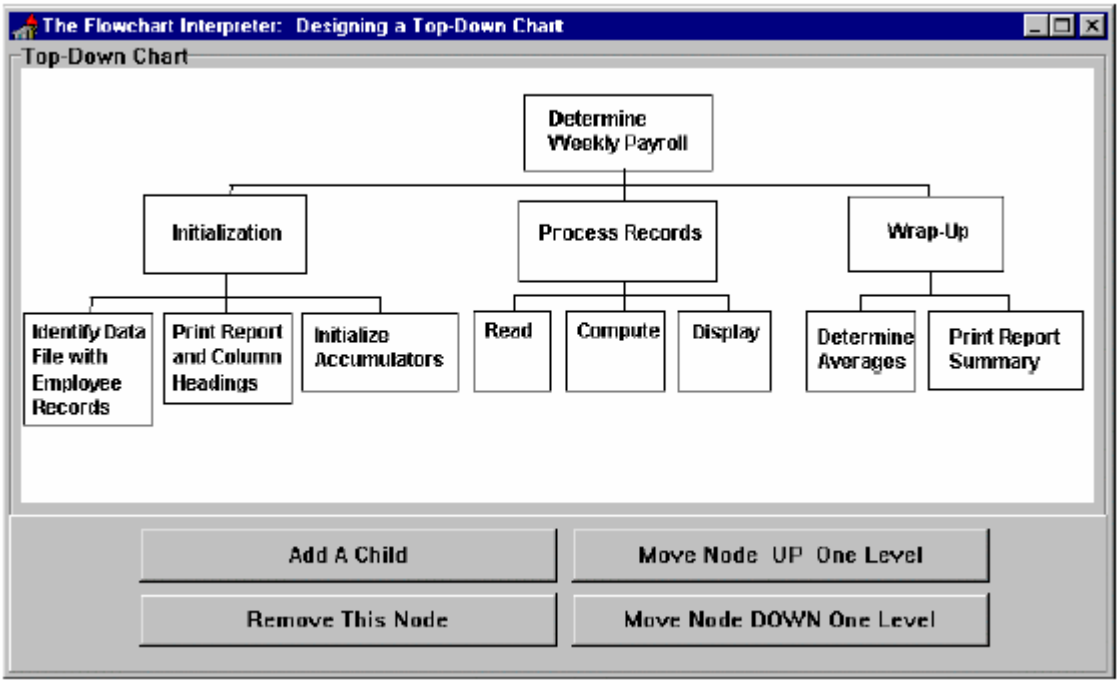
Problemi analiz ettikten sonra gerekli olan basamak çözüm üretmektir. Bu, öğrencilerin analiz ettikleri problem hakkında bazı biçimlerde çözüm üretmeleri basamağıdır. Geçmişte bu basamak için elle çizilmiş akış diyagramları ve yalancı kodlar kullanılmaktaydı.

Karel the Robot (Şekil 11) isimli program bu basamakta da yardımcı olabilecek bir programdır. Program içerisinde Robot'a verilen veya verilecek olan komutların yalancı kodlarla gösterilmesi mümkündür. (Sağa dön, Sola dön, dur gibi komutlar)



Şekil 11 : Karel The Robot Yazılımının Ekran Görünütüsü ve Bazı Komutları

Akış diagramları yardımıyla görsel algoritmalar tasarlamak için geliştirilmiş bir araç olan FLINT (Şekil 12) programlama dilinin yazımsal kurallarının detaylarından uzaklaşarak öğrencilerin resimsel arayüzler yardımıyla akış diagramları geliştirmelerini sağlayan bir araçtır.

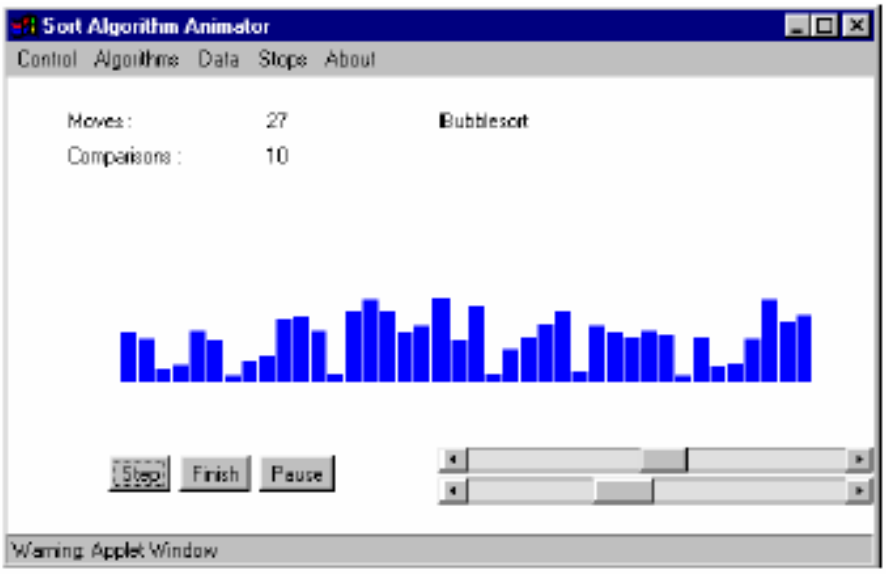


Şekil 12 : FLINT İsimli Yazılımın Ekran Görüntüsü

Temel Algoritma Animasyon Araçları

Bu algoritmalar dizi içerisinde sıralama, dizi veya dosya içerisinde eleman arama, iki dosyayı birleştirme gibi algoritmalardır. Programlama öğretmenleri genellikle var olan görselleştirme araçları yerine alışılagelmiş olan yola yani tahta ve tebeşir kullanma tekniğine başvururlar.

Bu konuda internet üzerinde çok sayıda bulunan Java Appletleri gibi bir çok araç geliştirilmiştir. Ploadereder tarafından 2000 yılında geliştirilen "The Sort Algorithm Animator V1.0" (Şekil 13) bunlara bir örnek olabilir. Animasyon çalıştırıldığında çeşitli yüksekliklere sahip barlar birbirleriyle karşılaştırılarak gerekli olduğunda yer değiştirilirler. Program otomatik olarak başlayabileceği gibi istenilen yerde algoritma animasyonun durdurulmasına ve hızının ayarlanmasına da olanak tanır. (Ploedereder, 2000)



Şekil 13 : Sort Algorithm Animator Yazılımının Ekran Görüntüsü

Adım 3: Algoritmayı Uygulamak

Yazılım geliştirmenin son adımı olarak adlandırılan bu adım, öğrencilerin bir önceki adımda geliştirdikleri algoritmayı çalışabilecek programlama koduna dönüştürme basamağıdır. Karel the Robot gibi sistemler bu basamak için de

kullanılabilir. Ayrıca FLINT de bu üçüncü basamak için çok fazla geliştirme araçlarına da sahiptir.

Görselleştirme, alana ait soyut ve karmaşık kavramları anlamadaki yararları nedeniyle uzun zamandır bilgisayar bilimleri eğitimi kullanılmaktadır. Yaklaşımların çoğu algoritma animasyonları, programın yapısı ve çalışması hakkındadır. Bu yaklaşımların yanında tasarım yaklaşımı da araştırmalarca etkili bulunmuş ve yine kullanılmaya başlanmıştır.

Vince adı verilen ve internet üzerinden çalışan araç öğrencilerin C programlarının çalışmasını anlamalarına yardımcı olmak üzere Rowe ve Thorburn (2000) tarafından geliştirilmiştir. Bu araç, C programlarının çalışmasını adım adım gösterir ve bilgisayar belleğinin içeriğini her adımda kullanıcılara açıklayarak görsel hale getirir. Öğretmen tarafından tasarlanan örnek programların, öğrenciler tarafından kendi isteklerine göre değiştirilmesi ve programın nasıl çalıştığının gösterimi özellikleri de bulunur. Yapılan araştırma göstermiştir ki, Vince, programcılığa giriş kurslarında öğrencilere programcılığı öğrenmeleri konusunda olumlu olarak etki etmiştir.

3. Yöntem

3.1. Araştırma Deseni

Araştırmada alt problemlerde ifade edilen bulgulara ulaşabilmek için deney ve kontrol grupları oluşturulmuş, öntest-uygulama-sontest deseni kullanılmıştır. Testlerde Ek 1'de verilen ve araştırmacı tarafından geliştirilen başarı testi öğrencilere uygulanmıştır. Daha sonra (20 gün arayla) aynı test öğrenme ürünlerinin kalıcılığını belirlemek amacıyla her iki gruba da sunulmuştur. Testler üzerinde istatistiksel analizler yapılarak gruplar arasındaki farklılıklar araştırılmıştır.

3.2. Çalışma Grubu

Bu çalışmanın deneysel tasarımına uygun olması ve çalışmanın özel bir öğrenme alanına yönelik olması nedeniyle örnekleme yöntemi ile verilere ulaşma yerine özel çalışma grubu üzerinde uygulama yapılmıştır.

Çalışmanın deneysel yapısı gereği çalışma grubunun daha önceden akademik açıdan herhangi bir programlama dili almamış olması gereklidir. Bundan dolayı "çalışma grubu" Hacettepe Üniversitesi Bilgisayar ve Öğretim Teknolojileri Eğitimi Bölümü 1. sınıf öğrencilerinden oluşturulmuştur. Ancak ortaöğretimde programlama dillerine yönelik ders gören öğrencilerin varlığı nedeniyle bu öğrencilerin çalışma grubundan çıkartılması yerine önceden öğrenme yaşantısına sahip olma durumları alt problemlere eklenerek çalışmanın zenginleştirilmesi amaçlanmıştır.

Çalışma grubu Hacettepe Üniversitesi Bilgisayar ve Öğretim Teknolojileri Eğitimi Bölümü 1. sınıf öğrencileri (Uygulamaya katılan 34 öğrenci) olarak seçilmiştir.

Hacettepe Üniversitesi Bilgisayar ve Öğretim Teknolojileri Eğitimi Bölümü 1. sınıf öğrencileri 18 kişi kontrol ve 16 kişi deney grubu olmak üzere rastgele dağıtılmışlardır.

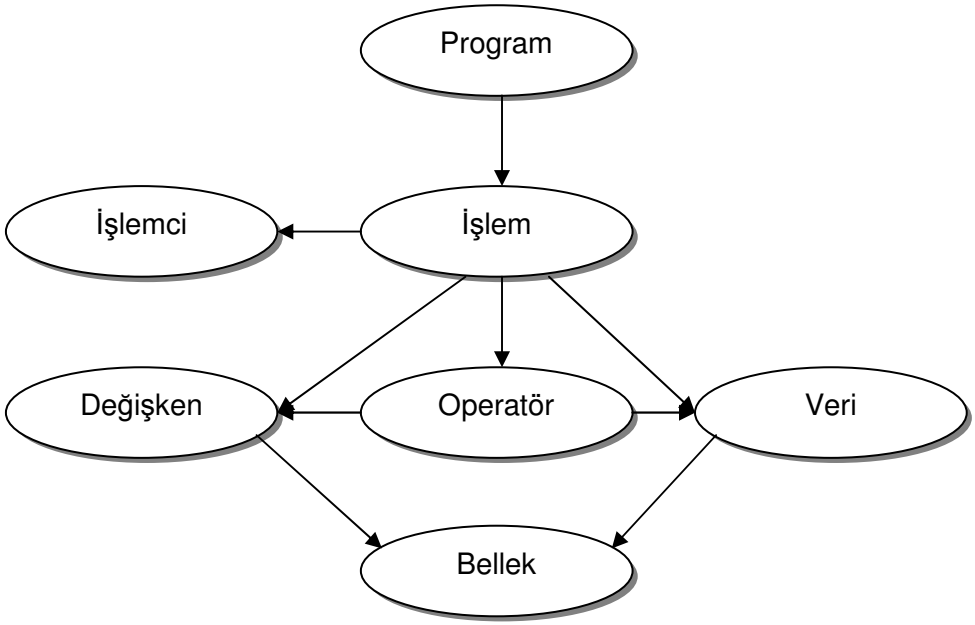
3.3. Çoklumortama Dayalı Eğitim Yazılımı ve Geliştirilme Süreci

Bu araştırmada öğrencilerin yazılımı kullanarak kendi başına daha önceki bilgilerine de dayanarak programlama bilgisi alması ve bu bilgileri çeşitli testlerle deneyerek anlamlı hale getirilmesi amaçlanmıştır. Bunun için araştırmacı tarafından her programlama dilinde bulunan temel kavramlar bir problem durumu ile temellendirilmiş olup, çokluortam ile desteklenerek kullanıcıya sunulabilir hale getirilmiştir. Genellikle programlama dillerindeki bu temel kavramlar soyut oldukları için anoloji yöntemi ile kavramlar nesnelleştirilmiştir. Programın belirli bir problem durumuna dayandırılması ve bu problem durumunun yazılımın her noktasında ve her programlama teriminin anlatımında sürekli devam etmesi, güncel hayattan bir problemi çözmesi, programlama mantığının daha da iyi anlaşılmasına olanak tanımaktadır.

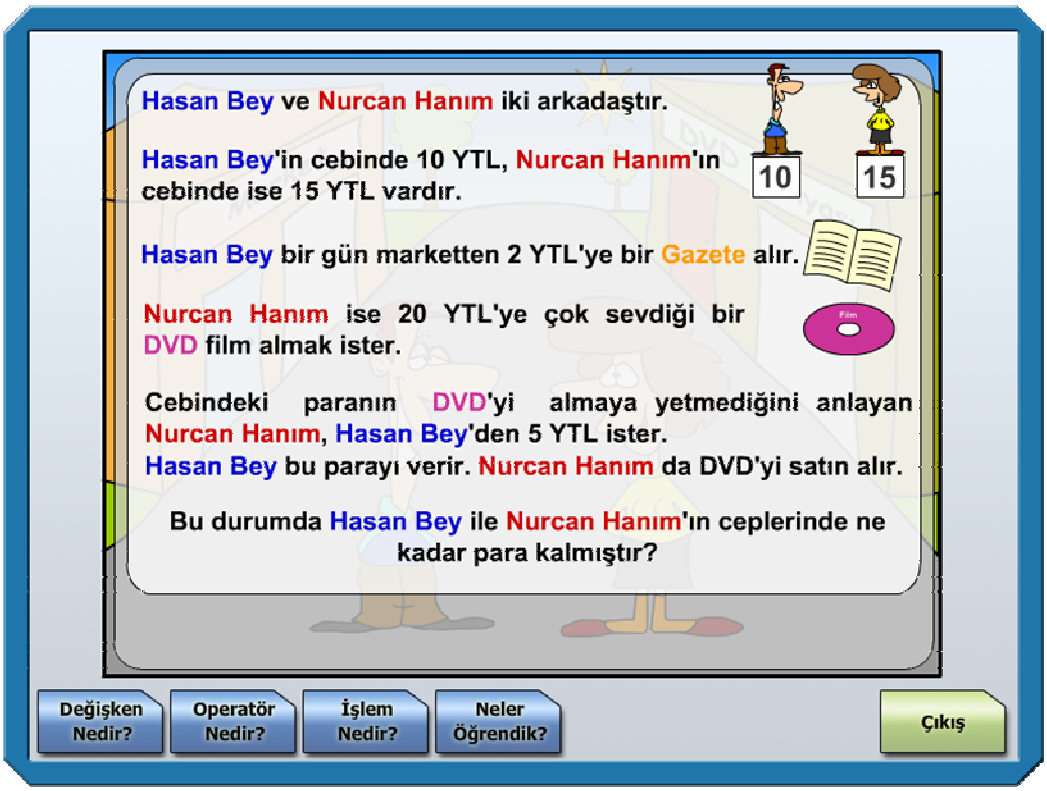
Bir diğer amaç ise daha önceden programlama dersi almış olan bireylerin kavram yanılgısına düşmesini önlemek ve programcılıktaki temel kavramların aslında nasıl düşünülmesi gerektiğini göstermektir. Hem ilk defa programlama dersi alan, hem de daha önceden programcılıkla uğraşmış, bu konuda formal ya da informal bir eğitim almış bireylerin aslında programlama yaparken bilgisayarın bu işi nasıl yaptığını nesnelleştiren ve verilen problem dumundaki problemin çözümüne yönelik yapılan işlemleri görselleştiren bu yazılım, bireylerin işlem adımlarını da kolayca takip edebilecekleri alanlar içermektedir.

ÇDEY, bildirimsel bilgilere başlamadan önce tüm bu kavramların ortak bir problem durumu üzerinde anlatılabilmesi için ilk olarak öğrencilere günlük yaşamda her zaman karşılaşılabilecek bir olayı “Problem Durumu” olarak aktarır (Şekil 15). Daha sonra bildirimsel bilgilerden biri olan “Değişken Nedir?” bölümü ile devam eder (Şekil 16). “Değişken”in bilgisayar belleğinde tutulduğu bilgisi ve “Bellek” ile insan hafızasının benzerliği ilerleyen adımlarda sunulmaktadır (Şekil 17). Daha sonra, değişkenlerin bellek içerisinde tuttukları değerler olan “Veri”ler (Şekil 18), veriler arasında işlemler yapabilmek için kullanılan semboller olan “Operatör”ler (Şekil 19) ve bu simgeler sayesinde kendi başına çalışabilen “İşlem(Komut)” (Şekil 20) adı verilen yapılar da program içerisinde aktarılır. İşlem

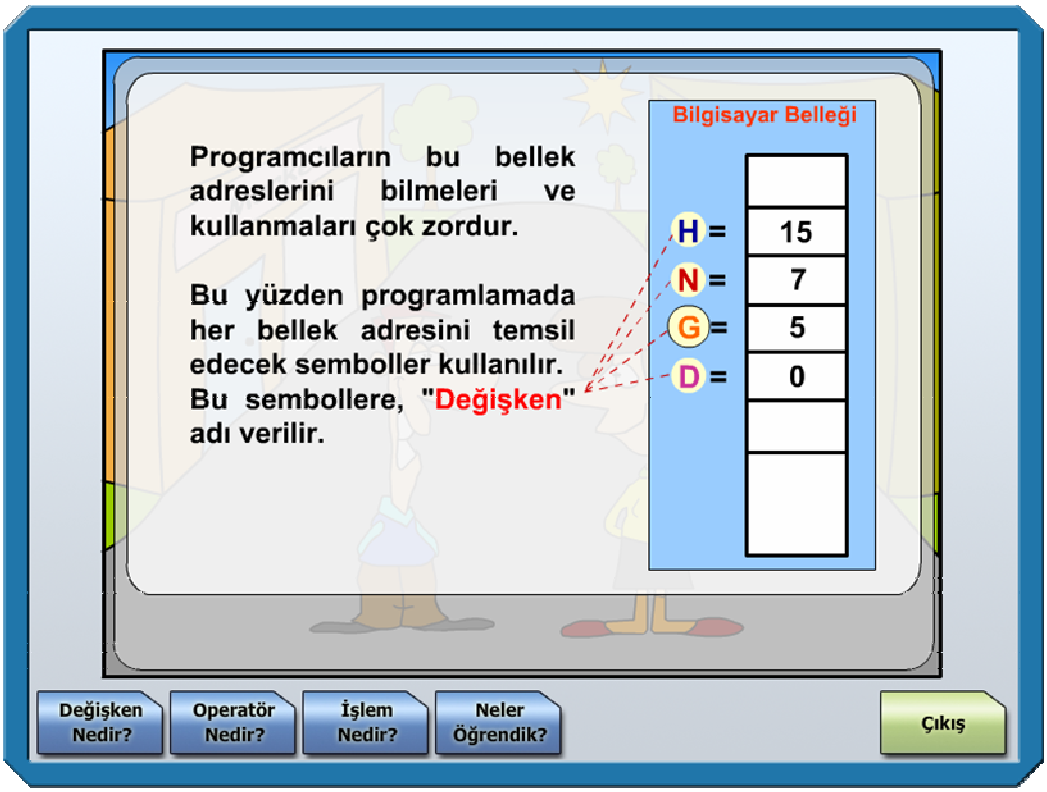
kavramı verilirken Operatör, Değişken ve Veri kavramları arasındaki ilişki görsel olarak verilir (Şekil 21). İşlemlerin "İşlemci" (Şekil 22) adı verilen birimde yapıldığı anlatılır. İşlemlerin kurallı bir bütün oluşturmasıyla "Program"ın (Şekil 23) oluştuğu aktarılır. Şekil 24'te ise anlatılan tüm kavramların ilişkileri sunulmaktadır. Şekil 14, ÇDEY'nın içerdiği "Bildirimsel" bilgileri ve bu bilgilerin bir biriyle olan ilişkilerini gösterir.



Şekil 14 : ÇDEY'nın İçerdiği "Bildirimsel" Bilgiler



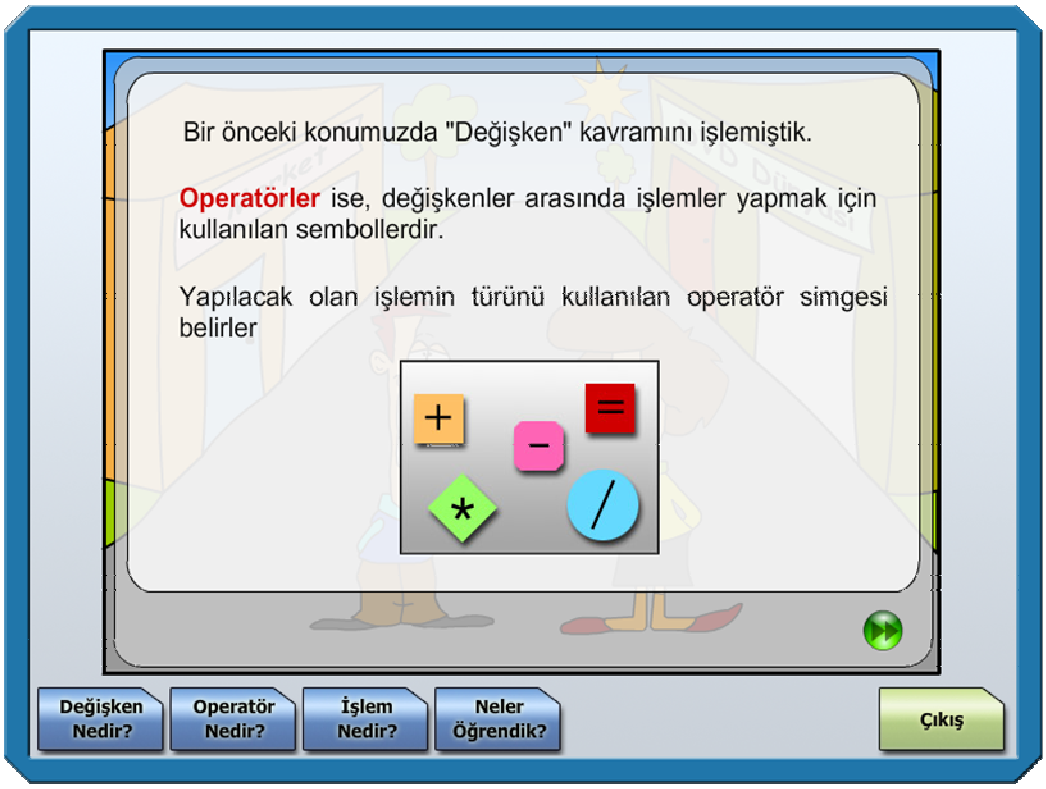
Şekil 15 : ÇDEY'in İçerdiği "Öykü Anlatımı"



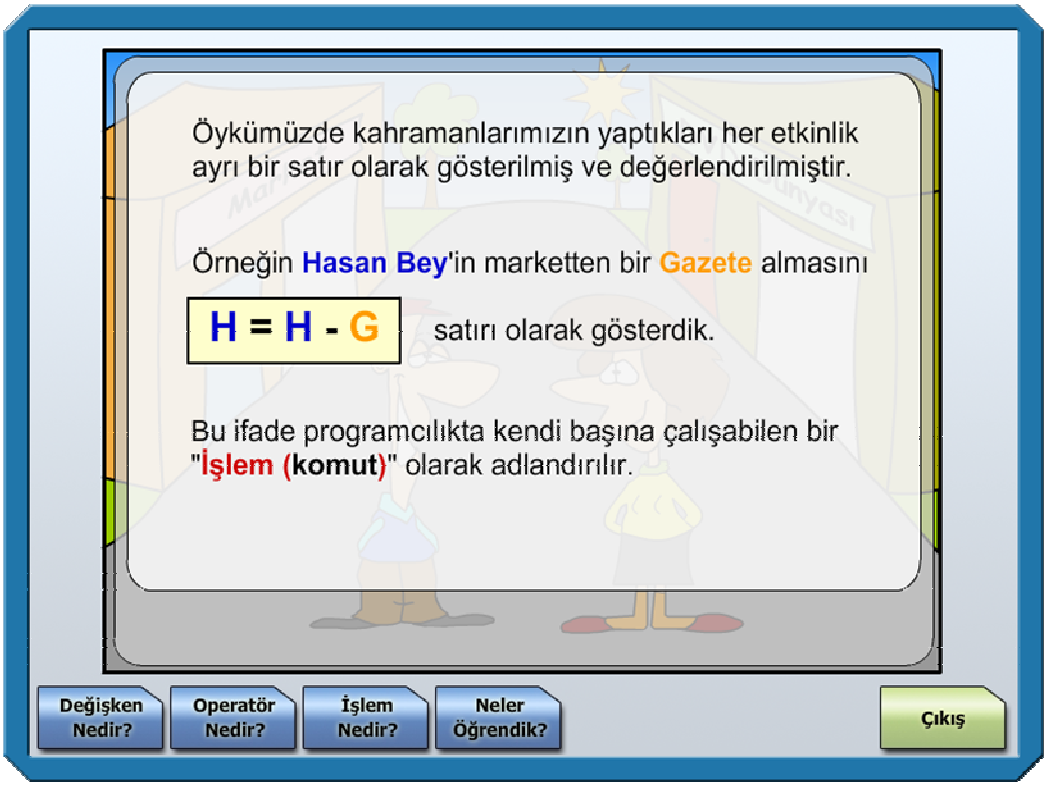
Şekil 16 : ÇDEY İçindeki "Değişken Nedir?" Ekran Görüntüsü

Şekil 17 : ÇDEY İçindeki İnsan Zihni ile Bellek Arasındaki Metaforun Verildiği Ekran Görüntüsü

Şekil 18 : ÇDEY İçindeki "Veri Nedir?" Ekran Görüntüsü

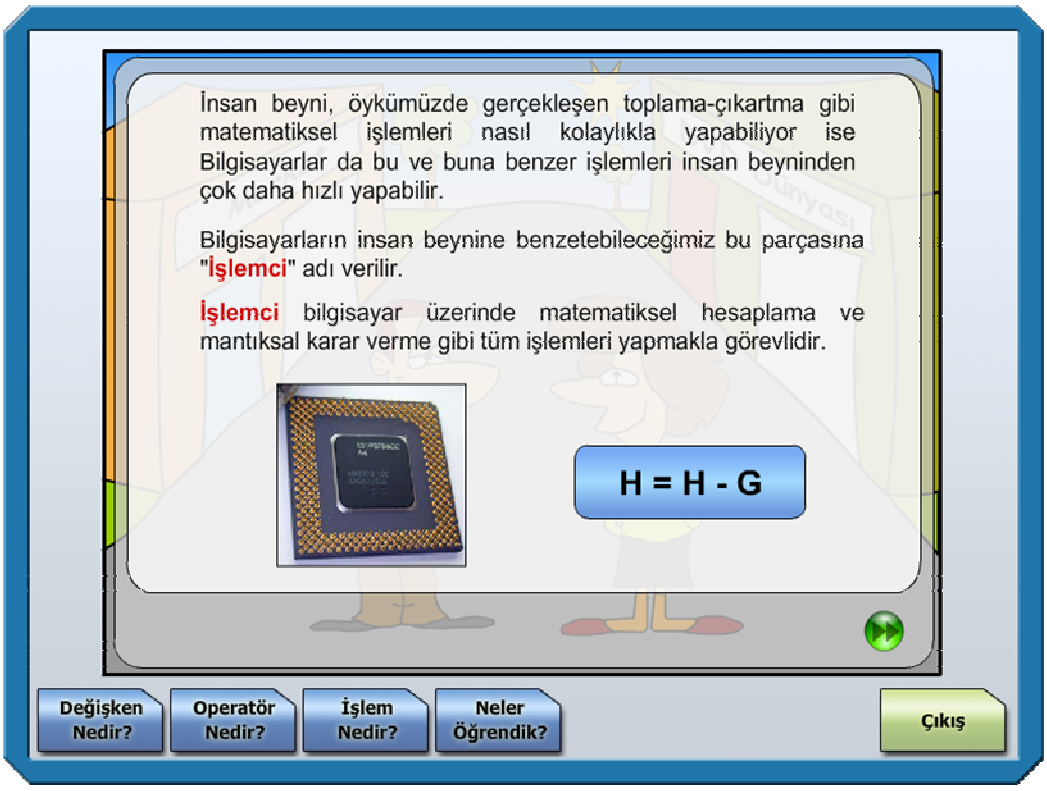


Şekil 19 : ÇDEY İçindeki "Operatör Nedir?" Ekran Görüntüsü

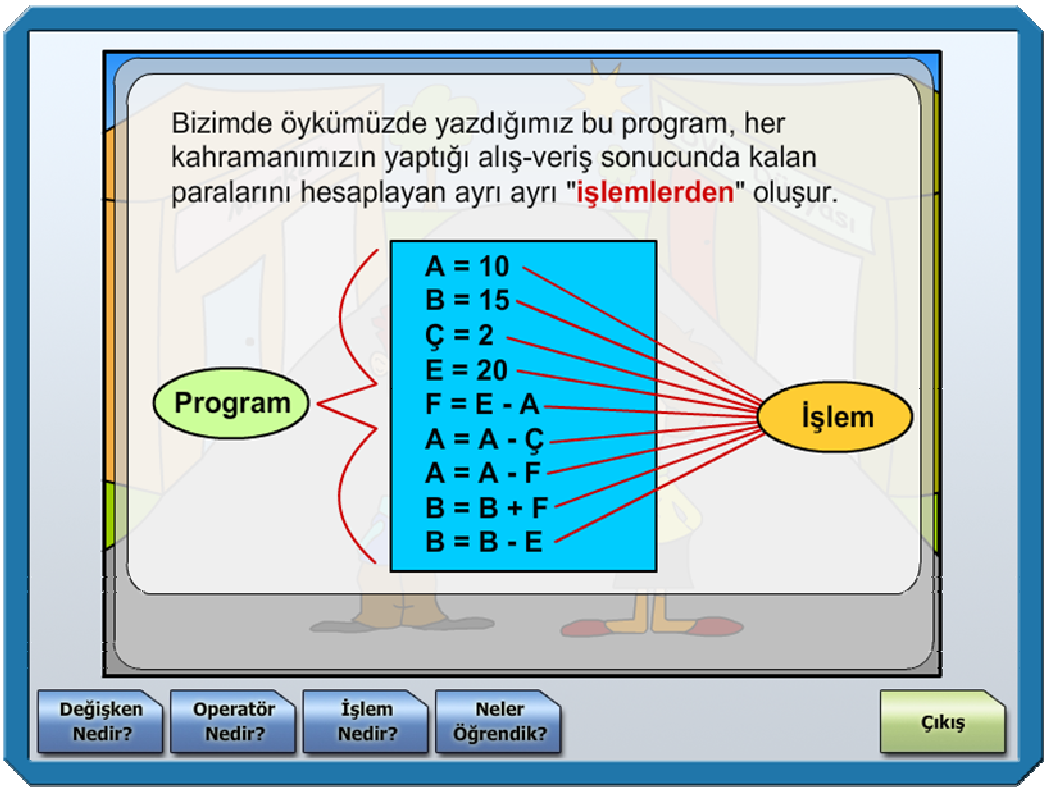


Şekil 20 : ÇDEY İçindeki "İşlem Nedir?" Ekran Görüntüsü

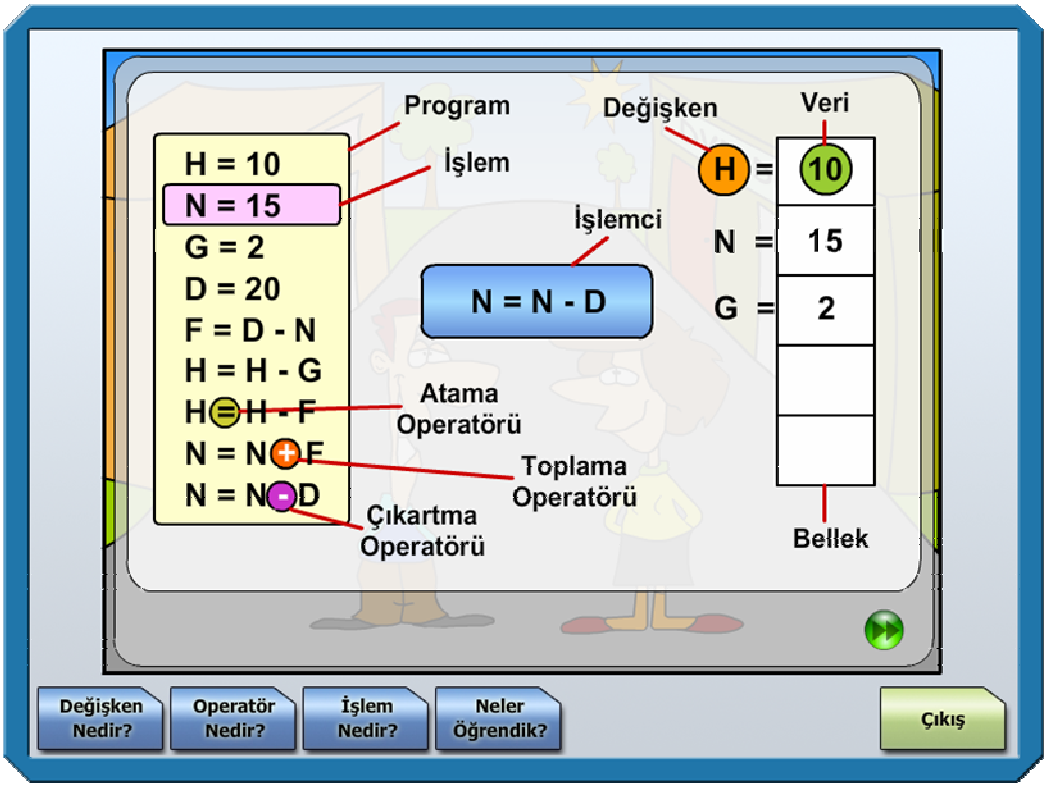
Şekil 21 : ÇDEY İçindeki "Operatör", "Değişken" ve "Veri" Kavramları Arasındaki İlişkiyi Gösteren Ekran Görüntüsü



Şekil 22 : ÇDEY İçindeki "İşlemci Nedir?" Ekran Görüntüsü



Şekil 23 : ÇDEY İçindeki "Program Nedir?" Ekran Görüntüsü



Şekil 24 : ÇDEY İçindeki "Neler Öğrendik?" Ekran Görüntüsü

3.4. Araştırmada izlenen Analiz Süreci ve Yöntemleri

Kontrol (18 öğrenci) ve deney grupları (16 öğrenci) rastgele seçilerek oluşturulmuş ve her iki gruba da öğretim sürecinde ele alınacak kazanımlara yönelik başarı testi uygulanmıştır (öntest). Bu test (Ek 1) temel programlama kavramlarını ve bu kavramların birbirleriyle olan ilişkilerini sorgulayan on adet açık uçlu soru içermektedir. Bu öntest sonrasında kontrol grubuna araştırmacı tarafından alışılagelmiş yöntem kullanılarak programlamadaki temel kavramlar ve bu kavramların ilişkileri anlatılmıştır. Deney grubuna ise yine araştırmacı tarafından hazırlanmış "Çokluortama Dayalı Eğitim Yazılımı" ile bir öğretim süreci uygulanmıştır. Öğrencilere uygulama sonunda yine başarı testi uygulanmış ve programlama becerisi için gerekli olan temel kavramların öğrenciler tarafından öğrenilip öğrenilmediği kontrol edilmiştir. Elde edilen başarı puanları arasındaki farklılık öğretim yöntemleri açısından varyans analizi ile araştırılmıştır. Aynı şekilde başarı puanları arasındaki farklılıklar, öğrencilerin cinsiyeti ve önceden alınan eğitim durumlarına göre varyans analizi yöntemleri ile analiz edilmiştir.

Kalıcılığı ölçmek amacıyla kontrol ve deney grubuna, üç haftalık bir aradan sonra bir başarı testi daha uygulanmıştır. Buna göre kazanılan öğrenme düzeyleri (sontest başarı puanları) ile kalıcılık testinden almış oldukları puanlar öğretim yöntemleri açısından karşılaştırılmıştır. Bu karşılaştırmada bağımlı örneklem t testi (paired samples t test) kullanılmıştır.

Bu çalışmada alt problemlere ilişkin hipotezler oluşturulmuş ve bu hipotezler parametrik ve parametrik olmayan test istatistikleri ile sınanmıştır. Her iki test istatistiklerinin aynı sonucu vermeleri nedeniyle, çalışmada yalnızca parametrik test istatistiğine dayalı bulgular rapor edilmiştir. Diğer taraftan hipotezlerin sınanmasında yanılma düzeyi (I. tür hata) 0,05 olarak belirlenmiştir.

4. Bulgular ve Yorumlar

4.1. Alt Problem 1

Araştırmanın birinci alt problemi "Bildirimsel konuda hazırlanmış ÇDEY ile alışılagelmiş öğretim yöntemleri arasında öğrencilerin başarıları yönünden farklılığın araştırılması" şeklinde ifade edilmiştir.

Deney ve kontrol grubunda yer alan öğrencilerin öntest ve sontest başarı puanları arasındaki farklılık varyans analizi yöntemi ile karşılaştırılmıştır. Çizelge 2'de bu puanlara ilişkin betimsel bulgular, Çizelge 3'de ise varyans analiz çizelgesi verilmiştir.

Çizelge 2 : Deney ve kontrol gruplarındaki öğrencilerin sonteste ilişkin betimsel bulguları

	Gruplar	Ortalama	Std. Sapma	N
Öntest	Kontrol	2,61	3,05	18
	Deney	2,62	2,33	16
	Toplam	5,23	5,38	34
Sontest	Kontrol	13,50	3,09	18
	Deney	14,75	3,53	16
	Toplam	28,25	6,62	34

Çizelge 2 incelendiğinde her iki grubun (deney-kontrol) öntest başarı puanları ortalamasının (2,62) ve sontest başarı puanları ortalaması (14,75) arasında oldukça büyük bir farklılık görülmektedir. Bu farklılık her iki öğretim sürecinin etkin bir şekilde gerçekleştiğini göstermektedir. Yine Çizelge 2'ye göre deney ve kontrol gruplarındaki öğrencilerin öntest başarı puanlarının birbirine çok yakın olduğu görülmektedir. Kontrol grubunun öntest puanı 2,61 ve deney grubunun ise öntest başarı puanı 2,62 olarak elde edilmiştir. Ancak öğretim süreci sonunda gerçekleştirilen sontest başarı puanları arasındaki farklılığın ise deney grubundaki öğrenciler lehine arttığı görülmektedir. Bu farklılığın istatistiksel olarak anlamlı olup olmadığı ise varyans analizine dayalı F testi ile test edilmiş ve sonuçlar Çizelge 3'de verilmiştir.

Çizelge 3 : Deney ve kontrol gruplarındaki öğrencilerin başarı puanlarına ilişkin bulgular

	Değişim Kaynağı	Ser. Derecesi	Kareler Ortalaması	F	P
Öntest	Gruplar	1	0,02	0,00	0,99
	Hata	32	7,50		
Sontest	Gruplar	1	19,23	4,35	0,05
	Hata	32	4,42		

Çizelge 3'e göre, deney ve kontrol gruplarında yer alan öğrencilerin öntest başarı puanlarına göre 0,05 yanılma düzeyinde anlamlı bir farklılık (P>0,00) elde edilememiştir. Bu durum ise deneysel düzeneğin homojen olduğunun bir göstergesi olarak ifade edilebilir. Deney ve kontrol gruplarındaki öğrencilerin başarı puanları arasındaki farklılık 0,05 yanılma düzeyinde istatistiksel olarak anlamlı olduğu (P≤0,05) görülmüştür. Buna göre; ÇDEY öğretim etkinliği ve alışılagelmiş öğretim ile yapılan öğrenme etkinliklerine ilişkin sontest başarı puanları arasındaki farklılık ÇDEY ile yapılan etkinlik lehinde elde edilmiştir.

4.2. Alt Problem 2

Araştırmanın ikinci alt problemi "Bildirimsel konuda hazırlanmış yazılımın programlama başarısındaki etkisi cinsiyete göre farklılığının araştırılması" şeklinde belirtilmiştir.

Tüm deneysel çalışmaya katılan öğrencilerin cinsiyetlerine ilişkin betimsel bulgular Çizelge 4'te verilmiştir.

Çizelge 4 : Öğrencilerin cinsiyetlerine göre başarı puanlarına ilişkin betimsel bulgular

	Gruplar	Cinsiyet	Ortalama	Std. Sapma	N
Öntest	Kontrol	Erkek	1,63	1,69	8
		Kız	3,4	3,72	10
		Toplam	5,03	5,41	18
	Deney	Erkek	2,7	2,67	10
		Kız	2,5	1,87	6
		Toplam	5,2	4,54	16
Sontest	Kontrol	Erkek	12,5	3,34	8
		Kız	14,3	2,79	10
		Toplam	26,8	6,13	18
	Deney	Erkek	13,7	3,53	10
		Kız	16,5	3,02	6
		Toplam	30,2	6,55	16

Çizelge 4'e göre kız öğrencilerin öntest başarı puanları ortalaması (2,50) erkek öğrencilerin başarı ortalamasına (2,70) göre daha düşük iken uygulama sonrasında bu oran değişmiş ve kız öğrencilerin ortalaması (16,50) erkek öğrencilerin başarı ortalamasından (13,70) daha yüksek çıkmıştır. Kontrol grubundaki kız öğrencilerin hem öntest başarıları hem de sontest başarılarının erkek öğrencilerden daha yüksek olduğu gözlenmiştir.

Kız ve erkek öğrencilerin öntest ve sontest başarı puanları arasındaki farklılığın istatistiksel olarak anlamlılığı için iki yönlü varyans analizi yapılmış ve elde edilen sonuçlar Çizelge 5'te verilmiştir.

Çizelge 5'e göre öntest puanlarına göre kız ve erkek öğrencilerin başarı puanları arasında istatistiksel olarak anlamlı bir farklılık gözlenmez iken ($P=0,42>0,05$), sontest başarı puanlarına göre kız öğrenciler lehine anlamlı bir farklılık çıkmaktadır ($P=0,04<0,05$).

Çizelge 5 : Deney ve kontrol gruplarındaki öğrencilerin cinsiyetlerine göre başarı puanlarına ilişkin bulgular

Değişim Kaynağı	Değişken	Ser. Derecesi	Kareler Ortalaması	F	Sig.
Cinsiyet (C)	Sontest	1	48,17	5,24	0,03
	On Test	1	5,04	0,67	0,42
Gruplar (G)	Sontest	1	40,58	4,41	0,04
	On Test	1	0,06	0,01	0,93
C * G	Sontest	1	3,26	0,36	0,56
	On Test	1	7,93	1,05	0,31
Hata	Sontest	30	9,19		
	On Test	30	7,53		
Toplam	Sontest	34			
	On Test	34			

Çizelge 5'e göre başarı puanları üzerinde cinsiyet ve (deney-kontrol) grupların birlikte (etkileşimli) etkisi anlamlı bulunmamıştır.

4.3. Alt Problem 3

Araştırmanın üçüncü alt problemi "Bildirimsel konuda hazırlanmış ÇDEY'nın öğrencilerin programlama başarısındaki etkisinin öğrencilerin ön öğrenmelerine göre farklılığının araştırılması" şeklinde ifade edilmiştir.

BÖTE öğrencilerinin bir kısmı önceki öğrenme yaşantılarında bilgisayar programlama dersleri gördükleri için ÇDEY'nın öğrencilerin başarı puanları açısından farklılıkları araştırılmıştır.

Çizelge 6 : Öğrencilerin önceki öğrenme yaşantılarına göre başarı puanlarına ilişkin betimsel bulgular

	Gruplar	Prog. Dersi Alma Durumu	Ortalama	Std. Sapma	N
On Test	Kontrol	Evet	3,91	3,27	11
		Hayır	0,57	0,79	7
		Toplam	4,48	4,06	18
	Deney	Evet	4,14	2,54	7
		Hayır	1,44	1,33	9
		Toplam	5,58	3,87	16
Sontest	Kontrol	Evet	13,36	3,26	11
		Hayır	13,71	3,04	7
		Toplam	27,07	6,3	18
	Deney	Evet	14	3,11	7
		Hayır	15,33	3,91	9
		Toplam	29,33	7,02	16

Çizelge 6'ya göre daha önceden bilgisayar programlama dersi alan öğrencilerin öntest başarı puanları ortalaması (kontrol grubu=3,91, deney=4,14), ilk defa programlama dersi alan öğrencilerin başarı puanları ortalaması göre (kontrol grubu=0,57, deney=1,44) çok yüksek elde edilmiştir. Ancak etkinlik sonrasında bu oranlar ters yönde gözlenmiştir. Sontest başarı puanlarına göre daha önceden bilgisayar programlama dersi alan öğrencilerin başarı puanları ortalaması (kontrol grubu=13,36, deney=14,00), ilk defa programlama dersi alan öğrencilerin başarı puanları ortalamasına göre (kontrol grubu=13,71, deney=15,33) daha düşük gözlenmiştir.

Çizelge 7 : Deney ve kontrol gruplarındaki öğrencilerin önceki öğrenme yaşantılarına göre başarı puanlarına ilişkin bulgular

Değişim Kaynağı	Değişken	Ser. Derecesi	Kareler Ortalaması	F	Sig.
Ön Yaşantılar	SonTest	1	12,789	1,268	,269
	OnTest	1	75,487	14,222	,001
Hata	SonTest	31	10,087		
	OnTest	31	5,308		
Toplam	SonTest	34			
	OnTest	34			

Çizelge 7'ye göre önceden programlama dersi alan öğrencilerin öntest başarı puanları, ilk defa programlama dersi alan öğrencilere göre daha yüksek olduğu (Çizelge 6) ve bu farklılığın istatistiksel olarak anlamlı olduğu (P=0,001<0,005) görülmektedir. Ancak sontest başarı puanları ortalamasına göre ise her iki gruptaki öğrencilerin arasında anlamlı farklılık bulunamamıştır. Fakat ilk defa bilgisayar programlama etkinliği yaşayan öğrencilerin sontest başarı uygulamasının daha yüksek olduğu Çizelge 6'e göre ifade edilebilir.

4.4. Alt Problem 4

Araştırmanın dördüncü alt problemi "Bildirimsel (declarative) konuda hazırlanmış ÇDEY ile alışılagelmiş öğretim yöntemi arasında kazanılan bilginin kalıcılığı yönünde farklılığın araştırılması" şeklinde ifade edilmiştir.

Bu bulguya ulaşabilmek için deney ve kontrol grubunda yer alan öğrencilerin sontest başarı puanları ile kalıcılık testinden aldıkları puanlar arasındaki

deney grubu→ [sontest-kalıcılık testi]

kontrol grubu→ [sontest-kalıcılık testi]

farklılıkları araştırılmıştır. Araştırmalarda bağımlı örneklem t testi (paired samples t test) kullanılmıştır.

Çizelge 8 : Sontest-Kalıcılık Testi Başarı Puanları Karşılaştırması

	Ortalamalar		**Bağımlı Örneklem Testi**		
	Sontest	**Kalıcılık Testi**	**t**	**P**	**Etki Genişliği**
Kontrol Grubu	13,50	8,28	5,13	0,00	0,63
	3,09	3,40			
Deney Grubu	14,75	12,50	2,80	0,02*	0,32
	3,53	3,22			

* Yanılma düzeyi 0,05'e göre istatistiksel fark vardır.
Yanılma düzeyi 0,01'e göre istatistiksel fark yoktur.

Çizelge 8'de öğrencilerin sontest ve kalıcılık testinden aldıkları başarı puanları ortalamaları bağımlı örneklem t testi ile karşılaştırılmıştır. Deney ve kontrol grubundaki kalıcılık testi puanlarında sonteste göre bir azalma söz konusudur. Ancak deney grubunda yer alan öğrencilerin sontest – kalıcılık testi başarı puanları

arasındaki farklılık, kontrol grubuna göre daha düşük çıkmıştır. Etki genişliklerine göre başarı puanlarındaki bu düşüşün kontrol grubunda daha etkin olduğu gözlenmiştir.

5. Sonuç, Tartışma ve Öneriler

5.1. Sonuç ve Tartışma

Bu çalışmanın amacı çokluortama dayalı eğitim yazılımının programlama dilleri öğretiminde öğrenci başarıları üzerine etkisini araştırmak olarak belirlenmiştir. Bu "etki araştırması" belirtilen eğitim yazılımı ve alışılagelmiş öğretim yöntemi ile öğretim sürecinde yer alan öğrenciler üzerinde gerçekleştirilmiştir.

1) ÇDEY ile gerçekleştirilen öğretim sürecinin alışılagelmiş yöntem ile gerçekleştirilen öğretim sürecinden daha etkin olduğu gözlenmiştir. Elde edilen bu sonuç, ilgili konuda yapılan diğer araştırmaların bulguları ile tutarlıdır.

Yapılan araştırmalarda genellikle, BDÖ, öğrenci başarısı, öğrenme süresi, öğrenmenin kalıcılığı ve öğrencinin öğrenme konusuna yönelik tutumları açısından alışılagelmiş öğretim yöntemleri ile karşılaştırılmıştır. Bunlardan Vinsonhaler ve Bass (1972) 10 bağımsız araştırmayı inceleyerek, bir konu ile ilgili alıştırmaları bilgisayarda yapan ilkokul öğrencilerinin yapmayanlara göre 1 ile 8 ay arasında daha ileride oldukları sonucuna varmışlardır. Kulik, Bangert ve Williams (1983), ortaöğretimde bilgisayar uygulamaları ile ilgili birbirinden bağımsız 51 araştırmayı meta-analiz tekniğine göre birleştirerek, bilgisayara dayalı öğretimin beş değişken üzerindeki etkisini incelemişlerdir. Buna göre:

İncelenen araştırmalarda, BDÖ ile öğrenen öğrencilerin dönem sonu sınav notları, alışılagelmiş yöntemle öğrenen öğrencilerin dönem sonu sınav notlarından 0,32 standart sapma daha yüksektir. Bir başka deyişle, alışılagelmiş yöntemlerle öğrenenlerin dönem sonu not ortalaması 50 iken, BDÖ ile öğrenen öğrencilerin dönem sonu not ortalaması 63 olabilmektedir (Aşkar,1990).

Bu açıdan bakıldığında ÇDEY ile gerçekleştirilen öğretim süreci, alışılagelmiş yöntem ile gerçekleştirilen öğretim sürecine göre, öğrencilerin başarı puanlarının artması yönünde olumlu etki göstermiştir.

2) ÇDEY ile öğretim süreci kız öğrenciler üzerinde daha etkili bir öğretim gerçekleştirmiştir.

Bu konuya ilişkin yapılan çalışmalarda programlama dillerindeki kız ve erkek öğrencilerin başarıları arasında anlamlı bir fark çıkmadığı (Grant, 2003, McKenna, 2004), ancak bazı kaynaklarda ise programlama dillerindeki performans bakımından kız öğrencilerin daha başarılı olduğu ifade edilmiştir (Carter & Jenkins, 1999).

Programlama kavramlarını öğrenme konusunda ise Murhpy ve diğerleri (2006) kız ve erkek öğrenciler arasında fark olmadığını rapor etmiştir.

Bu çalışmada ise elde edilen bulgular ışığında kız öğrencilerin hem alışılagelmiş öğretim hem de ÇDEY ile gerçekleştirilen öğretim sürecinde daha başarılı oldukları bulgusuna ulaşılmıştır. Bu durum Carter ve Jenkins (1999)'in çalışması ile tutarlı bulunmuştur.

3) ÇDEY ile hazırlanan öğretim ortamında yer alan ve ilk defa bilgisayar programlama dersi alan öğrenciler ile daha önceden programlama dersi alan öğrencilerin başarı puanları arasında fark gözlenmemiştir.

4) ÇDEY ile gerçekleştirilen öğrenmeler, alışılagelmiş öğretim ortamında gerçekleştirilen öğrenmelere göre daha kalıcıdır.

Ele alınan çalışmalarda alışılagelmiş yöntem ile BDÖ arasında hatırlama açısından anlamlı bir fark olup olmadığı incelenmiş ve BDÖ uygulanan öğrencilerin 2 ile 6 ay sonra uygulanan sınav sonuçlarında daha başarılı oldukları görülmüştür (Aşkar, 1990).

Araştırmada da bulunan bu sonuç literatürde bulunan bazı çalışmalarla benzerlik göstermektedir. Martens, Valcke ve Portier (1997) metin ağırlık bir materyal ile etkileşimli çokluortam destekli bir öğretim materyalinin öğrenci başarısı ve bilginin kalıcığı üzerine etkisini araştırmışlardır. Elde edilen bulgular,

etkileşimli çokluortam destekli öğretim materyalinin metin tabanlı öğretim materyaline göre öğrenci başarısı üzerinde daha etkili olduğunu göstermiştir.

Angelides ve Agius(2002)'un yaptığı bir başka bir araştırmada ise, metin+ses+video içeren çokluortam tabanlı öğrenme ortamlarının öğrenci başarısına olumlu yönde etki ettiği saptamıştır.

Bir başka araştırma olan Tsou,Wang ve Tzeng(2004)'in çalışmasında yabancı dil öğretiminde çokluortam destekli öğrenme ortamlarının, öğretim sürecinde kullanılmasının öğrenci başarısını arttırıcı bir etken olduğu sonucuna ulaşılmıştır.

ÇDEY ile gerçekleştirilen öğrenmeler, yukarıdaki araştırmalarda verilen değerlerle uyuşmuş ve alışılagelmiş eğitim yöntemiyle gerçekleştirilen öğrenmelerden daha kalıcı olduğu tespit edilmiştir.

5.2. Öneriler

- Bu çalışma içerisinde programlama dilleri öğretiminde geliştirilen yazılım, çalışmanın sınırlılığı nedeniyle, öğrenmenin zihinsel süreçleri olarak yalnızca "bildirimsel bilgi" düzeyinde hazırlanmıştır. Aynı şekilde geliştirilen yazılım, "işlemsel bilgi"yi de kapsayacak şekilde geliştirilebilir.

- Bu çalışmada programlama dilleri öğretiminde geliştirilen yazılım, çalışmanın sınırlılığı nedeniyle programlama öğretimi aşamalarından yalnızca "kavramsal bilgi" düzeyinde hazırlanmıştır. Aynı şekilde geliştirilen yazılım "yazımsal bilgi" ve "problem çözme bilgisini" de kapsayacak şekilde geliştirilebilir.

- Araştırmada, geliştirilen eğitim yazılımının başarı üzerine etkisinin öğrencilerin tutum, motivasyon, öz-yeterlik, öz-düzenleme gibi özellikleriyle olan ilişkisi irdelenebilir.

- Araştırmada, geliştirilen eğitim yazılımının başarı üzerine etkisinin farklı yaş ve eğitim düzeylerindeki farklılığı araştırılabilir.

- Geliştirilen eğitim yazılımında kullanılan problem durumu unsurlarına karşı öğrenci tutumları araştırılabilir.

Kaynakça

Akkoyunlu, B. (1994), Bilgisayarların müfredattaki yeri ve öğretmen Eğitimi. I. Eğitim Bilimleri Kongresi, 28 - 30 Nisan. Balcalı - Adana. Cilt 1. 415 - 420.

Angelides, M. C. & Agius, H. V. (2002), An Interactive multimedia learning environment for VLSI built with COSMOS. Computer & Education, 39(2),145-160.

Aşkar, P. (1990), Okullarda Bilgisayar Destekli Öğretim Uygulamaları, (2. baskı, 1998). BİTAV

Aşkar, P. (1991), Bilgisayar Destekli Öğretim Ortamı, Eğitimde Nitelik Geliştirme Eğitimde Arayışlar I. Sempozyumu Bildiri Metinleri, İstanbul.

Aşkar, P. (1992), İlköğretimde Bilgisayar: Kuram ve Uygulamalar, H.Ü. Eğitim Fakültesi Dergisi, Türkiye'de İlköğretim Sempozyumu 21-22 Mayıs 1992, Sayı: 8, Ankara.

Ausubel, D. P. (1960), The Use of Advance Organizers in the Learning and Retention of Meaningful Verbal Material, Journal of Educational Psychology, 51(5), 267-272

Backer, B. W. (2001), Teaching CS1 with Karel the Robot in Java, ACM SIGCSE Bulletin, in Proceeding of the 32nd SIGSCE technical symposium on Computer Science Education.

Baldwin L. P., Kuljis J. (2000), Visualisation Techniques for Learning and Teaching Programming, 22nd Int. Conf. Information Technology Interfaces IT1 2000, June 13-16, 2000, Pula, Croatia

Baldwin L. P., Kuljis J. (2001), Learning Programming Using Program Visualization Techniques, Proceedings of the 34th Hawaii International Conference, 2001

Bayman, P., & Mayer, R. (1988), Using conceptual models to teach BASIC computer programming, Journal of Educational Psychology, 80(3), 291-298.

Bergin J., Martinez M. P. (1996), An overview of visualization: its use and design, Report of the Working Group on Visualization, Integrating Tech. into C.S.E. 6/96 Barcelona, Spain

Brown, J. S., Collins, A., Duguid, P. (1989, January-February). Situated cognition and the culture of learning. Educational Researcher, 32-42.

Carter, J., Jenkins, T. (1999), Gender and Programming: What's Going On?, ITiCSE '99 6/99 Cracow. Poland

Cooper ,S. , Dann, W., Pausch, R. (2003), Teaching Object-first in Introductory Computer Science, Avalilable: http://www.alice.org/publications/pubs/TeachingObjects-firstInIntroductoryComputerScience.pdf

Dann, W., Cooper, S., Pausch, R. (2000), ALICE : A 3D Tool for Introductory Programming Concepts. The Journal of Computing in Small Colleges, Proceedings of the 5th annual CCSC northeastern conference on the Journal of Computing in small colleges, V 15 , I 5.

Dershem, H. L., Brummund, P. (1998), Tools for Web-Based Sorting Animation, ACM SIGCSE Bulletin, Proceeding of the 29th annual SIGCSE conference on technical symposium on Computer Science Education, V30, I 1.

Esteves M., Mendes A. J. , (2004), A Simulation Tool To Help Learning Of Object Oriented Programming Basics, 34 ASEE/IEEE Frontiers in Education Conference, October 20 – 23, 2004, Savannah, GA

Gagne, R. M., Briggs, L. J., & Wager, W. W. (1988), Principles of instructional design (2nd ed.) New York: Holt, Rinehart and Winston, Inc.

Garner S., (2003), Learning Resources and Tools to Aid Novices Learn Programming, Informing Science InSITE - "Where Parallels Intersect", June 2003

Gentner, D., & Gentner, D. R. (1983), Flowing waters or teeming corwds: Mental models of electricity. In D. Gentner & A. L. Stevens (Eds.), Mental Models (pp.99-129). Hillsdale, NJ: Lawrence Erlbaum Associates.

George, C. E. (2000), EROSI – Visualizing Recursion and Discovering New Errors,In Proceeding of the Thirty-First SIGCSE Technical Symposium on Computer Science Education, Austin Texas, March 2000 : 305-309

Glynn, S. M., Britton, B. K., Semrud-Clikeman, M. & Muth, K. D. (1989), Analogical reasoning and problem solving in science textbooks, In J. A. Glover, R. R. Ronning, & C. R. Reynolds (Eds), Handbook of

Creativity: Assessment Theory and Research (pp.383-398). New York: Plenum.

Grant, N. S. (2003), A Study on Critical Thinking, Cognitive Learning Style, and Gender in Various Information Science Programming Classes, CITC4'03, October 16–18, 2003, Lafayette, Indiana, USA.

Hirakawa. M., Tanaka M. and Ichikawa T. (1990), HI-VISUAL Iconic programming environment, In T. Ichikawa, E. Jungert and R.R. Korfhage (Eds.) Visual Languages and Applications. Plenum Press, New York, 12, 1-145

Kulik, J.A., Bangert, L.R. ve Williams, G.W. (1983), Effects of Computer-based Teaching on Secondary School Students, Journal of Educational Psychology, 75, 19-26.

LAI S., REPMAN J. L. (1996), The Effects of Analogies and Mathematics Ability on Students' Programming Learning Using Computer-Based Learning, International Journal of Instructional Media v23 no4 p355-64

Lakoff G., Johnson M. (1980), Metaphors we live by, Component instructional design model for training complex, University of Chicago Press, Chicago, IL

Linn, M. C. (1985), The Cognitive Consequences of Programming Instruction in Classrooms, Educational Researcher, 14(5), 14-16, 25-29.

Martens, R. L., Valcke, M. M. A. & Portier, S. J. (1997). Interactive learning environments to support independent learning: the impact of discernability of embeded support devices. Computer & Education, 28(3)

McGill, T. J., Volet, S. E. (1997), "A Conceptiual Framework for Analyzing Students' Knowledge of Programming, Journal of Research on Computing in Education, Vol 29(3), 276-197

MCKENNA, P. (2004), Gender and Black Boxes in the Programming Curriculum, ACM Journal of Educational Resources in Computing, Vol. 4, No. 1, March 2004.

Miyadera Y., Huang N. & Yokoyama S. (2000), A programming language education system based on program animation, International Conference On Educational Uses of Communication And Information Technologies, 2000

Murphy, L., McCauley ,R., Westbrook, S. (2006), Women Catch Up: Gender Differences in Learning Programming Concepts. SIGCSE'06, March 1–5, 2006, Houston, Texas, USA.

Naps, T. L., Eagan, J. R., Norton, L. L. (2000), JHAVE- An Environment to Actively Engage Students in Web-based Algorithm Visualisations", ACM SIGCSE Bulletin, Proceeding of the 31th SIGCSE technical symposium on Computer Science Education, V32, I 1.

Newby, T. J. Ertmer, P. A., & Stepich, (1995), Instructional analogies and the learning of concepts, Educational Technology, Research, & Development, 43(1), 19-30.

Odabaşı, F. (1998), Bilgisayar Destekli Eğitim, T.C. ANADOLU ÜNİVERSİTESİ YAYINLARI NO: 1059, p. 133-147, 1988

Olsen, K. A., Pedersen, B., Harnes, P. and Tosse, O.J., (1990), A visual system to support teaching of programming, In S-K. Chang (ed.) Visual Languages and Visual Programming. Plenum Press, New York 277-288

Ploedereder, E. (2000). The Sort Algorithm Animator V1.0. Retrieved from the World Wide Web 20/08/2006 http://www.informatik.uni-stuttgart.de/ifi/ps/Ploedereder/sorter/sortanimation2.html

Raeder, G. (1985), A survey of current graphical programming techniques. IEEE Computer. August 1985, 11-25

Reigeluth, C. M. (1983), Meaningfulness and instruction: Relating what is being learned to what a student knows, Instructional Science, 12, 197-218.

Rodger, S. (2002), Using Hands-on Visualisation to Teach Computer Science from Beginning Courses to Advanced Courses, Second Program Visualisation Workshop, Denmark, Available : http://www.cs.duke.edu/~rodger

Rowe, G. & Thorburn, G. (2000), VINCE - an on-line tool for teaching introductory programming, British Journal of Education Technology, 31(4), 359-370.

Shneiderman, B. (1977), Measuring computer program quality and comprehension, International Journal of Man-Machine Studies, 9, 465-478.

Shneiderman, B. & Mayer, R. (1979), Syntactic/semantic interactions in programmer behavior: a model and experimental results,

International Journal of Computer and Information Sciences, 8(3), 219-238.

Smith D.C., Cypher A., Tesler L. (2000), Novice Programming Comes of Age, Communications of the ACM, Vol. 43, No. 3, 2000, pp. 75-81.

Stasko, J. T., Byrne, M. D., Catrembone, R. (1996), Do Algorithm Animations Aid Learning?, Tech. Rep. GIT-GVU-96-18, GVU Centre, Georgia Institute of Technology, Atlanta, GA.

Tsoua, W., Wang, W. & Tzeng, Y. (2004). Appliying a multimedia storytelling website in foreign language learning. Computer & Education, In press.

Treagust, D. F. (1993), The evolution of approach for using analogies in teaching and learning science, Research in Science Education, 23, 293-301.

Vinsonhaler,J.F., & Bass, R.K. (1972), A summary of ten major studies on CAI drill and practice, Educational Technology, 1972,Vol.12,pp:29-32

Wilson, B. & Cole, P. (1991). A review of cognitive teaching models. Educational Technology Research and Development, 39(4), 47-64

Ekler

Ek 1 : Başarı Testi

Sevgili Öğrenciler;

Bu uygulama sizin bilgisayar programlamasında yer alan temel kavramlara ilişkin ön bilgilerinizi belirlemek için hazırlanmıştır. Uygulama 2 bölümden oluşmaktadır. İlk bölüm kişisel bilgilerinizi, diğer bölüm ise programlamaya ilişkin bilgilerinizi içeren sorulardan oluşmaktadır.

BÖLÜM I

Ad-Soyad : …….....

No : …….....

Cinsiyet : Erkek ( ) Kız ( )

Mezun Olduğunuz Okul: …….....

Mezun Olduğunuz Bölüm: …….....

Daha Önce Programlama Eğitimi Aldınız mı? Evet ( ) Hayır ( )

Yanıtınız "Evet" ise Nerede Aldınız?

Okulda ( ) Kursta ( )

Kendi Başıma Öğrenmeye Çalıştım ( )

Diğer, Belirtiniz …….....

BÖLÜM II

Sorular:

1) Programlamada kullanılan;

a) "değişken" kavramını tanımlayınız.

……..…

…….....

b) "veri" kavramını tanımlayınız.

….....……...……

…….....

c) "operatör" kavramını tanımlayınız.

……...……

…….....

d) “işlem (komut)” kavramını tanımlayınız.

……..……

…….....

2) Değişken ile bellek arasındaki ilişkiyi açıklayınız.

……

…...…….....

3) İşlem (komut) ile program arasındaki ilişkiyi açıklayınız.

……

…...……

4) Programlamada “=” sembolü ne işe yarar? Belirtiniz.

……

…...……

5) Bir “işlem (komut)” hangi bileşenlerden oluşur? Sıralayınız.

……

…....……

6) İşlem ve işlemci arasındaki ilişkiyi açıklayınız.

……

…...……

7) Programlamada kullanılan 3 adet operatör sembolünü aşağıya yazınız..

……

……

Uzun yanıtlarınız için (ilgili soruyu belirttikten sonra) aşağıdaki boş alanı kullanabilirsiniz.