

# Veri Yapıları Dersinin Listeler Konusu için Simülasyonlu Bir Eğitim Materyali Yazılımı

Tuncay Aydoğan<sup>1</sup>, Kasım Delikanlı<sup>2</sup>

<sup>1</sup> Süleyman Demirel Üniversitesi, Teknik Eğitim Fakültesi, Elektronik-Bilgisayar Bölümü, 3200, Isparta

<sup>2</sup> Süleyman Demirel Üniversitesi, Uluborlu MYO, Bilgisayar Programı 3200, Isparta

taydogan@tef.sdu.edu.tr, kasim\_d@hotmail.com

**Özet:** Programlama eğitiminin bazı konularında çeşitli nedenlerden dolayı zorluk çekilmektedir. Bunun nedeni bazı kavramların anlamlarındaki benzerliklerden doğan karmaşıklıklar ve bazı konuların somut örneklerle temsil edilememesidir. Günümüzde birçok konunun eğitiminde bilişim, eğitim, öğretim teknolojileri kullanılmaktadır. Bu çalışmada da, Veri Yapıları dersinin Listeler konusunun daha iyi ve çabuk öğretilmesi/öğrenilmesine katkıda bulunacak bir materyal geliştirilmiştir. Hazırlanan materyal, etkileşimli bir simülasyon programı olarak oluşturulmuştur.

**Anahtar Kelimeler:** Veri Yapıları, Eğitim Materyali, Simülasyon, Uzaktan Eğitim

## A Simulated Education Material for Lists Subject of Data Structure Lecture

**Abstract:** There are some difficulties in some specific chapters of programming education. Some of these problems caused owing to the similarity of some terms and those terms cannot be presented by concrete examples. Nowadays in teaching of many subjects; information technologies, educational technologies and instructional technologies are being used. In this study, a material that is aimed to support the instruction of a specific subject, "Linked Lists", of the lecture "Data Structures" more accurately and more quickly was developed. The material developed is created as an interactive simulation program.

**Keywords:** Data Structure, Education Material, Simulation, Remote Education

### 1. Giriş

Son teknolojik ve bilimsel atılımlar sonunda, haberleşme ve iletişim teknolojilerindeki gelişmelerden dolayı, adeta insanlar arasındaki sınırlar kalkmıştır. Böylece, bilgi eskisine göre daha kolay paylaşılabilir, dağıtılabilir ve ulaşılabilir olmuştur. Bu durum, dünyanın her alandaki dengelerini etkilemiş ve birçok şeyin bu değişen düzene göre yeniden gözden geçirilmesine, hatta yeniden yapılandırılmasına sebep olmuştur. Günümüzün "Bilişim Çağı" olarak adlandırılmasına neden olan bu gelişmeler hayatımıza çeşitli araçlarla/cihazlarla yansımaktadır. Bu araçlardan biriside bilgisa-

yarlardır. Diğer araçların da çoğu aslında bilgisayarlı veya programlanabilirdir. Bu yüzden bir cihazın programlanabilir olması onun teknolojik olması anlamına da gelmektedir.

Gittikçe daha da çeşitlenen ve karmaşıklaşan programlanabilir cihazlar tasarımlarına göre farklı tekniklerle ve algoritmalarla programlanır. Bazen basit, sıradan algoritmalarla programlar hazırlamak yeterli olsa da bazen de alışılmış algoritmaların dışına çıkmak gerekmektedir. Uygulama alanlarına göre grafik programlama, sistem programlama, ağ programlama, web programlama, mobil cihaz programlama, veritabanı programlama gibi birçok programlama

çeşidi vardır. Her programlama alanının kendine has teknikleri, algoritmaları hatta eğitimleri mevcuttur. İhtiyaçlara ve eğitim koşullarına göre daha etkili/verimli programlama eğitimi verebilmek için çeşitli eğitim ve öğretim teknolojisi yöntemleri kullanılmaktadır.

Eğitim teknolojisi öğrenme sürecini geliştirmek için oluşturulan her türlü sistemi, tekniği ve yardımı içerir. Böyle bir yapıda şu dört özellik önemlidir [1]; (1) Öğrencinin ulaşması hedeflenen amaçların tanımlanması, (2) Öğrenilecek konunun öğretim ilkelerine göre analiz edilip, öğrenilmeye uygun şekilde yapılandırılması, (3) Konunun aktarılabilmesi için uygun medyanın seçilip kullanılması, (4) Dersin ve derste kullanılan araçların etkililiğini ve öğrencilerin başarı durumlarını değerlendirmek için uygun değerlendirme yöntemlerinin kullanılması.

Öğretim teknolojileri ise iki şekilde tanımlanmaktadır [2]; (1) İletişim devrimi ile birlikte şekillenen medyanın, öğretmen, kitap, yazı tahtası ile beraber öğretimsel amaçlar için kullanılmaya başlamasıdır, (2) Belirlenmiş hedefler uyarınca, daha etkili bir öğretim elde etmek için, öğrenme ve iletişim konusundaki araştırmaların ve ayrıca insan kaynakları ve diğer kaynakların beraber kullanılmasıyla tüm öğrenme/öğretme sürecinin sistematik bir yaklaşımla tasarlanması, uygulanması ve değerlendirilmesidir.

Uzaktan eğitim, e-eğitim yöntemleri ve etkili eğitim materyalleri günümüzün geliştirilmekte olan popüler eğitim, öğretim teknolojilerinin bir ürünü olarak ortaya çıkmıştır. Programlama eğitimlerinde de bunlardan faydalanılmaktadır. Konuyla ilgili literatürde ve uygulamada birçok örneğe rastlanmaktadır [3, 4, 5, 6].

Bu çalışmada, programlama eğitimlerinin temel derslerinden olan “Veri Yapıları”nın, “Listeler” konusunun anlaşılmasını zorlaştıran nedenlere değinilerek, anlaşılmasına katkıda bulunması için hazırlanan bir simülasyon programı sunulmaktadır.

## 2. Listeler Konusu Eğitimi ve Hazırlanan Programın Yapısı

Veri, bilgisayar ortamında sayısal, alfasayısal veya mantıksal biçimlerde ifade edilebilen her türlü değerdir. Örn; 10, -2, 0 tamsayıları, 27.5, 0.0256, -65.253 gerçel sayıları, ‘A’, ‘B’ karakterleri, “Yağmur”, “Merhaba” karakter katarları, 0, 1 mantıksal değerleri, ses ve resim sinyalleri vb. bir veridir.

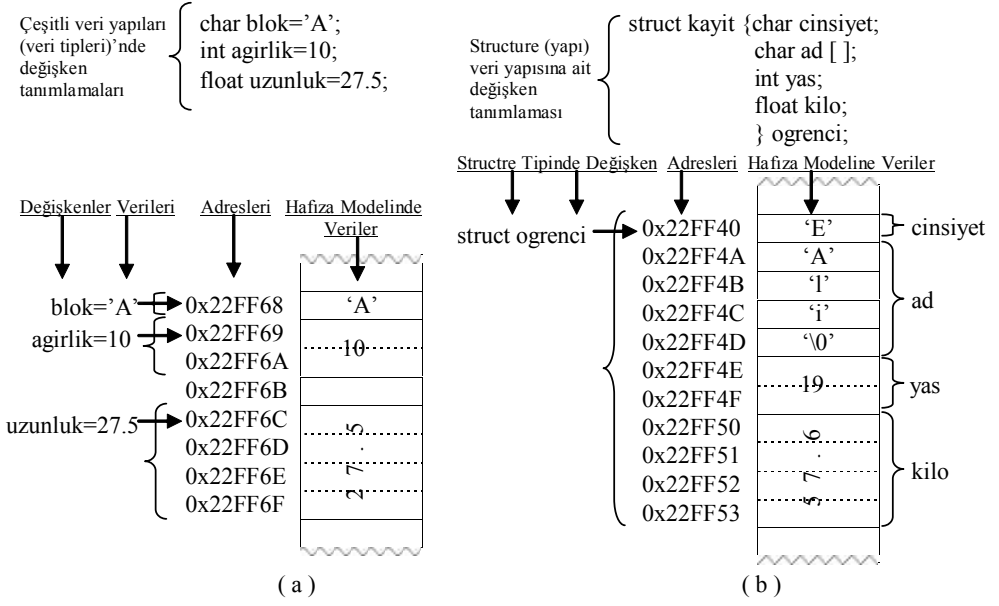
Bilgi ise verinin işlenmiş ve bir anlam ifade halidir. Örn; 10 kg, -2 derece, 0 noktası anlamlarındaki tamsayılar, 27.5 cm, 0.0256 gr, -65.253 volt anlamlarındaki gerçel sayılar, ‘A’ bina adı, ‘B’ sınıfın şubesi anlamlarındaki karakterler, “Yağmur” öğrencinin ismi, “Merhaba” selamlama kelimesi karakter katarları, boş anlamında 0, dolu anlamında 1 mantıksal değerleri, anlamı bilinen ses ve resim sinyalleri verilerin bilgi haline dönüşmüş halleridir.

Veriler büyüklüklerine göre bilgisayar belleğinde farklı boyutlarda yer kaplarlar. Büyüklüklerine, kapladıkları alan boyutlarına ve tanım aralıklarına göre veriler “Veri Tipi”leri ile sınıflandırılmışlardır. Örn; C programlama dilinde char veri tipi bellekte 8 Bit alan kaplarken tanım aralığı -127 ila 127 arasındadır, int veri tipi bellekte işlemciye göre 16 veya 32 Bit alan kaplarken tanım aralığı -32,767 ila 32,767 arasındadır. Her programlama dilinin bu örnekteki benzer, kabul edilmiş veri tipi tanımlamaları vardır. Programcı, programını yazacağı problemi incelerken, program algoritmasını oluştururken programda kullanacağı değişken ve sabitlerin veri tiplerini bu tanımlamaları dikkate alarak belirler. Çünkü veriler bellekte veri tiplerinden kendisine tanımlanan özelliklerinde saklanır.

Veri Yapısı, verileri tanımlayan veri tiplerinin, birbirleriyle ve hafızayla ilgili tüm teknik ve algoritmik özellikleridir. C Veri Yapıları; Temel/İlkel (Primitive), Basit (Simple), Birleşik (Compound) olarak üç sınıfta incelenebilir. Temel/İlkel (Primitive) veri yapıları, en çok kulla-

nilan ve diğer veri yapılarının oluşturulmasında kullanılan integer, float, boolean, char veri yapılarıdır. Basit (Simple) veri yapıları, Temel/İlkel (Primitive) veri yapılarından faydalanılarak oluşturulan diziler (arrays), karakter katarları (stringler), yapılar (structures) ve birlikler (uni-

ons) dir. Birleşik (Compound) veri yapıları, Temel/İlkel (Primitive) ve Basit (Simple) veri yapılarından faydalanılarak oluşturulan diğerlerine göre daha karmaşık olan stack, queue, list, tree, graph vb. veri yapılarıdır [7, 8, 9].



Şekil 1.a.b. C Programlama Diline Ait Bazı Veri Yapılarının Örnek Hafıza Modeli Üzerinde Gösterimi

Program, işlemci ve işletim sistemi her veri yapısına ait verileri, farklı biçim ve teknikler kullanılarak, bellekte yazma ve okuma işlemleriyle uygulamalara taşırlar. Bu işlemlere kısaca “Veri Yapıları Algoritmaları” denir. Çeşitli veri yapıları oluşturmak ve bunları programlarda kullanmak programcıya programlama esnekliği sağlarken, bilgisayar donanım ve kaynaklarından en etkin biçimde faydalanma olanakları sunar, ayrıca programın hızını, etkinliğini artırır, maliyetini düşürür. Ancak, özellikle Birleşik (Compound) veri yapılarının öğrenilmesi, öğretilmesi diğerlerine göre biraz daha zor olmaktadır.

Veri yapıları eğitimi aşağıdaki süreçlerle tamamlanmaktadır;

- veri kavramından bilgi kavramına geçiş
- veri yapılarının özelliklerinin öğrenilmesi

- belleğin yapısı ve özelliklerinin anlaşılması
- veri yapılarının bellekte temsilinin somutlaştırılması
- ‘&’ Adres Operatörü ve ‘\*’ Pointer veri tipinin veri yapıları tanımlamalarında kullanılmasının anlaşılması
- bir program kapsamında veri yapıları algoritmalarının uygulanması

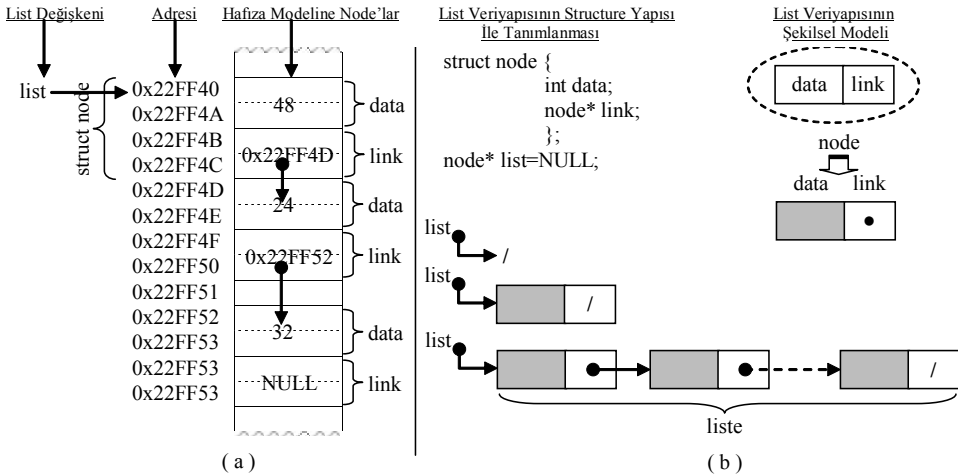
Bu eğitim süreci, alışılmış programlama algoritmalarının eğitim sürecine göre daha yavaş olmaktadır. Bunun nedeni yeni kavramları, tipleri ve operatörleri anlayabilme, algoritmaların bellekteki etkilerini somutlaştırarak kavrayabilmekteki zorluklardanır. Bu zorluklar genelde konuyu Şekil 1.’de görülen hafıza modellerine benzer modeller üzerinde anlatılarak aşmaya çalışılır. Şekil 1.a.’da hafızanın 1’er bytelik birbirinden bağımsız, farklı adreslere

sahip hücrelerden oluştuğu, programda tanımlanan değişkenlerin ve onların değerlerinin hafızada nasıl temsil edildiğini anlatılır. Bu bilgiler diğer veri yapıları için Şekil 1.b, ve Şekil 2.'deki gibi modeller üzerinde anlatılır.

Birleşik veri yapılarından listeler konusu ise node, liste kavramlarının tanımlanma, kodlanma biçimleri Şekil 2.a.'daki gibi hafıza modellerine benzer modeller üzerinde anlatılmaya çalışılır. Bir derece daha somutlaştırılarak anlatmak için Şekil 2.b.'deki gibi daha şekilsel bir model kullanılır.

Bu çalışmada, veri yapıları temel kavramlarının öğrenilmesi, Listeler konusunun daha iyi ve çabuk anlaşılabilmesi için işlev ve yapıları özel olarak tasarlanarak hazırlanmış 13 adet fonksiyon kullanılarak, bir simülasyon programı hazırlanmıştır. Bu programda simülasyonu yapılan fonksiyonlar ve görevleri Tablo 1.'de verilmiştir.

Bir Bağlı Doğrusal Listeler için gerçekleştirilen program çalıştırıldığında ilk önce Şekil 3.'de görülen "Liste Oluşturma Ekranı" gelmektedir. Bu ekranda, kullanıcı önce oluşturmak istediği listenin node sayısını belirler, hafızanın duru-



Şekil 2.a.b. C Programlama Diline Liste Veri Yapıları Modellerinin Gösterimi

Tanımlama ve fonksiyonlar	Anlamları
<pre> struct node { int data;               node* link; }; node* list=NULL; node* l1=NULL; node* l2=NULL;                     </pre>	programda kullanılacak node yapısının ve kullanılacak listelerin tanımlanması
void dumplist(node* list)	list listesini ekrana liste
node* newnode()	hafızadan yeni bir node al
node* last(node* list)	list listesinin son nodunu bul
void addhead(node* node , node*& list)	list listesinin başına node nodunu ekle
void concatenate(node*& l1, node* l2)	l1 listesinin sonuna l2 listesini ekle
node* cons(int data )	yeni bir node'un data'sına data değerlerini ata
node* copy(node* list)	list listesinin kopyasını oluştur
node* locate(int data , node* list)	list listesinde data'sı data olan node varsa adresini al
bool member(node* node , node* list)	list listesinde node adresli node varmı
node* cuthead(node*& list)	list listesinin ilk nodunu kes
void free(node*& list)	list listesini iptal et/sil
bool advance(node*& point)	adresli point olan node'dan sonra bir node varsa poin'i ilerlet
bool deletenode(node* node , node*& list)	list listesinde node adresli node varsa bul ve sil

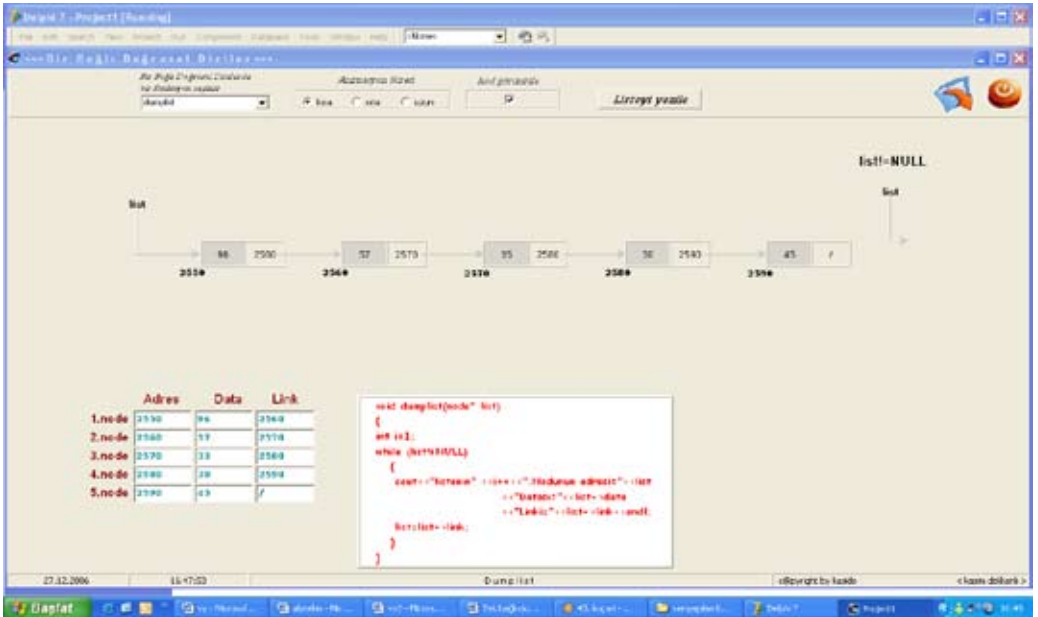
Tablo 1. Programda Simülasyonu Yapılan Fonksiyonlar ve Görevleri

muna göre rasgele oluşturulan nodlardan meydana gelen listenin adres, data, link değerleri tablo halinde görüntülenir. Kullanıcı tablodaki değerleri isterse değiştirme yetkisine sahiptir. Daha sonra “Liste Oluştur” butonuna basılmasıyla, katarlar biçiminde oluşmuş liste simülasyon işleminin gerçekleşeceği bir diğer ekranda karşımıza gelecektir (Şekil 4).

Şekil 4.’deki “Simülasyon Ekranı”nın seçenekler bölümünde, kullanıcı önce Tablo 3.’de listesi verilen fonksiyonlardan simülasyonu yapmak istediği fonksiyonu seçer. Sonra, simülasyonun işlem adım süresini kısa, orta, uzun seçeneklerinden belirler.



Şekil 3. Liste Oluşturma Ekranı



Şekil 4. Seçilen dumplist Fonksiyonuna İlişkin Simülasyon Görüntüsü

Ayrıca, isterse fonksiyon komut satırlarını ekranda görüntülenmesini de bu bölümden ayarlayabilir.

Şekil 4.’deki simülasyon ekranının simülasyon bölümünde, katar biçiminde oluşturulmuş liste üzerinde, seçilen fonksiyonun işlevinin simülasyonu gerçekleşir. Kullanıcı simülasyonu so-

nuna kadar izlerken hangi adımda hangi komut satırlarının liste üzerinde ne gibi etkileri olduğunu şekilsel olarak gözleyebilir.

Kullanıcı simülasyonu tekrarlamak için “Listeyi Yenile” butonunu kullanabilir. Seçenekler bölümündeki diğer butonlar ile önceki ekrana dönebilir veya programdan çıkabilir.

### 3. Sonuç

Bu çalışmada, programlamanın diğer konularına göre anlaşılması daha güç olan Veri Yapıları dersinin Listeler konusuyla ilgili bir simülasyon programı, ders materyali olarak geliştirilmiştir. Program sayesinde öğrenci, liste veri yapılarının özelliklerini, listelerin bellekte nasıl oluşturulduğunu ve temsil edildiğini, adres operatörü ‘&’ ve pointer ‘\*’ kullanımını daha kolay kavrayabilir. Soyut olan bu kavramları nispeten daha somut olarak düşünebilir.

### 4. Kaynaklar

- [1]. Collier, K. G. et al., 1971, “Colleges of education learning programmes: A proposal”, (Working Paper No.5). Washington, DC: Commission on Instructional Technology.
- [2]. Commission on Instructional Technology, 1970, “To improve learning. A report to the President and the Congress of the United States”, Washington, DC: Commission on Instructional Technology.
- [3]. Miyadera Y., Huang N., Yokoyama, S., “A programming language education system based on program animation”, Tokyo Gakugei University.

[4]. Matsuda, H., Shindo, Y., “Effect of using Computer Graphics Animation in Programming Education”, Nippon Institute of Technology.

[5]. Chen, T., Sobh T., “A Tool For Data Structure Visualization And User-Defined Algorithm Animation”, October 10 - 13, 2001 Reno, NV, 31st ASEE IEEE Frontiers in Education Conference.

[6]. Farklı Veri Yapıları Algoritmalarının Java Applet Olarak Hazırlanmış Simülasyonları <http://www.cosc.canterbury.ac.nz/people/mukundan/dsal/appldsal.html>.

[7]. Çölkesen, R., “Bilgisayar Programlama Ve Yazılım Mühendisliğinde Veri Yapıları Ve Algoritmalar”, Papatya Yayınevi.

[8]. Weiss, M.A., “Data Structures & Algorithm Analysis In C++”, Addison-Wesley.

[9]. Horowitz, E., Sahni, S., “Data Structures In Pascal”, Computer Science Pres.