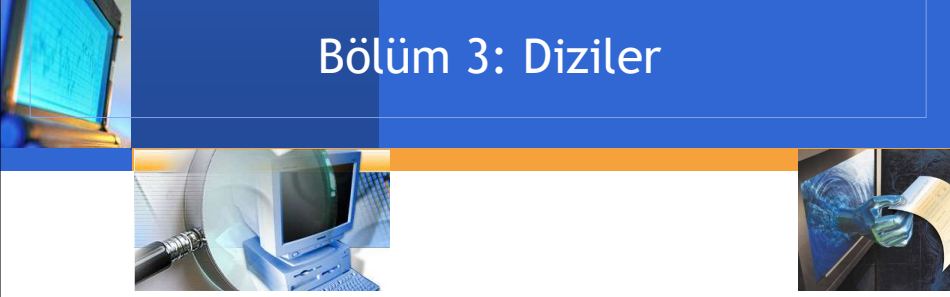


Doğu Akdeniz Üniversitesi  
Bilgisayar ve Teknoloji Yüksek Okulu  
Bilgisayar Programcılığı Bölümü

**BTEP 102 – Veri Yapıları ve Programlama**

**Bölüm 3: Diziler**



R. KANSOY

1

**Konu Başlıkları**


3. Diziler

- 3.1 Tek Boyutlu Diziler
- 3.2 Çok Boyutlu Diziler
- 3.3 Katar Dizileri

BTEP 102 – Veri Yapıları ve Programlama

2

2




## 3. Diziler

- ✓ Aynı isim altında, aynı türde birden fazla değer tutmak için kullanılan veri yapılarıdır.
- ✓ **Dizi** bir kümedir; aynı türde verilere tek bir isimle erişmek için kullanılır.
- ✓ Bir dizinin bütün elemanları bellekte ardışık olarak saklanır.
- ✓ Dizi elemanlarını birbirlerinden ayırt etmek için, dizi içindeki pozisyonları numaralandırılmıştır.
- ✓ Diziler bir veya daha çok boyutlu olabilirler.

3

3



## 3.1 Tek Boyutlu Diziler

- ✓ Bir grup sayısal değer ya da karakter veriyi içeren diziler tek boyutlu diziler olarak tanımlanır.
- ✓ Tek boyutlu diziler, **tür dizi\_adı [boyut]**, biçiminde tanımlanır.
  - **tür** Dizinin içerdiği değerlerin veri türü. Aynen değişken türlerinin tanımlandığı biçimde kullanılır.
  - **dizi\_adı** Dizinin mutlaka bir adı olmalıdır. Program içinde dizinin tüm elemanları bu ortak isim ile temsil edilir.
  - **boyut** Dizinin elemanları için bellekte ayrılacak yeri belirler. Ayrılan yerin tümüyle dolması gerekmez. Örneğin 10 elemanlık bir boyuta sahip dizinin 3 elemanı olabilir.

4

4

## 3.1 Tek Boyutlu Diziler

### Örnekler...

- ✓ `int a[6];`
  - 6 elemanlı, her elemanı int olan dizi.
- ✓ `char d2[15];`
  - 15 elemanlı, her elemanı char olan dizi.
- ✓ `float x[10];`
  - 10 elemanlı, her elemanı float olan dizi.
- ✓ `double k[10];`
  - 10 elemanlı, her elemanı double olan dizi.

5

5

## 3.1 Tek Boyutlu Diziler

### İndeks...

- ✓ Dizi içerisindeki her bir elemana, dizi isminden sonra, yazılan pozisyon numarası; yani indeks değeri ile ulaşılır.
- ✓ İndeks, bir dizinin her bir elemanına sırayla verilen bir numaradır.
- ✓ İndeks değerleri mutlaka tamsayı olmalıdırlar.
- ✓ İndeksler, sıfırdan başlayarak oluşturulurlar
- ✓ Buna göre dizinin birinci elemanının indeksi 0'dır. İkinci elemanının indeksi ise 1'dir.

Şekil 3.1: Bir dizinin her bir elemanı bir indekse göre yerleşmiştir.

6

6

### 3.1 Tek Boyutlu Diziler

**a**

a[0]	-12
a[1]	21
a[2]	35
a[3]	56
a[4]	0
a[5]	8

Dizi Adı

Elemanlar

İndeks numaraları

BTEP 102 – Veri Yapıları ve Programlama

7

7

### 3.1 Tek Boyutlu Diziler

#### Tek Boyutlu Dizilere Değer Atanması

- ✓ Bir dizi doğal olarak bazı veriler içerecektir.
- ✓ Bu değerler çoğunlukla program içinde belirli hesaplamalar sonucunda elde edilir.
- ✓ Bazı durumlarda atama yoluyla dizinin içi doldurulur.

**a[i]**

2	7	0	3	9
---	---	---	---	---

**i**


0	1	2	3	4
---	---	---	---	---

Şekil 3.2: Dizinin içi başlangıç değerler ile dolduruluyor.

BTEP 102 – Veri Yapıları ve Programlama

8

8




## 3.1 Tek Boyutlu Diziler

### Tek Boyutlu Dizilere Değer Atanması

- ✓ Bir diziye başlangıç değeri vermek için, ilgili değişkene o değer doğrudan atanır.
- ✓ `a[0]=2;`
  - dizinin 0 no'lu (ilk) elemanına 2 değerini ata.
- ✓ `a[2]=a[4]+3;`
  - dizinin 2 nolu elemanına 4 nolu elemanın 3 fazlasını ata.
- ✓ `for(i=0;i<6;i++)`  
`a[i] =0;`
  - 0'dan 6'ya kadar dizinin elemanına 0 değerini ata.

9

9




## 3.1 Tek Boyutlu Diziler

### Tek Boyutlu Dizilere Değer Atanması


**DİKKAT**

Aşağıdaki iki ifade birbirine eşit değildir.

**`a[i] + 1` ve `a[i+1]`**




Dizinin i no'lu  
indisindeki elemanın bir  
fazlası anlamına gelir.



Dizinin i'nin bir fazlası  
indisindeki elemanı  
anlamına gelir.

10

10



## 3.1 Tek Boyutlu Diziler


### Tek Boyutlu Dizilere Değer Atanması

- ✓ Diziye aynı anda birden fazla değer atanabilir.
- ✓ Bunun için söz konusu değerler { } işaretleri arasında yazılır doğal olarak bazı veriler içerecektir.
  - `a[5]={2,7,0,3,9}`
- ✓ Küme işaretleri içerisine dizinin eleman sayısından daha fazla eleman yazıldığında bir yazım hatası ortaya çıkar.
- ✓ Eleman sayısından daha az eleman yazıldığında ise, diğer elemanlar otomatik olarak sıfır değerini alır.

BTEP 102 – Veri Yapıları ve Programlama

11

11



## 3.1 Tek Boyutlu Diziler

### Tek Boyutlu Dizilere Değer Atanması

- ✓ İlk değer atanması yapılmayan dizilerin elemanları hafızadaki rasgele değerlerden oluşur.
- ✓ İlk değer atanması yapıldığında eleman sayısını yazma zorunluluğu yoktur. Ne kadar eleman yazılmışsa, eleman sayısı o kadar olur.

int a[5]={4,5};		int a[5]={0};		int a[5];		int a[]={4,5,3};	
a		a		a		a	
a[0]	4	a[0]	0	a[0]	Rastgele	a[0]	4
a[1]	5	a[1]	0	a[1]	Rastgele	a[1]	5
a[2]	0	a[2]	0	a[2]	Rastgele	a[2]	3
a[3]	0	a[3]	0	a[3]	Rastgele		
a[4]	0	a[4]	0	a[4]	Rastgele		

BTEP 102 – Veri Yapıları ve Programlama

12

12




## 3.1 Tek Boyutlu Diziler

### Tek Boyutlu Dizilere Değer Atanması

KOD 3.1 Dizi İçine Elemanlar Tek Tek Yerleştiriliyor	KOD 3.2 Dizi Elemanları Başlangıç değeri olarak Doğrudan Atanarak Yerleştiriliyor.	Sonuç
<pre>#include &lt;stdio.h&gt; int a[10]; main() {     a[0]=2;     a[1]=7;     a[2]=0;     a[3]=3;     a[4]=9;     printf("%d\n", a[0]);     printf("%d\n", a[1]);     printf("%d\n", a[2]);     printf("%d\n", a[3]);     printf("%d\n", a[4]); }</pre>	<pre>#include &lt;stdio.h&gt; int a[10]; main() {     int a[5]={2, 7, 0, 3, 9};      printf("%d\n", a[0]);     printf("%d\n", a[1]);     printf("%d\n", a[2]);     printf("%d\n", a[3]);     printf("%d\n", a[4]); }</pre>	<div>2</div> <div>7</div> <div>0</div> <div>3</div> <div>9</div>

13

13




## 3.1 Tek Boyutlu Diziler

### Örnek...

KOD 3.3 for Döngüsü ile Bir Dizi Elemanlarının Görüntülenmesi İşlemi	Sonuç
<pre>#include &lt;stdio.h&gt; int i; main() {     int a[5]={2,7,0,3,9};     for (i=0; i&lt;=4; i++)         printf("%d \n", a[i]); }</pre>	<div>2</div> <div>7</div> <div>0</div> <div>3</div> <div>9</div>

14

14



## 3.1 Tek Boyutlu Diziler

Örnek...

KOD 3.4 Dizinin İçerdiği Elemanların Toplanması	Sonuç
<pre>#include &lt;stdio.h&gt; int i, toplam; main() {     int a[5]={2,7,0,3,9};     for (i=0; i&lt;=4; i++)         toplam+=a[i];     printf("Toplam: %d", toplam); }</pre>	Toplam : 21

BTEP 102 – Veri Yapıları ve Programlama

15

15



## 3.1 Tek Boyutlu Diziler

Örnek...


KOD 3.5 Klavyeden girilen 5 adet tamsayıyı, giriş sırasının tersinden ekrana yazan C programı	Sonuç
<pre>#include &lt;stdio.h&gt; int main() {     int n[5], i;     printf("5 adet sayı giriniz\n");     for (i=0; i&lt;5; i++)         scanf("%d", &amp;n[i] );     printf("Girilen sayılar (sondan başa): \n");     for (i=4; i&gt;=0; i--)         printf("%d\n", n[i]);     return 0; }</pre>	<p>5 adet sayı giriniz:</p> <p>7</p> <p>16</p> <p>3</p> <p>6</p> <p>5</p> <p>Girilen sayılar (sondan başa):</p> <p>5</p> <p>6</p> <p>3</p> <p>16</p> <p>7</p>

BTEP 102 – Veri Yapıları ve Programlama

16

16






## 3.1 Tek Boyutlu Diziler

Örnek...

KOD 3.6 Dizi Elemanlarının Karekökünün Bulunması	Sonuç
<code>#include &lt;stdio.h&gt;</code>	2 sayisinin karekoku: 1.414214
<code>#include &lt;math.h&gt;</code>	7 sayisinin karekoku: 2.645751
<code>int i;</code>	0 sayisinin karekoku: 0.000000
<code>float karekok;</code>	9 sayisinin karekoku: 3.000000
<code>main()</code>	
<code>{</code>	
<code>int a[4]={2,7,0,9};</code>	
<code>for (i=0; i&lt;=3; i++)</code>	
<code>{</code>	
<code>    karekok=sqrt((float) a[i]);</code>	
<code>    printf("%d sayisinin karekoku: %f \n", a[i], karekok);</code>	
<code>}</code>	
<code>}</code>	

17

17



## 3.1 Tek Boyutlu Diziler

### Tek Boyutlu Dizilere Değer Atanması

**ÖRNEK:** Klavyeden girilen 10 adet tamsayı sınav notuna göre, ortalamanın üstünde olanları ekrana yazan C programı...

- ✓ Bu problem çözülürken öncelikle ortalamanın bulunması gereklidir.
- ✓ Ortalamanın bulunabilmesi için bütün notların toplamı alınacak ve not adedine bölünecektir.
- ✓ Bu noktadan sonra daha önceden girilmiş notların her biri sıra ile ortalamayla karşılaştırılacak büyük olanlar ekrana yazdırılacaktır.
- ✓ Çözümde dizi kullanılmayacak olsaydı, 20 ayrı değişkene ihtiyaç duyulacaktı.
- ✓ Çünkü, girilen her bir değere klavyeden girme işlemi bittikten sonra tekrar ulaşmak gerekecektir.

18

18



## 3.1 Tek Boyutlu Diziler

**KOD 3.7** Klavyeden girilen 10 adet sınav notundan, ortalamasının üstünde olanları ekrana yazan C programı...

**Sonuç**

```
#include <stdio.h>
int main()
{
    int n[10], i;
    double toplam=0, ortalama;
    printf("10 adet notu giriniz\n");
    for (i=0; i<10; i++) {
        scanf("%d", &n[i]);
        toplam=toplam+ n[i];
    }
    ortalama=toplam/10;
    printf("Sınıfın ortalaması=%.2f\n", ortalama);
    printf("Ortalamadan yüksek olan notlar:\n");
    for (i=0; i<10; i++) {
        if( n[i]>ortalama)
            printf("%d\n",n[i]);
    }
    return 0;
}
```

```
10 adet notu giriniz:
48
79
34
56
98
23
27
12
50
85
Sınıfın ortalaması = 51.2
Ortalamadan yüksek olan notlar:
79
56
98
85
```

19

19




## 3.2 Çok Boyutlu Diziler

- ✓ Birden fazla indeks numarası ile elemanlarına ulaşılan dizilere çok boyutlu diziler denir.
- ✓ C dilindeki dizilerin boyutları ikiden de fazla olabilir. Bir sınır olmamakla beraber en az 12 boyuta kadar destekler.
- ✓ Çok boyutlu bir dizi şu şekilde tanımlanmaktadır;
  - **Tür dizi\_adı [boyut1] [boyut2]..**

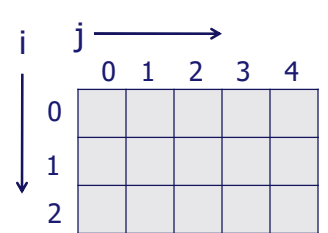
20

20



3.2 Çok Boyutlu Diziler

- ✓ Bunlardan en sık kullanılanı çift boyutlu dizilerdir.
- ✓ İki boyutlu dizi, satır ve sütunları olan bir tabloya benzer.
- ✓ Bu durumda 1. boyut satırları, 2.boyut ise sütunları gösterecektir.




Şekil 3.3: İki boyutlu bir dizinin görünümü.

- ✓ 3 satır ve 5 sütundan oluşan plan isimli iki boyutlu dizinin tanımı aşağıdaki gibidir;
  - `int plan [3] [5];`

BTEP 102 – Veri Yapıları ve Programlama

21

21



3.2 Çok Boyutlu Diziler

KOD 3.8 İki Boyutlu Diziye Değer Atama ve Görüntüleme İşlemi	Sonuç
<pre>#include &lt;stdio.h&gt;  char a[3][3] = {1,2,3,4,5,6,7,8,9}; int i,j;  main() {     for (i=0; i&lt;3; i++)     {         for (j=0; j&lt;3; j++)         {             printf("  %d", a[i][j] );         }         printf("\n");     } }</pre>	<pre>1  2  3 4  5  6 7  8  9</pre>

BTEP 102 – Veri Yapıları ve Programlama

22

22



## 3.2 Çok Boyutlu Diziler

**KOD 3.9 İki Boyutlu Dizi İçindeki Değerlerin Kareköklerini Bulma İşlemi**

```
#include <stdio.h>
#include <math.h>

float a[3][3] = {1.0,2.0,3.0,4.0,5.0,6.0,7.0,8.0,9.0};
int i,j;


main()
{
    for (i=0; i<3; i++)
    {
        for (j=0; j<3; j++)
            printf(" %f", sqrt(a[i][j]));
        printf("\n");
    }
}
```

**Sonuç**

1.000000	1.414214	1.732051
2.000000	2.236068	2.449490
2.645751	2.828427	3.000000

23

23



## 3.2 Çok Boyutlu Diziler

**KOD 3.10 İki Boyutlu Dizin Oluşturulması ve Görüntülenmesi**

```
#include <stdio.h>

char katar [4][3];
int i,j;

main()
{
    for (i=0; i<3; i++) /* Katarın içi dolduruluyor */
        for (j=0; j<4; j++)
            katar[i][j]=i*j;


    {
        for (i=0; i<3; i++) /* Katar Yazdırılıyor */
            for (j=0; j<4; j++)
                printf(" %d", katar[i][j]);
            printf("\n");
    }
}
```

**Sonuç**

0	0	0	0
0	1	2	3
0	2	4	6

24

24




## 3.3 Katar Dizileri

- ✓ Karakterlerin gruplanması ile oluşturulmuş veri yapılarına **katar** (string) adı verilir.
- ✓ C dilinde karakterler tek tırnak arasına alınmış ifadelerdir.
  - 'r'
  - 's'
- ✓ Karakter serileri, rakamları, harfleri, özel karakterleri içerebilir.
- ✓ C dilinde katar (string) ifadeleri çift tırnak (" ") arasında gösterilirler.
  - "Raygan Kansoy" (ad soyad)
  - "BTEP102" (ders kodu)
  - "0533 333 33 33" (telefon numarası)

BTEP 102 – Veri Yapıları ve Programlama

25

25




## 3.3 Katar Dizileri

- ✓ Katarlar, karakterlerden oluşan normal bir boyutlu dizi olarak değerlendirilir.
- ✓ Bir katar, boşluk (NULL) ile veya bir başka deyişle '\0' karakteri ile son bulan bir karakter dizisi olarak kabul edilir.
- ✓ Hafızada tutulan bir katağa ilk karakterinin adresi ile erişilir.
- ✓ Başlangıç adresinden '\0' karakterine kadar olan bölgede, katarın kendisi vardır.

BTEP 102 – Veri Yapıları ve Programlama

26

26




## 3.3 Katar Dizileri

- ✓ C dilinde katar dizileri şu şekilde tanımlanır;
  - `char katar_adı [ katarın_boyutu ];`
- ✓ Her zaman katarın en son karakteri NULL olacağından, n boyutlu bir kata n+1 boyutluk yer ayırtmak gerekir.
- ✓ Örnek;
  - `char isim[14]=('r', 'a', 'y', 'g', 'a', 'n', ' ', 'k', 'a', 'n', 's', 'o', 'y', '\0');`
  - `char isim[14]="raygan kansoy";`

1	2	3	4	5	6	7	8	9	10	11	12	13	14
'r'	'a'	'y'	'g'	'a'	'n'	' '	'k'	'a'	'n'	's'	'o'	'y'	'\0'

27

27




## 3.3 Katar Dizileri

- ✓ Elemanları katar olan diziler tanımlamak mümkündür.
  - `char dizininadı[eleman_sayısı][katar_uzunluğu];`
- ✓ Örneğin en uzun 7 karakter olan 5 farklı isim bir çatı altında şöyle toplanabilir:
  - `char isim[5][8] = { "Semra", "Mustafa", "Ceyhun", "Asli", "Leyla" };`

	S0	S1	S2	S3	S4	S5	S6	S7
S0	'S'	'E'	'M'	'R'	'A'	'\0'		
S1	'M'	'U'	'S'	'T'	'A'	'F'	'A'	'\0'
S2	'C'	'E'	'Y'	'H'	'U'	'N'	'\0'	
S3	'A'	'S'	'L'	'I'	'\0'			
S4	'L'	'E'	'Y'	'L'	'A'	'\0'		

28

28




## 3.3 Katar Dizileri

### 3.3.1 Katarlar Üzerinde İşlem Yapan G/Ç Fonksiyonları

- ✓ Standart Giriş - Çıkış Fonksiyonları
  - `printf()` ve `scanf()`
- ✓ Formatsız Giriş ve Çıkış Fonksiyonları
  - `gets( )`; klavyeden girilen bir stringi bir değişkene aktarır.
  - `puts( )`; bir stringi ekrana yazar.
  - `getchar ( )`; klavyeden bir karakter okur ve enter tuşuna basılmasını bekler.
  - `putchar( )`; ekrana bir karakter yazar.

29

29




## 3.3 Katar Dizileri

### `printf()` ve `scanf()`

- ✓ `printf( )` ve `scanf( )` fonksiyonlar diğer tiplerde olduğu gibi formatlı okuma/yazma amaçlı kullanılır.
- ✓ String formatı `%s` dir.
- ✓ Katarlara değer atarken ya da katarlardan değer okurken, sadece katar adını yazmak yeterlidir. Yani `scanf( )` fonksiyonu içersine `&` işareti koymak gerekmez.
- ✓ `scanf( )`, katarın ilk adresinden başlayarak aşağıya doğru harfleri tek tek ataması gerektiğini bilir.
- ✓ Katar girilirken, `scanf` fonksiyonu, boşluk, tab, enter karakterlerini dizgi sonu olarak algılar.

30

30



## 3.3 Katar Dizileri


### printf() ve scanf()

KOD 3.11 Bir katarın farklı yöntemlerle ekrana yazılması	Sonuç
<pre>#include &lt;stdio.h&gt; int main() {     char dizi[7] = {'S', 'e', 'l', 'a', 'm', '!', '\0'};     int i;      /* Herbir karakteri ayrı ayrı alt alta yaz */     printf("Dizi elemanlari:\n");     for (i=0; i&lt;7; i++)         printf("dizi[%d] icerigi: %c\n", i, dizi[i]);     printf("\n");      /* 1. yöntem: her elemanı yanyana yaz */     printf("Butun dizi (1.yontem): ");     for (i=0; i&lt;7; i++)         printf("%c", dizi[i]);      /* 2. Yöntem: bütün diziyi yaz */     printf("\nButun dizi (2.yontem): ");     printf("%s\n", dizi);     printf("\n");     return 0; }</pre>	<p>Dizi elemanlari:</p> <p>dizi[0] icerigi: S  dizi[1] icerigi: e  dizi[2] icerigi: l  dizi[3] icerigi: a  dizi[4] icerigi: m  dizi[5] icerigi: !  dizi[6] icerigi:</p> <p>Butun dizi (1.yontem): Selam!  Butun dizi (2.yontem): Selam!</p>

BTEP 102 – Veri Yapıları ve Programlama

31

31



## 3.3 Katar Dizileri

### printf() ve scanf()


<p><b>KOD 3.12</b> Klavyeden girilen ismi ekrana yazan C programı</p> <pre># include &lt;stdio.h&gt; int main() {     char isim[20];     printf("İsim giriniz");     scanf("%s", isim);     printf("\nGirdiğiniz isim: %s dir", isim);     return 0; }</pre>	<p>Yukarıdaki örnekte tek bir printf( ) fonksiyonuyla bütün katarı yazdırabilirken, aşağıdaki örnekte katar elemanları tek tek, karakter karakter yazdırılıyor. Çıkan sonuç her iki örnekte de aynı olacaktır .</p> <p>Özellikle <i>for</i> döngüsü içersinde bulunan "<i>isim[i]!='\0'</i>" koşuluna dikkat etmek gerekiyor. İsteseydik, "<i>i &lt; 20</i>" yazar ve katarın bütün hücrelerini birer birer yazdırabilirdik.</p>
<p><b>KOD 3.13</b> Klavyeden girilen ismi ekrana yazan C programı</p> <pre># include &lt;stdio.h&gt; int main() {     char isim[20];     int i;     printf("İsim giriniz");     scanf( "%s", isim );     printf("Girdiğiniz isim: ");     for( i = 0; isim[i]!='\0'; i++ )         printf("%c", isim[i] );     printf("\n"); }</pre>	<p>Kelimenin nerede sonlandığını belirlemek için "<i>isim[i]!='\0'</i>" koşulu kullanıldı. Bunun anlamı; <i>isim</i> katarının elemanları, "<i>\0</i>" yani boş karaktere eşit olmadığı sürece yazdırmaya devam edilmesidir. Ne zaman ki kelime biter, sıradaki elemanın değeri "<i>\0</i>" olur; işte o vakit döngü sonlandırılır.</p>

BTEP 102 – Veri Yapıları ve Programlama

32

32





## 3.3 Katar Dizileri

### printf() ve scanf()

**KOD 3.14** Bir stringin içindeki 'm' karakterinin sayısını öğrenme

```
#include <stdio.h>
int main()
{
    char ktr[20];
    int i,sayac = 0;


    printf("\nBir seyler yazın :");
    scanf("%s",ktr);

    for(i=0;ktr[i];i++)
        if( ktr[i]=='m' ) sayac++;

    printf("yazılanların içinde %d tane m harfi var",sayac);
}
```

33

33




## 3.3 Katar Dizileri

### puts() ve gets()

- ✓ gets( ) fonksiyonu bir katarı değeri atamak için kullanılır.
- ✓ puts( ) fonksiyonu ise, bir katarın içeriğini ekrana yazdırmaya yarar.
- ✓ gets( ) atayacağı değerin ayrımını yapabilmek için '\n' aramaktadır. Yani klavyeden Enter'a basılana kadar girilen her şeyi, tek bir katarı atayacaktır.
- ✓ puts( ) fonksiyonuysa, printf( ) ile benzer çalışır. Boş karakter ( NULL Character ) yani '\0' ulaşana kadar katarı yazdırır; printf( ) fonksiyonundan farklı olarak sonuna '\n' koyarak bir alt satıra geçer.
- ✓ Birden fazla kelimeden oluşan ve boşluk içeren cümleler için gets( ) ve puts( ) fonksiyonları kullanılmaktadır.

34

34



## 3.3 Katar Dizileri

### puts() ve gets()

**KOD 3.15 puts() ve gets() Fonksiyonlarının Kullanımı**


```
#include<stdio.h>

int main(void)
{
    char cumle[40];
    printf( "Cümle giriniz> ");
    gets( cumle );
    printf( "Girdiğiniz cümle:\n" );
    puts( cumle );
    return 0;
}
```

BTEP 102 – Veri Yapıları ve Programlama

35

35



## 3.3 Katar Dizileri

### puts() ve gets()

**KOD 3.16 Bir Katarın İlk Karakterini Yazdırma**

```
#include<stdio.h>

main()
{
    char ad[10],soyad[20];

    printf("ADI : ");
    gets(ad);


    printf("SOYADI : ");
    gets(soyad);

    printf("%c. %s",ad[0],soyad);
}
```

BTEP 102 – Veri Yapıları ve Programlama

36

36



## 3.3 Katar Dizileri

### puts() ve gets()

**KOD 3.17 Bir Katarın Kaç Karakterden Oluştuğunu Hesaplama**

```
#include<stdio.h>

main()
{
    char ktr[200];
    int i;


    puts("Bir seyler yazın");
    gets(ktr);

    for(i=0;ktr[i];i++); /* kosul ktr[i]!='\0' anlamindadir*/
    printf("yazılan %d harfdır",i);
}
```

BTEP 102 – Veri Yapıları ve Programlama

37

37



## 3.3 Katar Dizileri


### 3.3.2 Bazı Katar Fonksiyonları

- ✓ Bu fonksiyonlar standart C'de iki katarı karşılaştırmak, bir katarın içeriğini diğerine kopyalamak, katarın uzunluğunu bulmak vb işlemler için tanımlı fonksiyonlardır.
- ✓ Bu fonksiyonlar, string kütüphanesinde bulunmaktadır. Bu yüzden, bu fonksiyonların kullanılacağı programların başına, `#include<string.h>` eklenmesi gerekir.

BTEP 102 – Veri Yapıları ve Programlama

38

38




## 3.3 Katar Dizileri

### Katar Kopyalama: **strcpy()**

- ✓ Bir dizgi içine C programı içinde belirli bir katar yerleştirilmesi işlemi geleneksel yollarla yapılamaz.
- ✓ Örneğin, aşağıdaki şekilde bir tanım hatalıdır. Çünkü bu atama göstergeye yapılan atamadır. *(Göstergelerin ne olduğu bir sonraki bölümde ayrıntılı olarak ele alınacaktır)*
  - `char katar[50];`
  - `katar="abcde";`
- ✓ Atamanın bir katara yapılabilmesi için, C'nin standart **strcpy()** fonksiyonu kullanılır.
- ✓ Bu fonksiyon şu şekilde tanımlanmaktadır;
  - `strcpy(katar, "karakter ifade");`
- ✓ **strcpy()** fonksiyonu, program içinde tırnak işaretleri arasında tanımlanmış katarın herbir karakterini, bir dizinin hücreleri içine tek tek yerleştirir.

39

39




## 3.3 Katar Dizileri

### Katar Kopyalama: **strcpy()**

KOD 3.18 Katar Kopyalama İşlemi	Sonuç:
<pre>#include&lt;stdio.h&gt; #include&lt;string.h&gt;  main() {     char katar[50];     int i;      strcpy(katar, "Kazakistan");      for(i=0;katar[i];i++); /* kosul katar[i]!='\0' anlamindadir */     printf(" %c \n", katar[i]); }</pre>	K a z a k i s t a n

40

40



## 3.3 Katar Dizileri

**Katarların Birleştirilmesi: strcat()**

- ✓ İki katarın birleştirilerek tek bir katar haline dönüştürülmesi söz konusu ise C'nin strcat() fonksiyonu kullanılır.
- ✓ Bu fonksiyon şu şekilde tanımlanmaktadır;
  - strcat(katar1, katar2);
- ✓ Bu fonksiyon, var olan bir katarın sonuna bir başka katarı ekleyecektir.
- ✓ Yukarıdaki tanıma göre, **katar2**, **katar1**'in sonuna eklenecektir.
- ✓ Örneğin "abc" katarının sonuna "def" katarı strcat() fonksiyonu kullanılarak eklenebilir.

BTEP 102 – Veri Yapıları ve Programlama

41

41

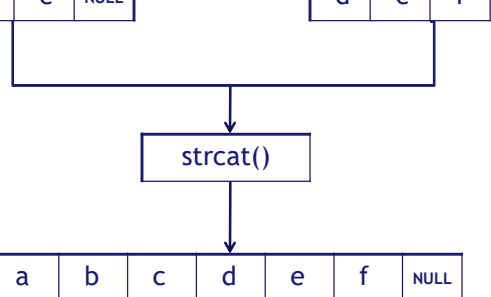


## 3.3 Katar Dizileri

**Katarların Birleştirilmesi: strcat()**

a	b	c	NULL
---	---	---	------

d	e	f	NULL
---	---	---	------



```

graph TD
    A[a b c NULL] --- J(( ))
    B[d e f NULL] --- J
    J --> C[strcat()]
    C --> D[a b c d e f NULL]
    
```


a	b	c	d	e	f	NULL
---	---	---	---	---	---	------

Şekil 3.4: strcat() fonksiyonunun çalışma biçimi.

BTEP 102 – Veri Yapıları ve Programlama

42

42



## 3.3 Katar Dizileri

**Katarların Birleştirilmesi: strcat()**

**KOD 3.19 İki Katarın Birleştirilmesi ve Sonucun Görüntülenmesi**

```
# include <stdio.h>
# include<string.h>

char katar[50];
int i;

main()
{
    strcpy(katar, "Kazakistan");
    strcat(katar, " ve Türkiye");

    for(i=0; katar[i]; i++)
        printf("%c", katar[i]);
}
```


Yandaki ifade dizinin her bir elemanını tek tek ekrana yazar. Bu ifadenin yerine printf(katar); ifadesi kullanılmış olsaydı aynı sonuç elde edilirdi.

Sonuç: Kazakistan ve Türkiye

BTEP 102 – Veri Yapıları ve Programlama

43

43



## 3.3 Katar Dizileri


**Katarların Karşılaştırılması: strcmp()**

- ✓ İki katarın birbirleriyle karşılaştırılarak, içerdiği karakterlerin aynı olup olmadıkları **strcmp()** fonksiyonu ile test edilir.
- ✓ Bu fonksiyon şu şekilde tanımlanmaktadır;
  - strcmp(katar1, katar2);
- ✓ Bu karşılaştırma sonucunda;
  - her iki katar birbirinin aynı ise, geriye "0",
  - ilk katar alfabetik olarak ikinciden büyükse, geriye **pozitif**,
  - ikinci katar birinciden büyükse, geriye **negatif** değer döner.

BTEP 102 – Veri Yapıları ve Programlama

44

44



## 3.3 Katar Dizileri

### Katarların Karşılaştırılması: **strcmp()**

**KOD 3.20 İki Katarın Karşılaştırılması**

```
# include <stdio.h>
# include <string.h>

main()
{
    char ktr1[10],ktr2[10];
    int sonuc;


    printf("1. katar:");gets(ktr1);
    printf("2. katar:");gets(ktr2);

    sonuc = strcmp(ktr1,ktr2);

    if(sonuc>0) puts("1. katar 2.den büyük");
    else if(sonuc<0) puts("2. katar 1.den büyük");
    else puts("1. katar 2. katara eşit");
}
```

45

45



## 3.3 Katar Dizileri

### Katarların Karşılaştırılması: **strcmp()**

**KOD 3.21 İki Katarın Karşılaştırılması**

```
# include <stdio.h>
# include <string.h>

char katar1[50], katar2[50];


main()
{
    strcpy(katar1, "Kazakistan");
    strcpy(katar2, " Kazak");

    if(strcmp(katar1,katar2))
        printf("Katarlar birbirinden farklı");
    else
        printf("Katarlar birbirinin aynı");
}
```

**Sonuç:** Katarlar birbirinden farklı

46

46



## 3.3 Katar Dizileri

### Katarların Karşılaştırılması: **strcmp()**

**KOD 3.22** Basit Bir Şifre Programı

```
# include <stdio.h>
# include <string.h>


main()
{
    char sifre[20];
    printf("SIFRE : ");
    scanf("%s",sifre);

    if( strcmp(sifre,"deneme")==0 )
        puts("sifre dogru girildi");
    else
        puts("sifre yanlis!");
}
```

BTEP 102 – Veri Yapıları ve Programlama

47

47



## 3.3 Katar Dizileri

### Katarların Uzunluğunu Bulmak: **strlen()**


- ✓ **strlen( )** fonksiyonu bir katarın uzunluğunu, yani kaç karakter içerdiğini elde etmek için kullanılır.
- ✓ Bu fonksiyon şu şekilde tanımlanmaktadır;
  - **strlen(katar);**
- ✓ Katarın uzunluğu bulunurken, içerdiği en son karakter olan **NULL** karakteri göz özüne alınmaz.
- ✓ Örneğin, katar "abc" değerlerini içeriyorsa, **strlen()** fonksiyonu bu katar için "3" değerini döndürür.

BTEP 102 – Veri Yapıları ve Programlama

48

48





## 3.3 Katar Dizileri

### Katarların Uzunluğunu Bulmak: **strlen()**

**KOD 3.23 Katar Uzunluğunun Bulunması**

```
# include <stdio.h>
# include <string.h>

main()
{
    char katar[50];


    strcpy(katar, "Kazakistan");
    printf("Uzunluk: %d", strlen(katar));
}
```

*Sonuç:* Uzunluk: 10

BTEP 102 – Veri Yapıları ve Programlama

49

49



## 3.3 Katar Dizileri

### Katarların Uzunluğunu Bulmak: **strlen()**

**KOD 3.24 Klavyeden Girilen Katarın Uzunluğunun Bulunması**


```
# include <stdio.h>
# include <string.h>

main()
{
    char ktr[100];
    puts("Birseyler yazın:");
    gets(ktr);
    printf("%s %d karakterden oluşmuştur.", ktr, strlen(ktr));
}
```

BTEP 102 – Veri Yapıları ve Programlama

50

50




## 3.3 Katar Dizileri

### Örnekler...

KOD 3.25 strcat fonksiyonunun kullanımı	Sonuç:
<pre># include &lt;stdio.h&gt; # include &lt;string.h&gt;  int main() {     char mesaj[20] = "Selam ";    /* 1. katar */     char isim[10];               /* 2. katar */      printf("Adiniz ? : ");     scanf("%s", isim);      strcat(mesaj, isim);          /* ekle */     printf("%s\n", mesaj);      return 0; }</pre>	<p>Adiniz ? : <b>Raygan</b> Selam Raygan</p>

51

51




## 3.3 Katar Dizileri

### Örnekler...

KOD 3.26 strcat fonksiyonunun kullanımı
<pre># include &lt;stdio.h&gt; # include &lt;string.h&gt;  int main() {     char ad[30], soyad[20];     char isim_soyad[50];      printf( "Ad ve soyadınızı giriniz&gt; " );     scanf( "%s%s", ad, soyad );      strcat( isim_soyad, ad );     strcat( isim_soyad, " " );     strcat( isim_soyad, soyad );      printf( "Tam İsim: %s\n", isim_soyad );      return 0; }</pre>

52

52



## 3.3 Katar Dizileri

### Örnekler...

KOD 3.27 Şifre Programı	Sonuç:
<pre># include &lt;stdio.h&gt; # include &lt;string.h&gt;  int main() {     char sifre[8];     int sonuc, hak=3;      while( hak-- &gt; 0 )     {         printf("Sifre : ");         gets(sifre); /* şifreyi al */          sonuc = strcmp(sifre,"elma%xj4");          if( sonuc==0 ) { /* şifre kontrol */             puts("sifre dogru");             break;         }         else             puts("sifre yanlis");     }     return 0; }</pre>	<pre>Sifre : admin sifre yanlis Sifre : root sifre yanlis Sifre : elma%xj4 sifre dogru</pre>

BTEP 102 – Veri Yapıları ve Programlama

53