

Programlamaya Giriş

Şadi Evren ŞEKER

Youtube: Bilgisayar Kavramları

1. Ders: Program Kavramı
2. Ders: Yazım Kuralları ve İşlemler (Operators)
3. Ders: Akış Kontrolü (Flow Control)
4. Ders: Döngüler (Loops)
5. Ders: Fonksiyonlar
6. Ders: Döngü (Loops) ve Fonksiyon Uygulamaları
7. Ders: Diziler (Arrays)
8. Ders: Göstericiler (Pointers)
9. Ders: Struct ve Dizgiler (Strings)
- 10.Ders: Dizi (Array) ve Gösterici (Pointer) Uygulamaları**
11. Ders: Dosya İşlemleri

www.youtube.com/sadievrenseker_bk

Bölüm İçeriği

- Dizi Örnekleri
- Permutasyon Algoritması
 - Sınırsız zar ihtimallerini basan algoritma
- Kutulama Problemi
- Metin Arama Algoritmaları
 - Algoritmanın Başarısı
 - Algoritmanın Çalışması ve bir Örnek
 - Algoritmanın Kodlanması

Örnek

- Sorumuz basitçe $n \times n$ boyutlarındaki bir matrisin (soruyu basitleştirmek için n sayısını tek sayı olarak kabul edeceğiz) içerisine aşağıdakine benzer şekilde ardışık sayıları sarmal olarak yerleştirmek.

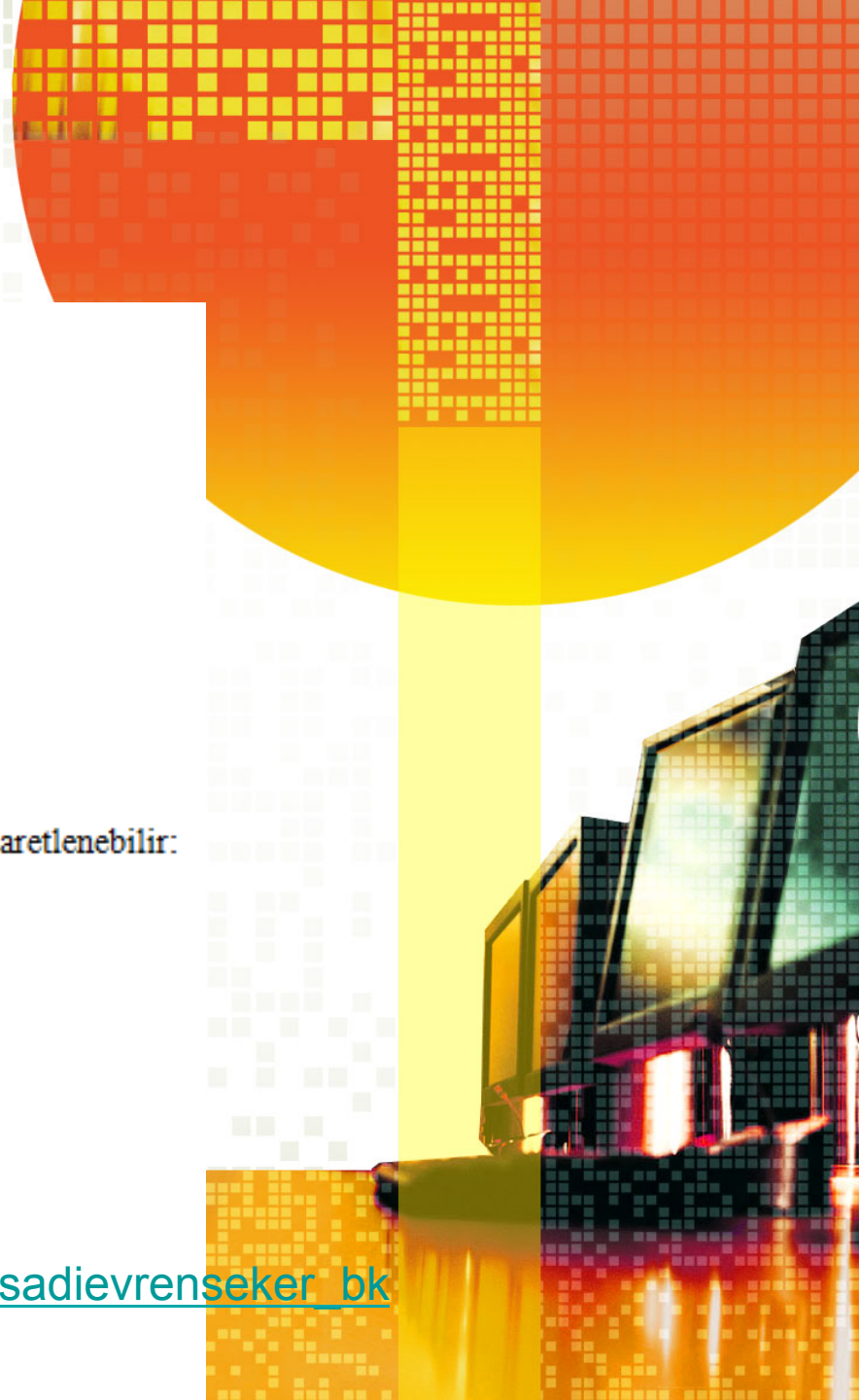
1	2	3
8	9	4
7	6	5

1	2	3	4	5
16	17	18	19	6
15	24	25	20	7
14	23	22	21	8
13	12	11	10	9

1	2	3	4	5	6	7
24	25	26	27	28	29	8
23	40	41	42	43	30	9
22	39	48	49	44	31	10
21	38	47	46	45	32	11
20	37	36	35	34	33	12
19	18	17	16	15	14	13

Click to add title

- İlk tespitimiz, 2 boyutlu bir matris içerisinde ardışık sayıları yerleştirmek dolayısıyla matrisin ortasındaki sayının, matris boyutunun karesi olacağını biliyoruz. Örneğin 7×7 boyutlarındaki matris için ortada 49 , veya 5×5 boyutlarındaki matris için ortadaki değerin 25 olması gibi.
- Ayrıca ortaya kadar olan köşegenleri bulursak tek boyutlu seriler oluşturmuş oluruz.
- Örneğin aşağıdaki renklendirilmiş değerlerden başlayan ve sağa doğru giden tek boyutlu seriler, 1'er 1'er artan değerlerdir:



1	2	3	4	5	6	7
24	25	26	27	28	29	8
23	40	41	42	43	30	9
22	39	48	49	44	31	10
21	38	47	46	45	32	11
20	37	36	35	34	33	12
19	18	17	16	15	14	13

Bu deęerlerden bařlayarak saęa doęru giden seri ařaęıdaki řekilde iřaretlenebilir:

1	2	3	4	5	6	7
24	25	26	27	28	29	8
23	40	41	42	43	30	9
22	39	48	49	44	31	10
21	38	47	46	45	32	11
20	37	36	35	34	33	12
19	18	17	16	15	14	13

- Serimizde bulunan sayılar:
- 1, 25, 41, 49
- Bu sayıların arasında bir bağlantı bulmamız gerekiyor. Sayıları tersten yazar ve aralarındaki farkları bulursak:
- 49, 41, 25, 1 ve aralarındaki farklar : 8 , 16, 24 şeklinde giden seridir. Bu serinin özelliği 8'in katları olmasıdır.

Çözüm

1	2	3	4	5	6	7
24	25	26	27	28	29	8
23	40	41	42	43	30	9
22	39	48	49	44	31	10
21	38	47	46	45	32	11
20	37	36	35	34	33	12
19	18	17	16	15	14	13

1	2	3	4	5	6	7
24	25	26	27	28	29	8
23	40	41	42	43	30	9
22	39	48	49	44	31	10
21	38	47	46	45	32	11
20	37	36	35	34	33	12
19	18	17	16	15	14	13

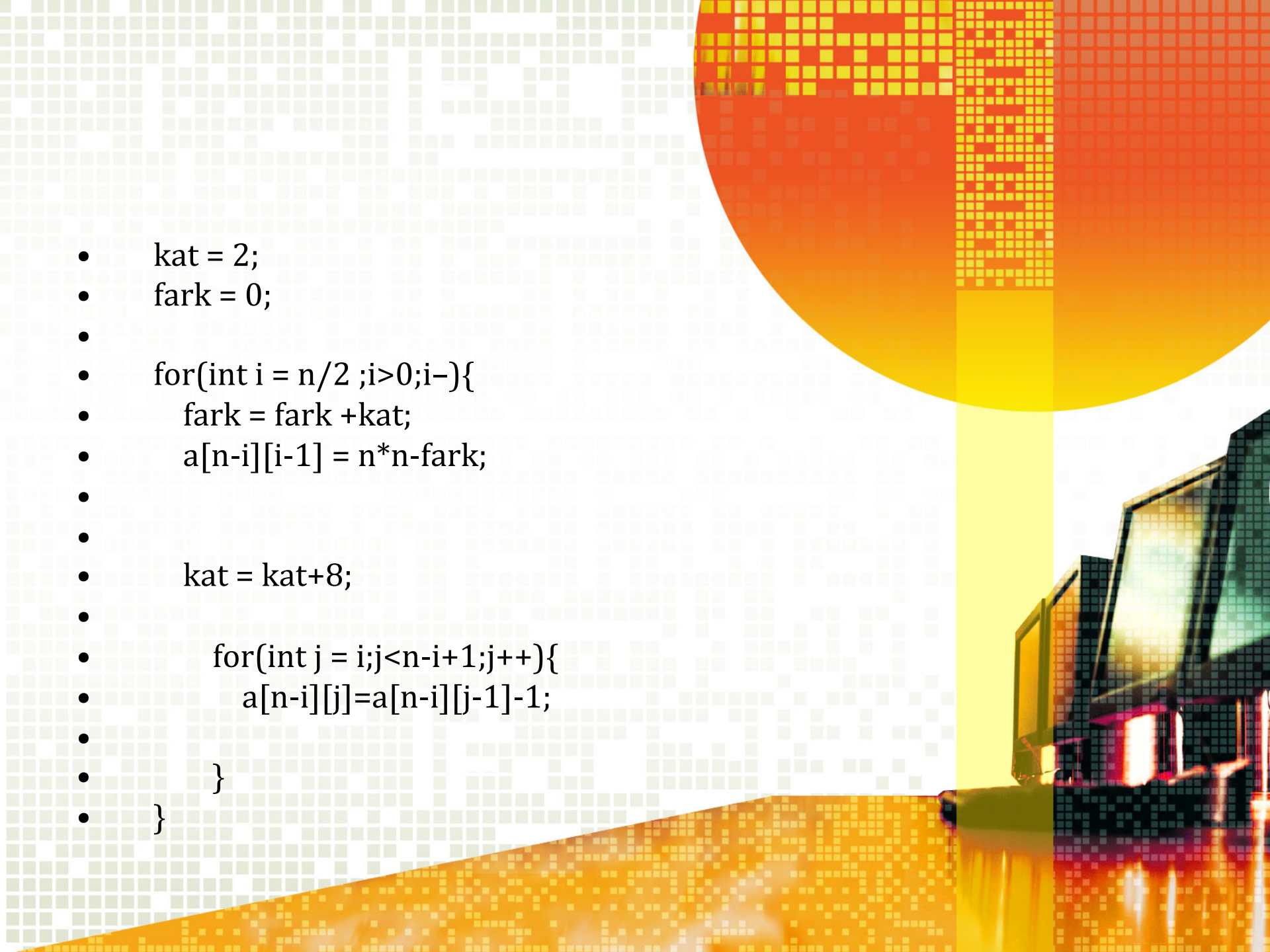
1	2	3	4	5	6	7
24	25	26	27	28	29	8
23	40	41	42	43	30	9
22	39	48	49	44	31	10
21	38	47	46	45	32	11
20	37	36	35	34	33	12
19	18	17	16	15	14	13

Özyineli Çözüm

- `int kat = 0;`
- `int fark = 0;`
- `for(int i = n/2 ;i>=0;i-){`
- `a[i][i] = n*n-fark;`
- `kat++;`
- `fark = fark + kat*8;`
- `for(int j = i+1;j<n-i;j++){`
- `a[i][j]=a[i][j-1]+1;`
- `}`
- `}`

- `for(int i = n/2 ;i>0;i-){`
- `for(int j = i;j<n-i+1;j++){`
- `a[j][n-i]=a[j-1][n-i]+1;`
- `}`
- `}`





- kat = 2;
- fark = 0;
-
- for(int i = n/2 ;i>0;i-){
- fark = fark +kat;
- a[n-i][i-1] = n*n-fark;
-
-
- kat = kat+8;
-
- for(int j = i;j<n-i+1;j++){
- a[n-i][j]=a[n-i][j-1]-1;
-
- }
- }

- `for(int i = 0 ;i<n/2;i++){`
- `for(int j = i+2;j<n-i;j++){`
- `a[n-j][i]=a[n-j+1][i]+1;`
- `}`
- `}`



- Kodun örnek çıktıları aşağıda verilmiştir:
- 7×7 boyutları için :

- 9×9 boyutları için:

C:\Users\shedai\Desktop\sarmal\Debug\sarmal.exe

1	2	3	4	5	6	7
24	25	26	27	28	29	8
23	40	41	42	43	30	9
22	39	48	49	44	31	10
21	38	47	46	45	32	11
20	37	36	35	34	33	12
19	18	17	16	15	14	13

C:\Users\shedai\Desktop\sarmal\Debug\sarmal.exe

1	2	3	4	5	6	7	8	9
32	33	34	35	36	37	38	39	10
31	56	57	58	59	60	61	40	11
30	55	72	73	74	75	62	41	12
29	54	71	80	81	76	63	42	13
28	53	70	79	78	77	64	43	14
27	52	69	68	67	66	65	44	15
26	51	50	49	48	47	46	45	16
25	24	23	22	21	20	19	18	17

Permutasyon

- *Bir fonksiyonumuz olsun ve parametre olarak zarSayisi alsin ve bu fonksiyon zar sayısına gore olası tum sonucları(zarların permutasyonunu ekrana bassın).ornegin zar sayısı 2 ise asagıdaki gibi sonuc elde etsin.*
- *Cıktı : 11 12 13 14 15 16 21 22 23 24 25 26 31 32 33 34 35 36 41 42 43 44 45 46 51 52 53 54 55 56 61 62 63 64 65 66.*
- *zarSayisi 3 ise aşagıdaki gibi*
- *Cıktı: 111 112 113 114 115 116 121 122 123 124 125 126 211 212... seklinde*


```

23 int main(){
24     int n ;
25     printf("zar sayisini giriniz");
26     scanf("%d",&n);
27     int basamak = 1;
28     int diz = 0;
29     for(int i = 0;i<n;i++){
30         diz = diz + basamak ;
31         basamak*=10;
32     }
33     printf("%d",diz);
34     int katar = diz;
35     diz = 0;
36     for(int i = 0;i<us(6,n);i++){
37         diz++;
38         printf(" %d\n",katar + tabancevir(diz,6));
39     }
40     getch();
41 }

```

C:\Dev-Cpp\Untitled1.exe

```

zar sayisini giriniz3
111 112 113 114 115 116 121 122 123 124 125 126 131 132 133 134 1
35 136 141 142 143 144 145 146 151 152 153 154 155 156 161 162 1
63 164 165 166 211 212 213 214 215 216 221 222 223 224 225 226 2
31 232 233 234 235 236 241 242 243 244 245 246 251 252 253 254 2
55 256 261 262 263 264 265 266 311 312 313 314 315 316 321 322 3
23 324 325 326 331 332 333 334 335 336 341 342 343 344 345 346 3
51 352 353 354 355 356 361 362 363 364 365 366 411 412 413 414 4
15 416 421 422 423 424 425 426 431 432 433 434 435 436 441 442 4
43 444 445 446 451 452 453 454 455 456 461 462 463 464 465 466 5
11 512 513 514 515 516 521 522 523 524 525 526 531 532 533 534 5
35 536 541 542 543 544 545 546 551 552 553 554 555 556 561 562 5
63 564 565 566 611 612 613 614 615 616 621 622 623 624 625 626 6
31 632 633 634 635 636 641 642 643 644 645 646 651 652 653 654 6
55 656 661 662 663 664 665 666

```

- Acaba bu zarların sırasız olması halini nasıl basarız sorunu cevaplamaya çalışalım.
- Problemi anlamak adına örneğin 2 adet zar atıldığında, 12 ile 21 ihtimalleri aynı oluyor. Yani zarların sırasının bir önemi kalmıyor.
- Bu durumda yukarıda yayınladığımız algoritma işe yaramaz. Çözüm olarak problemimizi, yukarıda anlattığımız üzere sayma problemi haline getirmeye çalışalım.



- $n = 1$ için
- 1,2,3,4,5,6
- $n = 2$ için
- 11,12,13,14,15,16,22,23,24,25,26,33,...
- $n = 3$ için
- 111,112,113,114,115,116,122,123,124,125,
126,133,...222,223,224,...


```
4 int *diziarattir(int *k,int n){
5     //www.bilgisayarkavramlari.com
6     k[n-1]++;
7     for(int i = n-1;i>=1;i--){
8         if(k[i]>6){
9             k[i]-=6;
10            k[i-1]++;
11        }
12    }
13    for(int i = 0;i<n-1;i++)
14        if(k[i+1]<k[i])
15            k[i+1]= k[i];
16    return k;
17 }
18 }
```

```
25 int main(){
26     //www.bilgisayarkavramlari.com
27     int n ;
28     printf("zar sayisini giriniz");
29     scanf("%d",&n);
30     int *k;
31     k=(int *) malloc(sizeof(int)*n);
32     for(int i = 0;i<n;i++){
33         k[i] = 1;
34     }
35     while(k[0]<=6){
36         dizibastir(k,n);
37         k= diziarattir(k,n);
38     }
39     getch();
40 }
41 }
```

C:\Users\sheda\\Desktop\bilgisayarkavramlari\tekrarsizzar.exe

zar sayisini giriniz1

1 2 3 4 5 6

C:\Users\sheda\\Desktop\bilgisayarkavramlari\tekrarsizzar.exe

zar sayisini giriniz2

11 12 13 14 15 16 22 23 24 25 26 33 34 35 36 44 45 46 55 56 66

C:\Users\sheda\\Desktop\bilgisayarkavramlari\tekrarsizzar.exe

zar sayisini giriniz3

111 112 113 114 115 116 122 123 124 125 126 133 134 135 136 144 145 146 155 156
166 222 223 224 225 226 233 234 235 236 244 245 246 255 256 266 333 334 335 336
344 345 346 355 356 366 444 445 446 455 456 466 555 556 566 666 _

C:\Users\sheda\\Desktop\bilgisayarkavramlari\tekrarsizzar.exe

zar sayisini giriniz5

11111 11112 11113 11114 11115 11116 11122 11123 11124 11125 11126 11133 11134 11
135 11136 11144 11145 11146 11155 11156 11166 11222 11223 11224 11225 11226 1123
3 11234 11235 11236 11244 11245 11246 11255 11256 11266 11333 11334 11335 11336
11344 11345 11346 11355 11356 11366 11444 11445 11446 11455 11456 11466 11555 11
556 11566 11666 12222 12223 12224 12225 12226 12233 12234 12235 12236 12244 1224
5 12246 12255 12256 12266 12333 12334 12335 12336 12344 12345 12346 12355 12356
12366 12444 12445 12446 12455 12456 12466 12555 12556 12566 12666 13333 13334 13
335 13336 13344 13345 13346 13355 13356 13366 13444 13445 13446 13455 13456 1346
6 13555 13556 13566 13666 14444 14445 14446 14455 14456 14466 14555 14556 14566
14666 15555 15556 15566 15666 16666 22222 22223 22224 22225 22226 22233 22234 22
235 22236 22244 22245 22246 22255 22256 22266 22333 22334 22335 22336 22344 2234
5 22346 22355 22356 22366 22444 22445 22446 22455 22456 22466 22555 22556 22566
22666 23333 23334 23335 23336 23344 23345 23346 23355 23356 23366 23444 23445 23
446 23455 23456 23466 23555 23556 23566 23666 24444 24445 24446 24455 24456 2446
6 24555 24556 24566 24666 25555 25556 25566 25666 26666 33333 33334 33335 33336
33344 33345 33346 33355 33356 33366 33444 33445 33446 33455 33456 33466 33555 33
556 33566 33666 34444 34445 34446 34455 34456 34466 34555 34556 34566 34666 3555
5 35556 35566 35666 36666 44444 44445 44446 44455 44456 44466 44555 44556 44566
44666 45555 45556 45566 45666 46666 55555 55556 55566 55666 56666 66666

Kutulama Problemi

- Bu problemde amaç bir çizgi veya hat gibi görülebilecek yapının içerisine farklı boyutlardaki çizgileri yerleştirmek olarak düşünülebilir.
- Örneğin zaman çizelgelemesinde, bir kişinin yapacağı işleri, zaman çizgisinin üzerine yerleştirmesi (ve her işin farklı miktarda zaman gerektirdiğini ve kişinin çalışma saatlerinin sınırlı olduğunu düşünürsek en fazla işi en az zamanda (örneğin 8 saatlik mesailer içinde) yapması) bir problemdir. Buradaki hem kişinin yapacağı işler hem de bu işlerin yerleştirileceği zaman çizgisi tek boyutludur.
- Daha basit olması açısından örneğin 100m uzunluğundaki bir ipi 7 ve 9m uzunluğundaki parçalara en az fire ile bölmek istiyoruz, en verimli bölme işleminde kaç adet 7 ve kaç adet 9 uzunluğunda ipimiz olur gibi soruları düşünebiliriz. Bu tip sorular tek boyutlu kutulama problemleridir.

- Şayet problem tek boyutlu ise ve homojen nesnelerin paketlenmesi olarak problemin çözülmesi isteniyorsa problem oldukça basit demektir ve basit matematiksel hesaplamalar ile problemi çözebiliriz.
- Örneğin tek nesne ve tek paket varsa işlem basitçe paketin nesneye bölümü olarak bulunur (zaten burada zor Bir şey de yok):
- Örneğin 100m uzunluğundaki bir ipten kaç tane 5m uzunluğunda ip kesilebilir:
- $100 / 5 = 20$
- biraz daha zorlaştırıp ip sayısını 2 çeşide veya 3 çeşide çıkarırsak problem np-tam (np-complete) bir hal alır. Örneğin aşağıdaki kodu inceleyelim:

```

1  #include <stdio.h>
2
3  int bul(int a,int b,int n,int m,int k){
4      if(k==0){
5          printf("\ncozum : %d*%d + %d*%d = %d",a,n,b,m,k);
6          return 1;
7      }
8      if(k<0)
9          return -1;
10     int p = bul(a,b,n+1,m,k-a);
11     int q = bul(a,b,n,m+1,k-b);
12     if(p>q)
13         return p;
14     return q;
15 }
16 //www.bilgisayarkavramlari.com
17 int main(){
18     int k = 100;
19     int a = 7;
20     int b = 9;
21     while(bul(a,b,0,0,k)<0){
22         printf("optimum cozum yok %d icin cozum deneniyor",--k);
23     }
24     return 0;
25 }

```

```

7*9 + 9*8 = 80
7*9 + 9*2 = 81
7*8 + 9*9 = 82
7*8 + 9*3 = 83
7*12 + 9*0 = 84
7*11 + 9*7 = 85
7*11 + 9*1 = 86
7*10 + 9*8 = 87
7*10 + 9*2 = 88
7*9 + 9*9 = 89
7*9 + 9*3 = 90
7*13 + 9*0 = 91
7*12 + 9*7 = 92
7*12 + 9*1 = 93
7*11 + 9*8 = 94
7*11 + 9*2 = 95
7*10 + 9*9 = 96
7*10 + 9*3 = 97
7*14 + 9*0 = 98
7*13 + 9*7 = 99
7*13 + 9*1 = 100

```

shedaipardus2011 ~ \$

Sonuç

- Bu derste, özellikle son iki haftada anlatılan Dizi ve Dizgi kavramlarını kapsayan örnekler gösterilmiştir. Programların performansını arttırıcı ve hafıza / zaman ikileminde doğru karar vermeyi sağlayıcı problemler özel olarak seçilmiştir. Bu problemlerin çözümü, sadece istenen sonucu veren birer program yazmak olarak düşünülmemeli, bunun ötesinde verimli program yazmaya giriş olarak değerlendirilmelidir.

10. Ders

- Bu derste, programlama dillerindeki temel özelliklerden birisi dizgilere yer verilecektir. Ders kapsamında temel olarak yazı şeklinde girilen, saklanan veya işlenen verileri, bilgisayar üzerinden nasıl işleyebileceğimizi göreceğiz.



Bölüm İçeriği

- Nesne Yönelimli Programlama ve Oluşum
 - C Dilinde Oluşum ve Yapılar
- Dizgiler (Strings)
 - Dizgi Parçalama (String Tokenizer)
 - Dizgilerin Kopyalanması
 - İlkel Tiplerden Dizgi Tipine Dönüş
- String Tokenizer
- Sorular



Oluşum (Composition)

- C dilinde ilkel veri yapılarının (primitive data structures) üzerine kurulan ve kullanıcı tarafından tanımlanabilen yapılar için struct kelimesi kullanılır. Programcı tanımladığı yapıları (struct) daha ileride yeni yapılar (struct) içerisinde de ilkel veri yapısı gibi kullanabilir.
- Aşağıda örnek yapı (struct) kullanımı verilmiştir:
- *typedef struct {*
- *int yas;*
- *char *isim;*
- *enum { bay, bayan } cinsiyet;*
- *} İnsan;*

- Yukarıdaki örnekte bir insan tipi tanımlanmış ve bu tipin özellikleri olarak, yaş, isim ve cinsiyet belirtilmiştir. Artık insan yapısında olan değişkenlerin bu bilgilerine erişilip istenilen veriler atanabilir.
- Örnek:
- *Insan ali;*
- *ali.yas=18;*
- *ali.isim="Ali Yildiz";*
- *ali.cinsiyet=bay;*

- Yukarıda tanımlanan yapı için aynı zamanda pointer (gösterici) kullanmak da mümkündür. Bu durumda:
- *Insan *ali;*
- *ali->yas=18;*
- *ali->isim="Ali Yıldız";*
- *ali->cinsiyet=bay;*



- *void yaslandir(Insan *yaslanacakInsan)*
- {
- *yaslanacakInsan->yas++;*
- }



- Bir yapının içerisinde farklı yapılar da kullanılabilir. Örneğin yazının başlarında geçen öğrencinin insan ve ders bilgileri olması durumunu ele alalım ve eksik olan ders yapısını tanımlayalım:
- *typedef struct {*
- *int kredi;*
- *char *isim;*
- *} Ders;*

- Şimdi yukarıdaki insan ve ders yapılarını birleştirerek öğrenci tanımlayabiliriz:
- *typedef struct {*
- *Ders *alınanDers;*
- *Insan *kisiselBilgi;*
- *Char *bolumu;*
- *} Ogresci;*



- Yukarıdaki yapıda öğrenci tanımlanmış ve bir öğrencinin ders tipinden alinanders'i insan tipinden kisiselbilgileri ve ilkel tipten (char) bolumu tanımlanmıştır. Artık aşağıdaki erişimler mümkündür:
- *Ogrenci ali;*
- *ali.alinanDers->kredi=3;*
- *ali.alinanDers->isim="C ile programlama";*
- *ali.kisiselBilgi->yas=18;*
- *ali.bolumu="Bilgisayar Muhendisligi";*



Dizgiler

- Bir dilde bulunan ve o dilin tanımlı olan alfabeti içerisindeki sembollerin çeşitli sayılarda ve çeşitli sırada dizilmesi ile elde edilen yazılardır.
- Örneğin bir dildeki alfabe aşağıdaki şekilde tanımlı olsun:
- $\Sigma_1 = \{0,1\}$
- Buna göre dilimizde sadece “0” ve “1” sembolleri tanımlı demektir. Bu dilde örneğin $w_1=0$ veya $w_2=10101011010$ gibi bir dizgi elde etmek mümkündür.
- Bir dizginin belirli bir kısmını içeren dizgiye ise alt dizgi adı verilir. Örneğin $w_3=1011$ dizgisi w_2 dizgisinin bir altdizgisidir.
- Ayrıca iki dizginin arka arkaya eklenmesine de üleştirme(concatenation) denilir.Örneğin w_1 ile w_3 dizgilerinin üleştirilmiş hali $w_4=01011$ olur.

Dizgiler

```
1  #include <stdio.h>
2  #include <conio.h>
3  //www.bilgisayarkavramlari.com
4  int main(){
5      char isim[100];
6      printf("isminizi giriniz:");
7      scanf("%s",isim);
8      printf("\nisminiz: %s",isim);
9      getch();
10 }
```

C:\Users\sheda\\Desktop\bilgisayarkavramlari\strscanf.exe

isminizi giriniz:ali

isminiz: ali_

C:\Users\sheda\\Desktop\bilgisayarkavramlari\strscanf.exe

isminizi giriniz:ali baba ve kirk haramiler
isminiz: ali

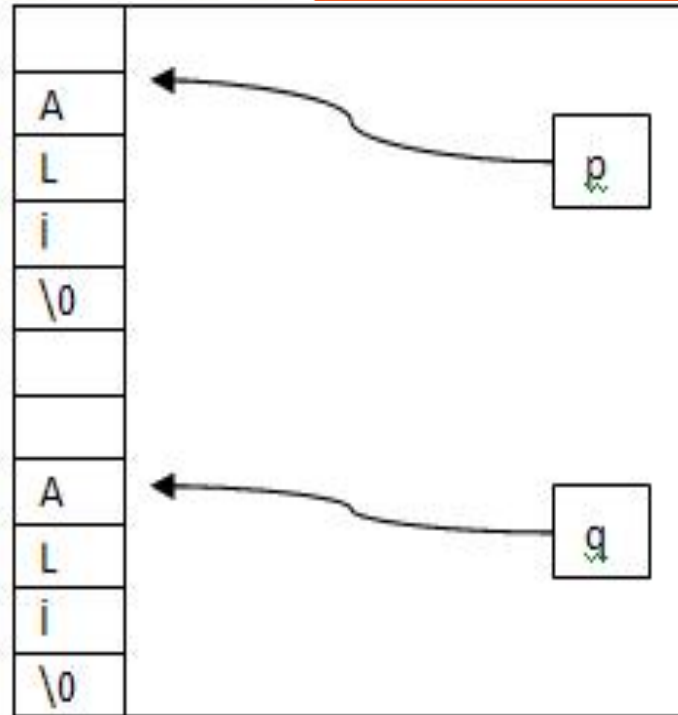
```
1  #include <stdio.h>
2  #include <conio.h>
3  //www.bilgisayarkavramlari.com
4  int main(){
5      char isim[100];
6      printf("isminizi giriniz:");
7      gets(isim);
8      printf("\nisminiz: %s",isim);
9      getch();
10 }
```

C:\Users\shedai\Desktop\bilgisayarkavramlari\strscanf.exe

isminizi giriniz:ali baba ve kirk haramiler
isminiz: ali baba ve kirk haramiler

Dizgilerde Eşitlik

- `int esitmi=1;`
- `int i;`
- `for(i = 0;p[i]!='\0';i++){`
- `if(p[i]!=q[i]){`
- `esitmi=0;`
- `break;`
- `}`
- `}`
- `if(esitmi==1&&q[i]=='\0'){`
- `printf("esitler");`
- `}`
- `else{`
- `printf("esit degiller");`
- `}`



- *if(!strcmp(p,q)){*
- *printf("eşitler");*
- *}*
- *else {*
- *printf("eşit değiller");*
- *}*



Dizgi Parçalama

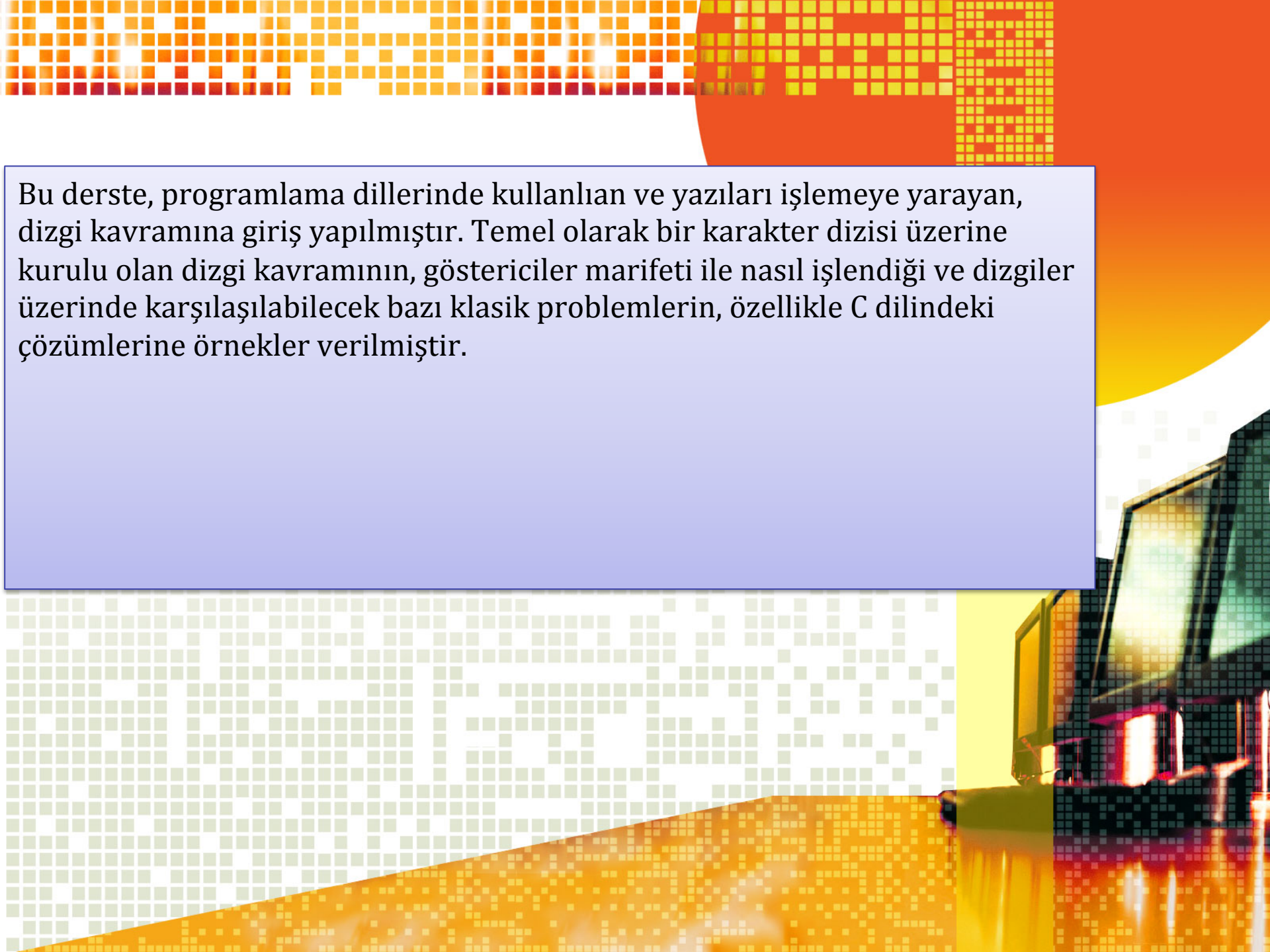
- */* strtok ornegi */*
- *#include <stdio.h>*
- *#include <string.h>*
- *int main ()*
- *{*
- *printf ("Girilen \ "%s\ " mesajini parcaliyoruz:\n",str);*
- *pch = strtok (str, " ,.-");*
- *// parcalama icin kullanilan ayiricilar (delimiters)*
- *while (pch != NULL)*
- *{*
- *printf ("%s\n",pch);*
- *pch = strtok (NULL, " ,.-");*
- *}*
- *return 0;*
- *}*

Dizgilerin Kopyalanması

- `char *a = "sadi evren seker";`
- `char b[200];`
- `strcpy(b,a);`
- Burada üzerine kopyalanan ilk, içinden dizginin okunduğu ikinci sırada verilmelidir.
- Örnek kod aşağıda verilmiştir:
- `#include <stdio.h>`
- `#include <string.h>`
- `int main ()`
- `{`
- `char str1[]="Benim adim sadi";`
- `char str2[40];`
- `char str3[40];`
- `strcpy (str2,str1);`
- `strcpy (str3,"basriyla kopyalandi");`
- `printf ("str1: %s\nstr2: %s\nstr3: %s\n",str1,str2,str3);`
- `return 0;`
- `}`

Dizgiye Dönüştürme

- *int x = atoi ("123");*
- satırından sonra x değişkeninin içinde sayısal olarak 123 değeri bulunur.
- Örnek kod aşağıda verilmiştir:
- *#include <stdio.h>*
- *#include <stdlib.h>*
- *int main ()*
- *{*
- *int i;*
- *char szInput [256];*
- *printf ("Bir sayi giriniz: ");*
- *fgets (szInput, 256, stdin);*
- *i = atoi (szInput);*
- *printf ("girilen sayi %d. iki misli %d.\n",i,i*2);*
- *return 0;*
- *}*



Bu derste, programlama dillerinde kullanılan ve yazıları işlemeye yarayan, dizgi kavramına giriş yapılmıştır. Temel olarak bir karakter dizisi üzerine kurulu olan dizgi kavramının, göstericiler marifeti ile nasıl işlendiği ve dizgiler üzerinde karşılaşılabilecek bazı klasik problemlerin, özellikle C dilindeki çözümlerine örnekler verilmiştir.

8. Ders

Bu derste, programlama dillerindeki temel özelliklerden birisi olan gösterici (pointer) kavramına giriş yapılacaktır ve örnek kodlar üzerinden konu açıklanacaktır. Temel olarak bir gösterici (pointer), hafızadaki bir alanın, programcı tarafından kontrol edilmesini sağlar.



Bölüm İçeriği

- Göstericiler ve Nesne Atıfları
 - C Dilinde Göstericiler
 - Atıf ile Çağırma
- Sorular



Tanım

Göstericiler (Pointers) hafızadaki herhangi bir adresi ve dolayısıyla bu adres içerisindeki veriyi gösteren yapılardır. Gelişmiş programlama dillerinin neredeyse tamamından gösterici kavramı ya doğrudan ya da nesne atfı (object referrer) olarak bulunur. Örneğin JAVA dilinde doğrudan bir gösterici yapısı bulunmaz ama bunun yerine bir nesne atfı yapısı bulunur ve hafızadaki nesneler bu sayede gösterilebilir.



Örnek

- *int *p;*
- *int a=10;*
- *p=&a;*

Adres değeri	içeriği	matiksal ismi
A101		
A102		
A103		
A104		
A105		
A106		
A107	A116	p
A108		
A109		
A110		
A111		
A112		
A113		
A114		
A115		
A116	10	a
A117		
A118		
A119		
A120		

Örnek

- *#include <stdio.h>*
- *#include <conio.h>*
- *int main(){*
- *int a=10;*
- *int *p;*
- *p=&a;*
- *printf("%d\n",*p); // p'nin gösterdiği yeri basar*
- *printf("%d\n",p); // p'nin değerini yani, p'nin*
gösterdiği yerin adresini basar
- *printf("%d\n",a); // a'nın değerini basar*
- *printf("%d\n",&a); //a'nın adresini basar*
- *printf("%d\n",&p); //p'nin adresini basar*
- *getch();*
- *return 0;*
- *}*

Gösterici(pointer) – Dizi (array)

- her dizi bir pointer her pointer da doğal bir dizidir.
- `int d[80], *p1;`
`p1 = d;`
- Burada `p1`, `d` dizisinin ilk elemanının adresinin değerini alır. Yani dizi adı, aslında o dizinin hafızadaki başlangıç adresini tutmaktadır. `d` dizisinin 5. Elemanına erişmek için ise:
- `d[4]` veya `*(p1+4)`
- ifadelerini kullanabiliriz. Her ikisinin de anlamı aynıdır.

Dinamik Hafıza Kullanımı

- diziler ile ayrılan yer miktarı sabittir.
- Buradaki sabitlik iki türlü anlaşılabilir. C dili için kodlama sırasında henüz kod çalışmadan bu yerin miktarı belirli olmalıdır. Yani aşağıdaki gibi bir tanım C dilinde hatalıdır:
 - `int a=10;`
 - `int d[a];`
 - Yukarıdaki kodda d dizisinin boyutu bir değişkenden atanmaktadır ve C99 standardında bu hatadır.
 - `int *p = (int *) malloc (sizeof(int)*10);`

Atıf ile Çağırma

- Programlama dillerinde, [fonksiyon](#) çağırma işlemi sırasında kullanılan yöntemlerden birisi de atıf ile çağırma (call by reference) yöntemidir. Genelde bir programlama dilinin standart çağırma yöntemi değer ile çağırmadır (call by value) ancak gösterici (pointer) desteği olan dillerde bu standart çağırma yöntemine ilave olarak atıf ile çağırmak da mümkündür.
- Bu yöntemde fonksiyona parametre (argüman) olarak bir değer geçirmek yerine bir gösterici (pointer) referansı geçirilir. Dolayısıyla fonksiyonun içindeki bir yerel değişken (local variable), fonksiyonun çağrıldığı yerdeki bir değeri göstermiş olur. Bu sayede fonksiyonda bu gösterici marifetiyle yapılan bütün işlemler fonksiyonun çağrılması sırasında parametre olarak verilen değer üzerine de etkili olur.



Atıf ile çağırma (Call by reference)

- *#include <stdio.h>*
- *void f(int *p){*
- **p=20;*
- *}*
- *int main(){*
- *int a=10;*
- *f(&a);*
- *printf("%d",a);*
- *}*



Örnek

- Bir dizinin içindeki sayıların ortalamasını, toplamını ve en büyüğünü ayrı ayrı döndüren bir fonksiyon yazınız.



Örnek - devam

- Bu sorunun çözümünde 3 farklı değerin döndürülmesi ve bir dizinin parametre olarak alınması istenmektedir.
- Klasik bir fonksiyon tek bir değer döndürür. Ayrıca diziye parametre olarak alamaz.
- Bu problemin çözümü için atıf ile çağırma kullanılmalıdır.



Çözüm

```
1.  #include <stdio.h>
2.  #include <conio.h>
3.  #include <stdlib.h>
4.  void f(int *a,int boyut,int *toplaml,int *ortalama, int *ebuyuk){
5.      int t=0,eb=a[0];
6.      for(int i = 0;i<boyut;i++){
7.          if(a[i]>eb)
8.              eb=a[i];
9.          t+=a[i];
10.     }
11.     *toplaml = t;
12.     *ortalama = t/boyut;
13.     *ebuyuk = eb;
14. }
15. int main(){
16.     int a[5]={3,8,2,5,1};
17.     int *toplaml=(int*)malloc(sizeof(int));
18.     int *ortalama=(int*)malloc(sizeof(int));
19.     int *ebuyuk=(int*)malloc(sizeof(int));
20.     f(a,5,toplaml,ortalama,ebuyuk);
21.     printf("toplaml : %d, ortalama : %d, enbuyuk : %d",
22.         *toplaml,*ortalama,*ebuyuk);
23. }
```


Enum

- Bilgisayar bilimlerinde alınabilecek alternatiflerin sayılması ve bu sayılan ihtimaller dışındaki ihtimallerin kabul edilmemesi durumudur (ihtimallerin tâtât edilmesi)
- Örneğin programlama dillerinde bir değişkenin alabileceği değerleri tanımlayarak bu değişkene sadece bu değerlerden birisinin konulması sağlanabilir.



Enum

```
enum gunler{  
    pts,sal,car,per,cum,cts,paz  
}gun;  
main(){  
    gun=sal;  
    printf("\n%d",gun); //2. gün yani 1  
    gun=car; // sonraki gun yani car yani 2  
    printf("\n%d",gun);  
    getch();  
}
```



Bu derste, gösterici kavramına giriş yapılmış ve hafızadaki herhangi bir alana doğrudan erişilmesini sağlayan göstericiler hakkında örnekler verilmiştir. Ayrıca geçen hafta anlatılan dizi konusunun da bir gösterici olarak kabul edilebileceği anlatılarak, dizilerin göstericiler ile nasıl kontrol edilebildiği gösterilmiştir.

SONUÇ



Diziler

Bu derste, programlama dillerindeki temel veri ünitelerinden birisi olan dizilere giriş yapılacaktır. Diziler hafızada (RAM) ardışık olarak aynı tipteki veriyi tutmak için tasarlanmıştır. Programımızda kullanılan değişken sayısının artması veya matematiksel olarak kullanılan veriler arasında bağlantı bulunması halinde diziler kullanılabilir.



Bölüm İçeriği

- Diziler
 - Çok Boyutlu Dizilere Giriş
 - Örnekler
- Çok Boyutlu Diziler
 - Çok Boyutlu Dizilerin Tanımlanması
 - Çok Boyutlu Dizilerin Kullanılması
 - İki'den Çok Boyutlu Diziler




```
int a[10]; /* 10 elemanlık bir dizi tanımla */  
for(int i = 0 ; i < 10 ; i++) /* 0'dan 10'a kadar  
dönen bir döngü oluştur */  
a[i]=i; /* a dizisinin i. elemanına i değerini koy  
*/
```

```
a[0] = 0;
```

```
a[1] = 1;
```

```
printf("%d",a[4]);
```

- hafızadaki görüntüsü aşağıdaki şekildedir:
- 0 1 2 3 4 5 6 7 8 9

- `int a[] = { 2, 3, 9, 8, 15, -6};`
- Yukarıdaki bu tanımlama sonucunda dizinin indisleri ile birlikte görüntüsü şu şekildedir:

Veri	2	3	9	8	15	-6
İndis	0	1	2	3	4	5

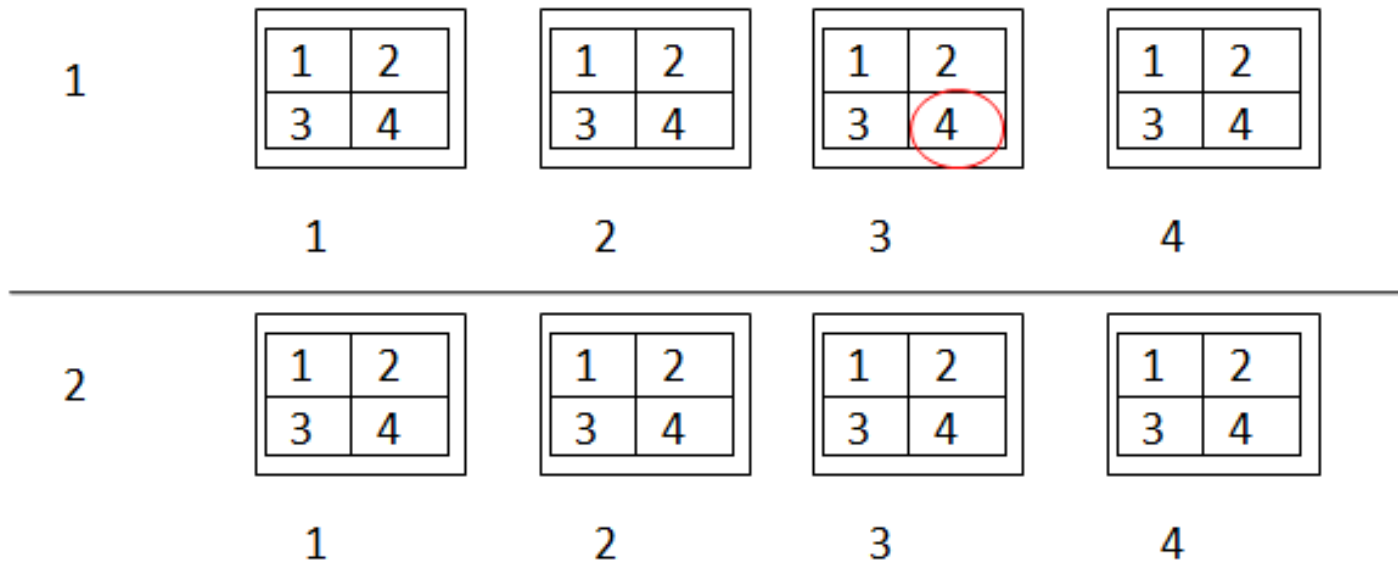
- `a[3] = 9;`
- `a[5] = -6;`

Çok Boyutlu Diziler

- Temel olarak 2 boyutlu bir diziyi bir matris olarak düşünmek mümkündür. Örneğin aşağıdaki matrisi ele alalım:
- $\begin{matrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{matrix}$
- Bu matrisi hafızada tutmak için iki farklı boyuttaki boyut değerleri tanımlanmalıdır. Yani A isminde bir dizi için:
- `int A[2][3]`
- olarak düşünmek mümkündür. Bu diziye ilk değer atamak için:
- `int A[2][3] = { {1, 2, 3}, {4, 5, 6} };`

2'den Çok Boyutlu Diziler

- `int a[2][4][2][2]`
- `a[0][2][1][1] = 4`



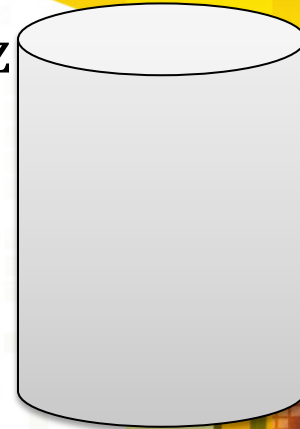
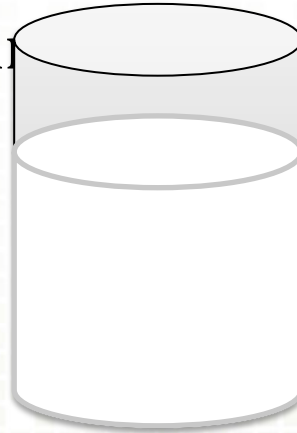
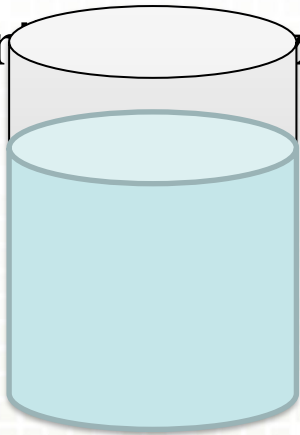
3 | 8 | 7 | 2 | 6

0 | 1 | 2 | 3 | 4

6 | 2 | 7 | 8 | 3

$a[0] = a[4]$

- Verilen arrayi tersine kod yazınız



swapping

```
1. #include <stdio.h>
2. #include <conio.h>
3. int main(){
4.     int a[] = { 3, 8 , 7, 2 , 6};
5.     for ( int i =0 ;i<5;i++){
6.         printf("a[%d] = %d \n",i,a[i]);
7.     }
8.     int gecici;
9.     for(int i =0;i<2;i++){
10.         gecici = a[i];
11.         a[i] = a[4-i];
12.         a[4-i] = gecici;
13.     }
14.
15.     for ( int i =0 ;i<5;i++){
16.         printf("a[%d] = %d \n",i,a[i]);
17.     }
18. getch();
19. }
```



- İlk 20 fibonacci sayısını hesaplayarak diziye yerleştiren (dizinin boyutu 20'lik olacaktır) program yazınız. (fibonacci sayıları 1 1 2 3 5 8 13 ... şeklinde gider ve her sayı kendinden önceki iki **sayının toplamıdır**)



- Dizileri kullanarak sayıları hafızada tutmak mümkündür. Buna göre fibonacci sayılarını üretirken dizinin içerisindeki sayılar kullanılabilir.
- Sayı serisinin ilk iki sayısını elle vermek gerekir. Bu sayılar serinin tanımında da geçmektedir. Dolayısıyla $a[0]=1$ ve $a[1]=1$ atamalarını elle yaptıktan sonra dizinin geri kalanı için döngü içerisinde atama yapılabilir. Bu atama sırasında kullanılacak formülde bulunan sayı, kendinden iki önceki ve bir önceki sayıların toplamı olacaktır. Bunu hesaplamak için örneğin “i” isimli bir döngü değişkeni ile
- $a[i] = a[i-1] + a[i-2]$
- şeklinde kod yazılabilir. Döngü ilk iki sayı belirli olduğu için ikinci sayıdan başlanacak ve istenilen sayıya kadar (bu soruda 20) ilerleyecektir.

Fibonacci Sayılarının Çözümü

```
1. #include <stdio.h>
2. int main(){
3.     int a[20];
4.     a[0] = 1;
5.     a[1] = 1;
6.     for(int i = 2; i < 20; i++){
7.         a[i] = a[i-1] + a[i-2];
8.     }
9.     return 0;
10.}
```



- Aşağıdaki şıklar için ayrı ayrı veya tek bir kod yazınız.
- a)Kullanıcıdan bir sınıfta bulunan öğrenci notlarını okutan bir program yazınız. Öğrenci sayısını kolay olsun diye 10 kabul edebilirsiniz. Buna göre kullanıcı 10 öğrenci için klavyeden not girecektir.
- b) a' kısmında yapılan programa ilave olarak notların ortalamasını, en büyüğünü ve en küçüğünü hesaplatınız.
- c) kullanıcıdan bir not okuyarak yukarıda girilen notlardan birisi olup olmadığını bulunuz. Ekran kullanıcıdan daha önceden girdiği bir not ise “not zaten girilmiş”, girmedeği bir not ise, “bu not daha önce girilmemiş” yazınız.

- `#include <stdio.h>`
- `int main(){`
- `int a[10];`
- `for(int i = 0;i<10;i++){`
- `printf("ogrenci %d notunu giriniz",i);`
- `scanf("%d",&a[i]);`
- `}`
- `int enbuyuk=0,enkucuk=100,toplam=0;`
- `for(i = 0;i<10;i++){`
- `if(a[i]>enbuyuk)`
- `enbuyuk = a[i];`
- `if(a[i]<enkucuk)`
- `enkucuk = a[i];`
- `toplam += a[i];`
- `}`
- `int ortalama=toplam/10;`
- `printf("en buyuk : %d en kucuk %d ortalama %d",enbuyuk,enkucuk,ortalama);`
- `printf("bir not daha giriniz");`
- `int yeninot;`
- `scanf("%d",¥inot);`
- `int x=0;`
- `for(i = 0;i<10;i++){`
- `if(yeninot==a[i]){`
- `x=1;`
- `printf("not zaten girilmis");`
- `break;`
- `}`
- `}`
- `if(x==0){`
- `printf("not girilmemis");`
- `}`
- `}`



Sonu

Bu derste, programlama dillerinde kullanılan dizi kavramına giriř yapılmıřtır. Bir programlama dilinde, birden fazla deęiřkenin, ardışık olarak hafızada tutulmasını saęlayan dizi kavramı, özel olarak C dilindeki örnekler üzerinden anlatılmıřtır.



6.2. Ders - Uygulamalar

- Bu derste, şimdiye kadar gördüğümüz konuları içeren örnek programlar yazmaya çalışacağız. Soru/Cevap şeklinde içeriğe sahip olan bu bölümde, ilk haftadan beri gördüğümüz konuları içeren sorular ve nasıl çözülebileceğini anlatan cevapları bulunacaktır.



Örnek

- Kullanıcıdan bir sayı okuyarak, ekrana, aşağıdaki şekilde, verilen boyutlarda bir kare matrisin bütün elemanları 0 ve sadece ortadaki elemanları 1 olan bir artı yazdırınız. (boyut çift sayı ise, artı iki sütun ve satırdan, tek sayı ise tek satır ve sütundan oluşabilir)**

Örneğin n=5 için

00100

00100

11111

00100

00100

n=6 için

001100

001100

111111

111111

001100

001100



Click to add title

```
1 #include <stdio.h>
2 #include <conio.h>
3
4 int main(){
5     int n;
6     printf("bir sayi giriniz");
7     scanf("%d",&n);
8
9     for(int i = 0;i<n;i++){
10         for(int j = 0;j<n;j++){
11             if((i==n/2 || j==n/2) || (n%2==0&&(j==n/2-1 || i==n/2-1 )))
12                 printf("1");
13             else
14                 printf("0");
15         }
16         printf("\n");
17     }
18     getch();
19 }
```

- **Parametre olarak bir sayı alan, aldığı bu sayıdan küçük, en büyük asal sayıyı döndüren fonksiyonu yazınız.**



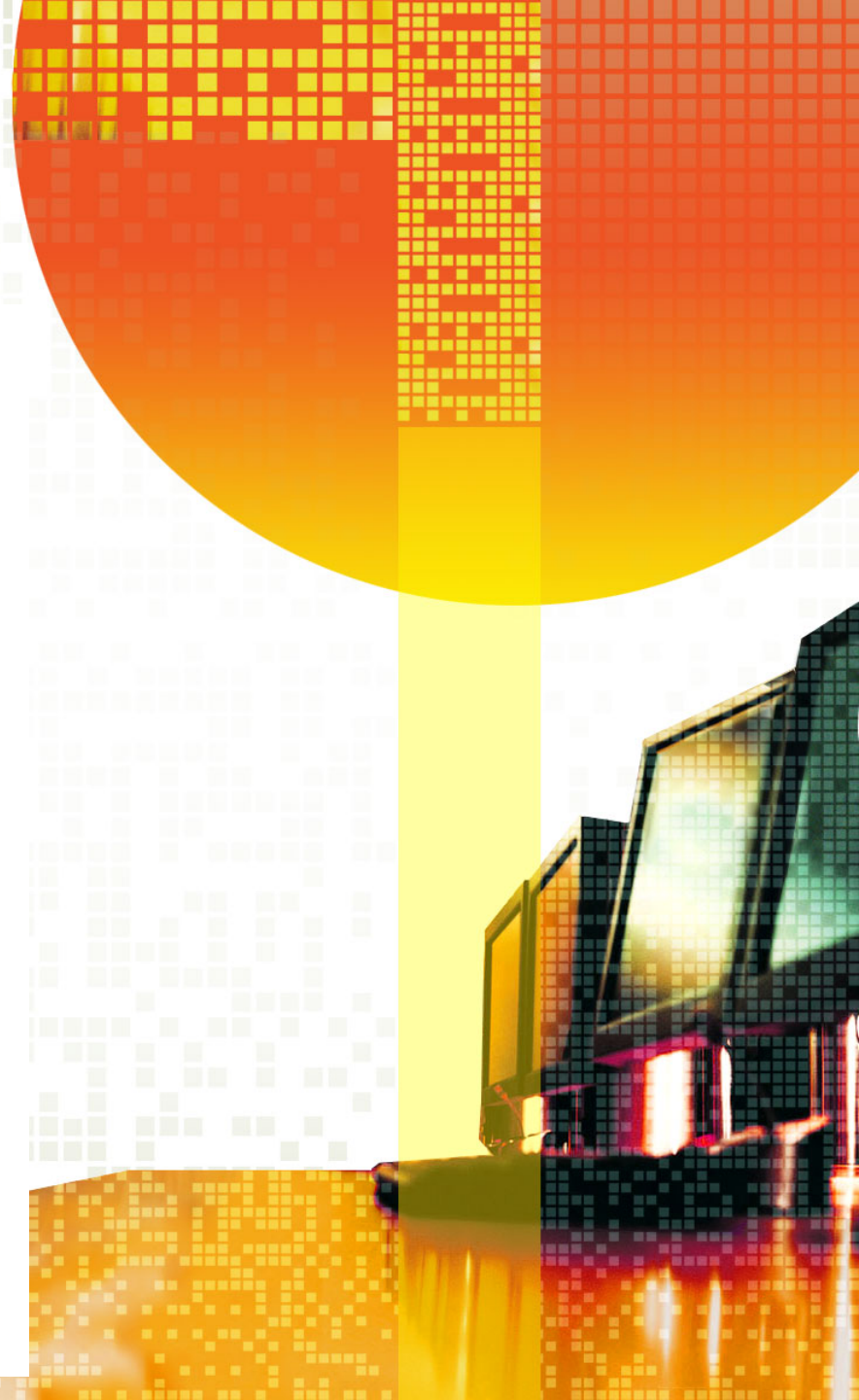
Çözüm

```
1 #include <stdio.h>
2 #include <conio.h>
3
4 int main(){
5     printf("bir sayi giriniz");
6     int n ;
7     scanf("%d",&n);
8     for(int i = n;i>=2;i--){
9         int asal =1;
10        for(int j=2;j<i;j++){
11            if(i%j==0)
12                asal = 0;
13        }
14        if(asal==1){
15            printf("%d",i);
16            break;
17        }
18    }
19    getch();
20 }
```



Özyineli Çözüm

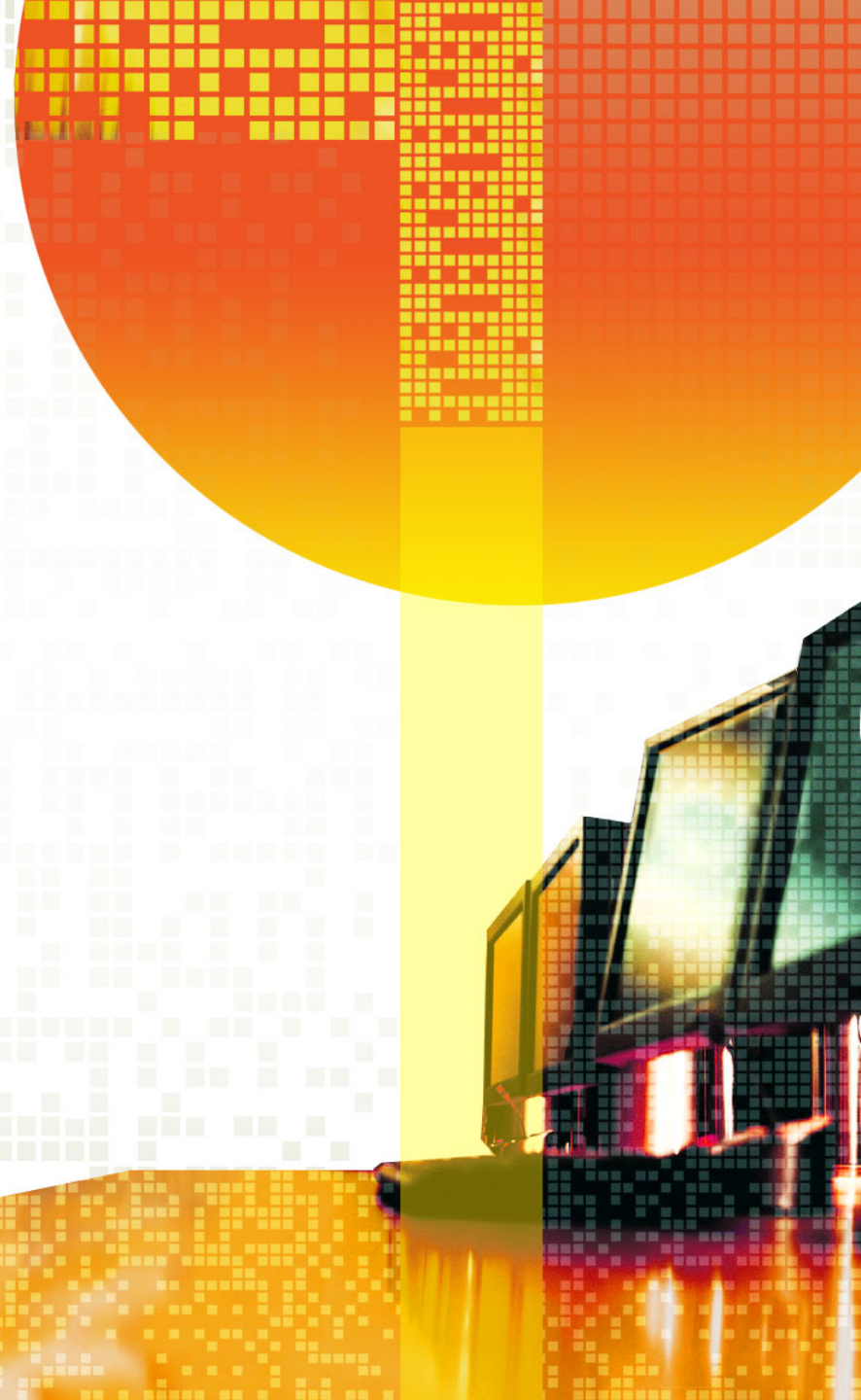
```
1  #include <stdio.h>
2  #include <conio.h>
3  int f(int n,int test){
4      if (n-1==test)
5          return 1;
6      if(n%test ==0)
7          return 0;
8
9      return f(n,test+1);
10 }
11 int g(int n){
12     if(f(n,2))
13         return n;
14
15     return g(n-1);
16 }
17
18
19 int main(){
20     printf("bir sayi giriniz");
21     int n;
22     scanf("%d",&n);
23     printf("%d",g(n));
24     getch();
25 }
```



- **Kullanıcıdan bir sayı okuyarak, okunan sayının 2 tabanında logaritmasını yaklaşık tam sayı olarak bastırınız. (örneğin 30 sayısının 2 tabanındaki logaritması, 30 sayısı 16 ve 32 sayıları arasında olduğu için 4 veya 5 olarak bulunabilir)**

2, 4, 8 , 16 , 32 , 64, 128 , 256, 512
1, 2, 3 , 4, 5 , 6


```
1 #include <stdio.h>
2 #include <conio.h>
3
4 int main(){
5     int n;
6     printf("bir sayi giriniz");
7     scanf("%d",&n);
8     int i ;
9     int c=0;
10    for( i = 1;i<n;i=i*2){
11        c++;
12    }
13    printf("%d",c);
14    getch();
15 }
```



- **Kullanıcıdan yaşını okuyan ve okunan yaşı ekrana basan kod yazınız. Bu soru basit bir okuma / yazma sorusudur.**



```
1 #include <stdio.h>
2 #include <conio.h>
3
4 int main(){
5     printf("Yasinizi Giriniz");
6     int yas;
7     scanf("%d",&yas);
8     printf("Yasiniz : %d",yas);
9     getch();
10 }
```



- **Sınıfta bulunan öğrencilerin yaşlarını okuyan ve ortalamasını alan bir kod yazmamız isteniyor. Soruda, ayrıca -1 girilene kadar öğrenci okumaya devam edeceğimiz belirtiliyor.**



```
1 #include <stdio.h>
2 #include <conio.h>
3
4 int main(){
5     int yas=0;
6     int toplam = 0;
7     int say = 0;
8     while(yas>=0){
9         toplam = toplam + yas;
10        say ++;
11        printf("siradaki ogrencinin yasini giriniz");
12        scanf("%d",&yas);
13    }
14    say --;
15    printf("\nortalama yas : %f",(float)toplam / (float) say);
16    getch();
17
18 }
```

- **Fibonacci serisi iki boyutlu olarak istenmiş. Bu sorunun örneğin 5 için sonucu aşağıdaki şekilde olacaktır.**

1	1	2	3	5
1	2	3	5	8
2	3	5	8	13
3	5	8	13	21
5	8	13	21	34

- Yukarıdaki tabloda görüldüğü üzere, iki boyutlu fibonacci matrisinin özelliği, sol üst köşeden başlamak üzere, sağa ve aşağı doğru izlenen bütün yollarda fibonacci serisi vermesidir.


```
1 #include <stdio.h>
2 #include <conio.h>
3
4 int main(){
5     int n;
6     printf("matrisin boyutunu giriniz");
7     scanf("%d",&n);
8     int a=1;
9     int b=1;
10    for(int i = 0;i<n;i++){
11        int p=a;
12        int q=b;
13        for(int j = 0;j<n;j++){
14            int r = p+q;
15            printf("%d ",p);
16            p=q;
17            q=r;
18        }
19        printf("\n");
20        int c= a+b;
21        a=b;
22        b=c;
23    }
24    getch();
25 }
```

- Verilen bir sayı kadar asal sayıyı basmamız istenmiş. Örneğin 5 girilmesi durumunda, ilk 5 asal sayıyı basan kod isteniyor ki bu sayılar 2 3 5 7 11 şeklindedir



```
1 #include <stdio.h>
2 #include <conio.h>
3
4 int main(){
5     int k;
6     printf("kac asal sayi istiyorsunuz");
7     scanf("%d",&k);
8     int sayac = 0;
9     int n = 2;
10    while(sayac < k){
11        int asal = 1;
12        for(int i = 2;i<n;i++){
13            if(n%i==0)
14                asal = 0;
15        }
16        if(asal == 1){
17            printf("%d ",n);
18            sayac ++;
19        }
20        n++;
21    }
22    getch();
23 }
```



Sonuç

- Bu derste, programlama dillerinde kullanılan fonksiyon kavramına giriş yapılmıştır. Yapısal programlama dillerinin temel üç özelliğinden birisi olan ve bir program bloğunun parametrik olarak farklı değerler için çalışabilmesini sağlayan fonksiyon kavramı, C dilindeki örnekler üzerinden anlatılmıştır. Ayrıca kendi kendini çağırma özelliği bulunan özyineli fonksiyonlar anlatılmış ve örnekler verilmiştir.



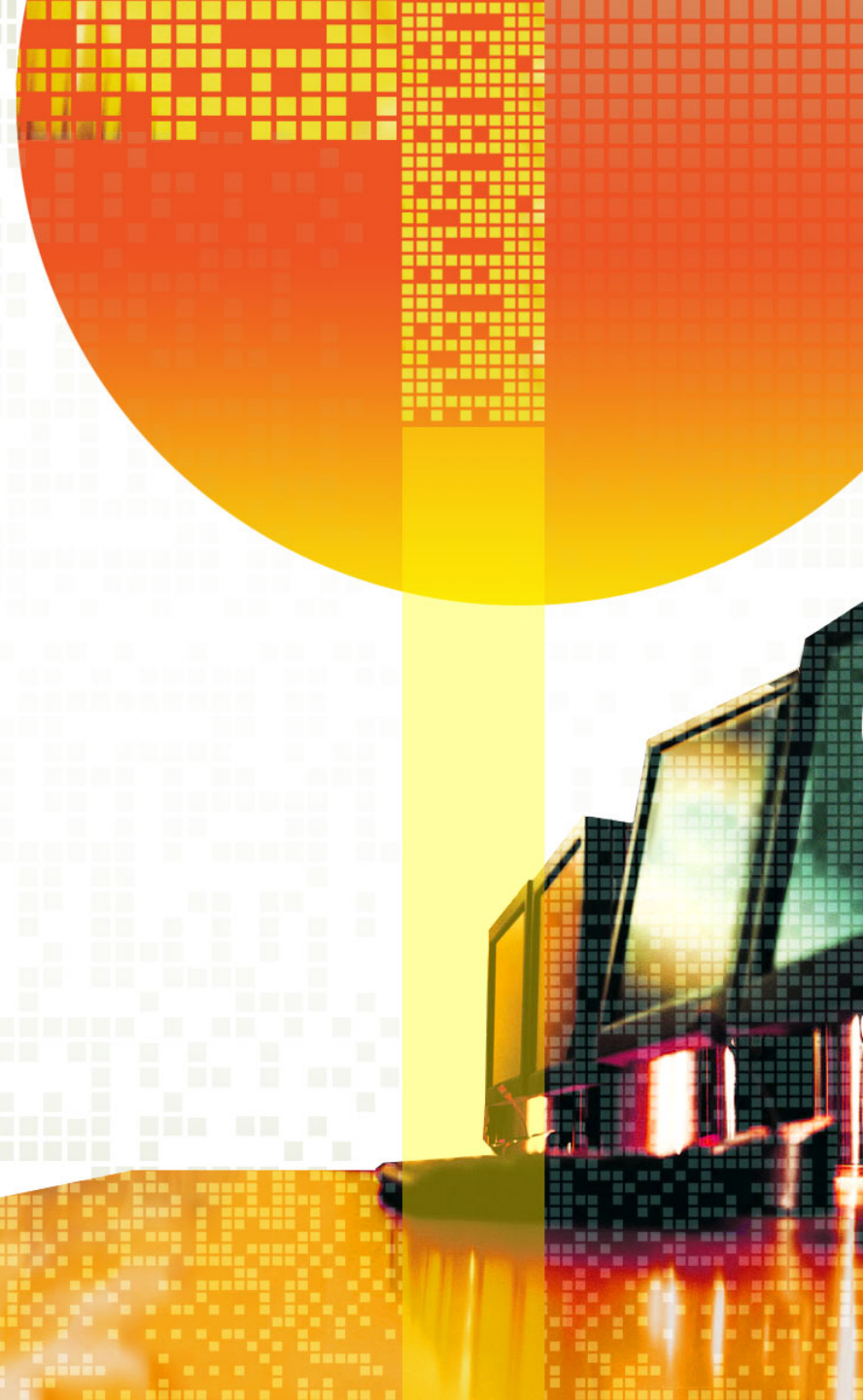
6. Hafta

Bu derste, Őimdiye kadar g rd ğ m z konuları i eren  rnek programlar yazmaya  alıŐacağ z. D ng ler ve i  i e d ng lere ait programları yazarken izlenmesi gereken y ntemler anlatılarak  rnek programlar  zerinden nasıl bir yol takip edilmesi gerektiğ  g sterilecektir.



Bölüm İçeriği

- Basit Döngüler
- İç İççe Döngüler



Örnek

```
1  #include <stdio.h >
2  #include <conio.h>
3  int main(){
4      for(int i = 1;i<=10;i++){
5          printf("sadi evrenseker");
6      }
7      getch();
8  }
```

C:\Users\shedai\Desktop\www.bilgisayarkavramlari.com\Unti4tled1.exe

sadi evrensekersadi evrensekersadi evrensekersadi evrensekersadi evrensekersadi evrensekersadi evrens
ekersadi evrensekersadi evrensekersadi evrensekersadi evrensekersadi evrenseker

Örnek

```
1  #include <stdio.h >
2  #include <conio.h>
3  int main(){
4      for(int i = 1;i<=10;i++){
5          printf("%d\n",i);
6      }
7      getch();
8  }
```

C:\Users\shedai\Desktop\www.bilgisayarkavramlari.com\Untitled21.exe

```
1
2
3
4
5
6
7
8
9
10
```

```
1  #include <stdio.h >
2  #include <conio.h>
3  int main(){
4      for(int i = 1;i<=10;i++){
5          printf("%d\n",i);
6      }
7      getch();
8  }
```

C:\Users\shedai\Desktop\www.bilgisayarkavramlari.com\Untitled211.exe



C:\Users\shedai\Desktop\www.bilgisayarkavramlari.com\Untitled1.exe

99 96 93 90 87 84 81 78 75 72 69 66 63 60 57 54 51

```
1
2 #include <stdio.h>
3 int main(){
4     for(int i = 100; i >= 50; i--){
5         if(i % 3 == 0){
6             printf("%d ", i);
7         }
8     }
9     getch();
10 }
```

- 1 -> 99 = $100 - 1 = 100 - 1$
- 2 -> 96 = $100 - 4 = 100 - 1 - 3$
- 3 -> 93 = $100 - 7 = 100 - 1 - 6$
- 4 -> 90 = $100 - 10 = 100 - 1 - 9$

```
1 #include <stdio.h >
2 #include <conio.h>
3 int main(){
4     int i = 100;
5     while(i>=50){
6         if(i%3==0){
7             printf("%d ",i);
8         }
```

```
1 #include <stdio.h >
2 #include <conio.h>
3 int main(){
4     for(int i = 1;i<=10;i++){
5         printf("%d %d %d\n",i*5,100-((i-1)*10+1),i);
6     }
7     getch();
8 }
```

C:\Users\shedai\Desktop\www.bilgisayarkavramlari.com\Untitled3.exe

```
5 99 1
10 89 2
15 79 3
20 69 4
25 59 5
30 49 6
35 39 7
40 29 8
45 19 9
50 9 10
```



```

1  #include <stdio.h >
2  #include <conio.h>
3  int main(){
4      int num;
5      printf("faktöriyelinin hesaplanmasını istediğiniz sayıyı giriniz");
6      scanf("%d",&num);
7      int fact = 1;
8      for(int i = 1;i<=num;i++){
9          fact = fact*i;
10     }
11     printf("%d",fact);
12     getch();
13 }

```

num	i	fact
6	1	1
6	2	2
6	3	6
6	4	24
6	5	120
6	6	720

C:\Users\shedai\Desktop\www.bilgisayarkavramlari.com\Untitled4.exe

faktöriyelinin hesaplanmasını istediğiniz sayıyı giriniz6
720

```
1  #include <stdio.h >
2  #include <conio.h>
3  int main(){
4      int top = 0;
5      for(int i = 1;i<=10;i++){
6          printf("bir sayı giriniz");
7          int sayi;
8          scanf("%d",&sayi);
9          top = top + sayi;
10     }
11     printf("Sayıların toplamı : %d \n",top);
12     printf("ortalaması : %d",top/10);
13     getch();
14 }
```

C:\Users\shedai\Desktop\www.bilgisayarkavramlari.com\Untitled5.exe

```
bir sayı giriniz5
bir sayı giriniz4
bir sayı giriniz3
bir sayı giriniz2
bir sayı giriniz8
bir sayı giriniz5
bir sayı giriniz4
bir sayı giriniz3
bir sayı giriniz9
bir sayı giriniz2
Sayıların toplamı : 45
ortalaması : 4_
```

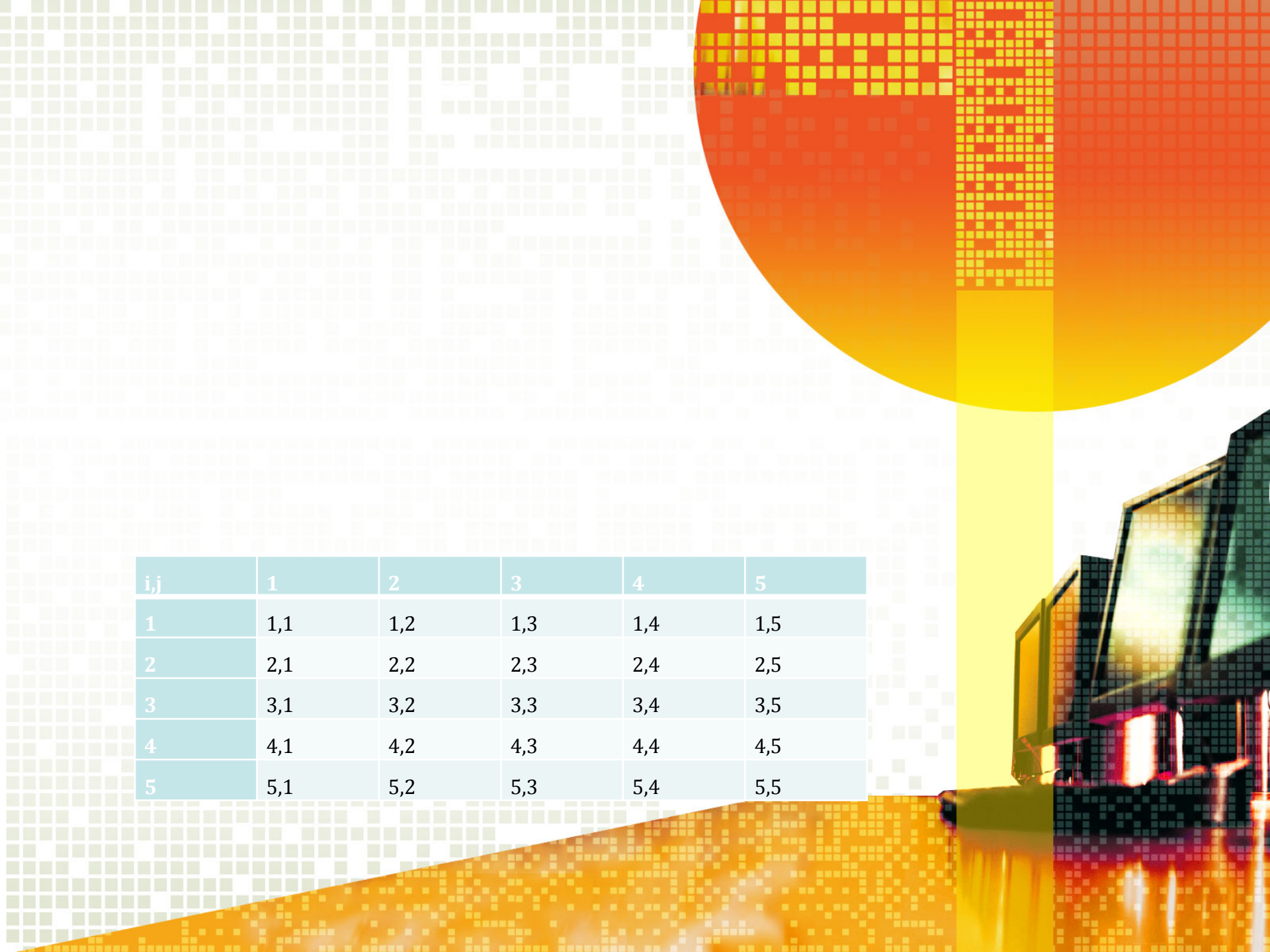
```
1  #include <stdio.h >
2  #include <conio.h>
3  int main(){
4      int top = 0;
5      int say = 0;
6      int sayi=0;
7      while(sayi!= -1){
8          printf("bir sayı giriniz");
9
10         scanf("%d",&sayi);
11         /* if(sayi==0)
12             continue;*/
13         if(sayi== -1)
14             break;
15         top = top + sayi;
16         say ++;
17         printf("top : %d sayi: %d say %d\n",top,sayi,say);
18     }
19
20     printf("toplama: %d \n",top);
21     printf("ortalama: %d",top/say);
22     getch();
23 }
```



```
1  #include <stdio.h >
2  #include <conio.h>
3  int main(){
4      for(int i = 1;i<=10;i++){
5          for(int j=1;j<=10;j++){
6              printf("%d\t",i*j);
7          }
8          printf("\n");
9      }
10     getch();
11 }
```

C:\Users\shedai\Desktop\www.bilgisayarkavramlari.com\Untitled7.exe

1	2	3	4	5	6	7	8	9	10
2	4	6	8	10	12	14	16	18	20
3	6	9	12	15	18	21	24	27	30
4	8	12	16	20	24	28	32	36	40
5	10	15	20	25	30	35	40	45	50
6	12	18	24	30	36	42	48	54	60
7	14	21	28	35	42	49	56	63	70
8	16	24	32	40	48	56	64	72	80
9	18	27	36	45	54	63	72	81	90
10	20	30	40	50	60	70	80	90	100



i,j	1	2	3	4	5
1	1,1	1,2	1,3	1,4	1,5
2	2,1	2,2	2,3	2,4	2,5
3	3,1	3,2	3,3	3,4	3,5
4	4,1	4,2	4,3	4,4	4,5
5	5,1	5,2	5,3	5,4	5,5

```

1  #include <stdio.h >
2  #include <conio.h>
3  int main(){
4      for(int i = 1;i<=10;i++){
5          for(int j=1;j<=10;j++){
6              if((i+j)%2==0)
7                  printf("*");
8              else
9                  printf(" ");
10         }
11         printf("\n");
12     }

```

C:\Users\shedai\Desktop\www.bilgisayarkavramlari.com\Untitled10.exe

```

* * * * *
* * * * *
* * * * *
* * * * *
* * * * *
* * * * *
* * * * *
* * * * *

```

i,j	1	2	3	4	5
1	2	3	4	5	6
2	3	4	5	6	7
3	4	5	6	7	8
4	5	6	7	8	9
5	6	7	8	9	10


```
1  #include <stdio.h >
2  #include <conio.h>
3  int main(){
4      for(int i = 1;i<=5;i++){
5          for(int j=1;j<=5;j++){
6              if((i+j)==6 || i==j)
7                  printf("*");
8              else
9                  printf(" ");
10         }
11         printf("\n");
12     }
13
14     getch();
15 }
```

C:\Users\shedai\Desktop\www.bilgisayarkavramlari.com\Untitled11.exe

```
*  *
* *
 *
* *
* *
```

```

1  #include <stdio.h>
2  #include <conio.h>
3  int main(){
4      for(int i = 0;i<=10;i++){
5          for(int j=0;j<=i;j++){
6
7              int nf=1;
8              for(int p =1;p<=i;p++){
9                  nf= nf*p;
10             }
11             int kf=1;
12             for(int p =1;p<=j;p++){
13                 kf= kf*p;
14             }
15             int nkf=1;
16
17             for(int p =1;p<=i-j;p++){
18                 nkf= nkf*p;
19             }
20             printf("%d ",nf/(kf*nkf));
21         }
22         printf("\n");
23     }
24
25     getch();
26 }

```

0,0					
1,0	1,1				
2,0	2,1	2,2			
3,0	3,1	3,2	3,3		
4,0	4,1	4,2	4,3	4,4	
5,0	5,1	5,2	5,3	5,4	5,5

C:\Users\shedai\Desktop\www.bilgisayarkavramlari.com\Untitled12.exe

```

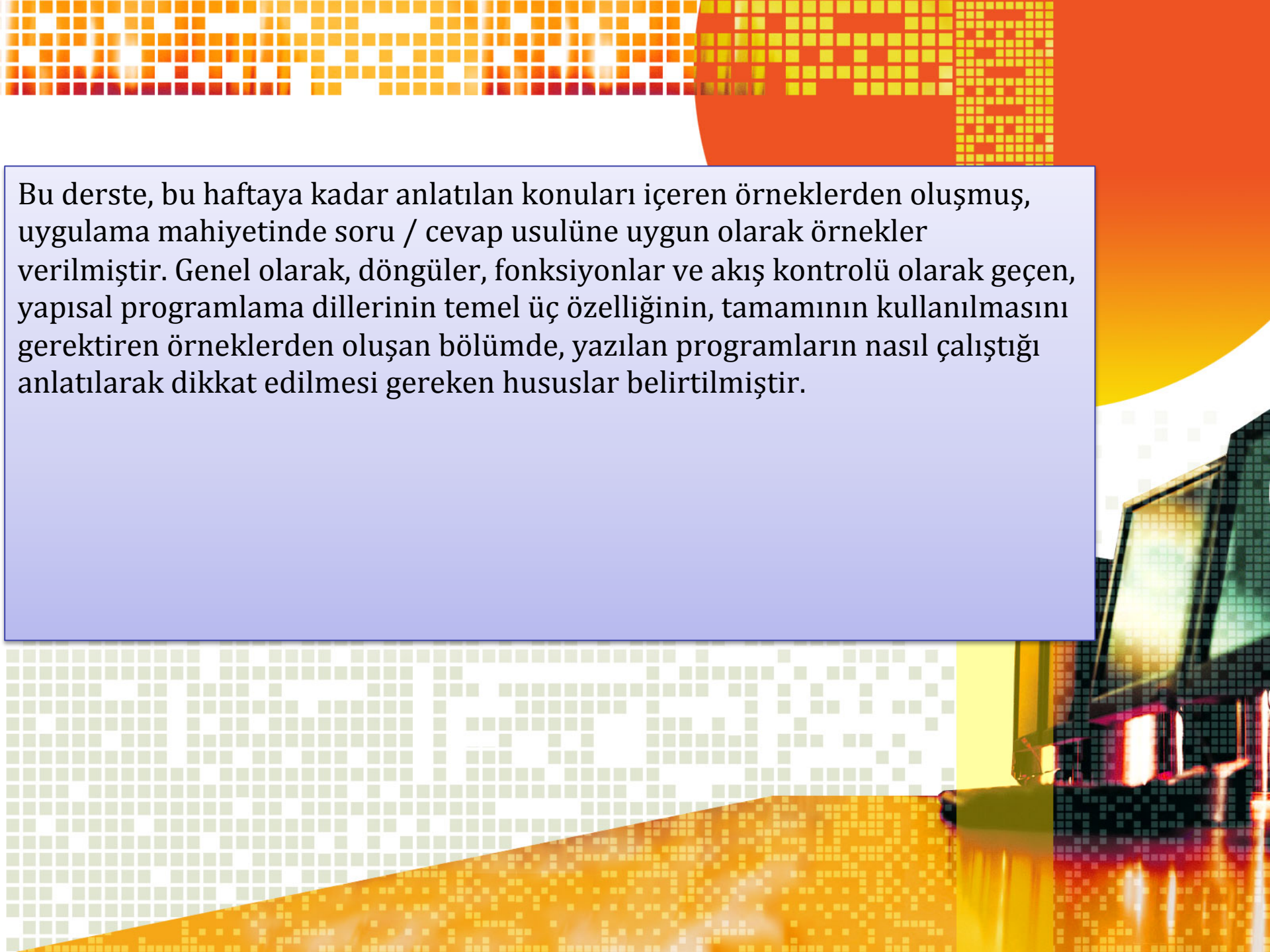
1
1 1
1 2 1
1 3 3 1
1 4 6 4 1
1 5 10 10 5 1
1 6 15 20 15 6 1
1 7 21 35 35 21 7 1
1 8 28 56 70 56 28 8 1
1 9 36 84 126 126 84 36 9 1
1 10 45 120 210 252 210 120 45 10 1

```

```
1  #include <stdio.h >
2  #include <conio.h>
3  int main(){
4      int sum = 0;
5      for(int d = 1;d<=100;d++) {
6          int p=0;
7          for(int i = 2;i<=d-1;i++){
8              if(d%i==0){
9                  p=1;
10             }
11         }
12         if(p==0){
13             printf("%d\n",d);
14             sum = sum +d;
15         }
16     }
17     printf("sum : %d",sum);
18     getch();
19 }
```

C:\Users\shedai\Desktop\www.bilgisayarkavramlari.com\Untitled13.exe

```
3
5
7
11
13
17
19
23
29
31
37
41
43
47
53
59
61
67
71
73
79
83
89
97
sum : 1061_
```

Bu derste, bu haftaya kadar anlatılan konuları içeren örneklerden oluşmuş, uygulama mahiyetinde soru / cevap usulüne uygun olarak örnekler verilmiştir. Genel olarak, döngüler, fonksiyonlar ve akış kontrolü olarak geçen, yapısal programlama dillerinin temel üç özelliğinin, tamamının kullanılmasını gerektiren örneklerden oluşan bölümde, yazılan programların nasıl çalıştığı anlatılarak dikkat edilmesi gereken hususlar belirtilmiştir.

5. Hafta

Bu derste, temel olarak bir bilgisayar programının, kodun bir kısmını, koşullu olarak nasıl tekrar ettiğini öğreneceğiz. Yapısal programlama dillerinin tamamında bulunan bu özellik, programcının, aynı veya benzer işlemleri, bilgisayara yaptırmak istemesi sırasında, koddan istediği bir bloğu, parametrik olarak şarta bağlaması ile olur.



Bölüm İçeriği

- Fonksiyonlar
 - Özyineli Fonksiyonlar (Recursive)
 - C Dilinde Fonksiyonlar
 - Örnekler
- Sorular

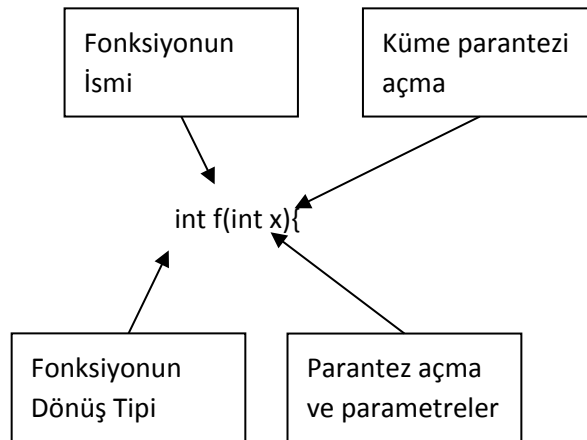


Matematikteki Fonksiyonlar ve Programlama Dilleri ilişkisi

- $f(x,y)=x^2+2xy+y^2$
- fonksiyonu tanımlıysa ve $f(3,4)$ şeklinde bu fonksiyon çağrılırsa $x=3,y=4$ şeklinde fonksiyon çözümlenir ve $9+24+16 = 41$ olarak bulunur.
- $g(x) = 2x$
- $f(3,4)+g(5)+ f(1,2) = 41 + 10 + 9 = 60$
- Yukarıda yazılmış olan fonksiyonu programlama diline çevirecek olursak:
- ```
int f(int x, int y){
```
- ```
    return 20 ;// x*x + 2*x*y + y*y;
```
- ```
}
```



# Fonksiyonun Anatomisi



# Örnek Kod

- Verilen bir sayının faktöriyelini bulan kod

- `#include <stdio.h>`
- `#include <conio.h>`
- `int fact(int);`
- `int main(){`
- `printf(" bir sayi giriniz:");`
- `int x;`
- `scanf("%d",&x);`
- `printf("5! + girilen sayının faktoriyeli : %d", fact(x)+fact(5));`
- `getch();`
- `}`
- `int fact( int x){`
- `int sonuc = 1;`
- `for(int i = 1;i<=x;i++){`
- `sonuc *=i;`
- `}`
- `return sonuc;`
- `}`





# Örnek Kod

- Verilen sayıların kombinasyonunu bulan kod

- `#include <stdio.h>`
- `#include <conio.h>`
- `int fact(int);`
- **`int comb(int,int);`**
- `int main(){`
- `printf(" iki sayi giriniz:");`
- `int x,y;`
- `scanf("%d%d",&x,&y);`
- `printf("kombinasyonu : %d", comb(x,y));`
- `getch();`
- `}`
- **`int comb(int x,int y){`**
- **`return fact(x)/(fact(y)*(fact(x-y)));`**
- **`}`**
- `int fact( int x){`
- `int sonuc = 1;`
- `for(int i = 1;i<=x;i++){`
- `sonuc *=i;`
- `}`
- `return sonuc;`
- `}`



# Fibonacci Sayıları

- Başlangıçtaki iki terimi 1 olan ve sonraki terimleri, kendinden önceki, son iki sayının toplamı olan seridir.
- $F(0) = 1, F(1) = 1, F(n) = F(n-1) + F(n-2)$  şeklinde gösterilebilir.
- 1,1,2,3,5,8,13,21 ...



# Örnek Kod

- Verilen sıradaki fibonacci sayısını bulan kod

- `#include <stdio.h>`
- `#include <conio.h>`
- `int fib(int);`
- `int main(){`
- `int x ;`
- `scanf("%d",&x);`
- `printf("%d ", fib(x));`
- `getch();`
- `}`
- `int fib(int n){`
- `int a = 1,b = 1;`
- `int c;`
- `for(int i = 3;i<=n;i++){`
- `c =a+b;`
- `a=b;`
- `b=c;`
- `}`
- `return c;`
- `}`

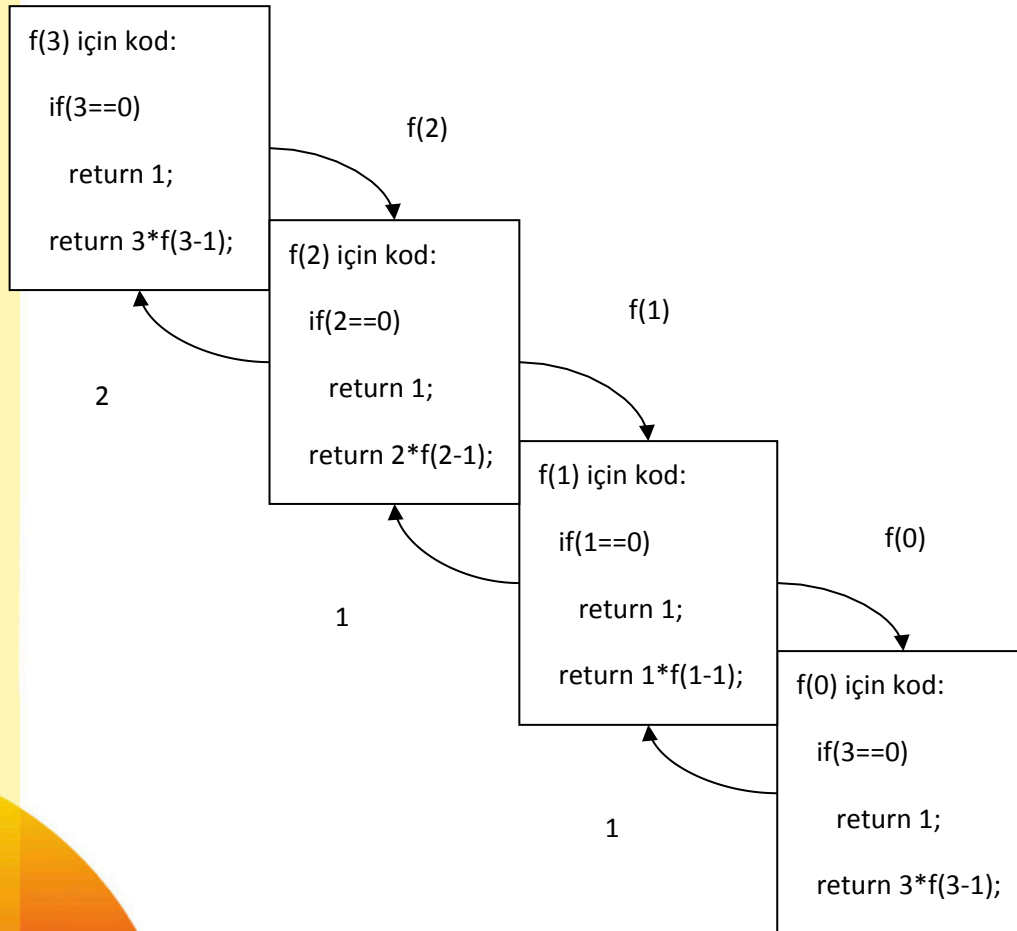




# Özyineli Fonksiyonlar (Recursive)

- Bir fonksiyonun, kendisini çağırması durumudur. Örneğin faktöriyel için:
- $n! = n * (n-1)!$
- Formülünde ! İşleminin tanımında yine ! cinsinden bir fonksiyon bulunur:
- ```
int f(int n){  
    if(n==0)  
        return 1;  
    return n*f(n-1);  
}
```

Özyineli Fonksiyonlar

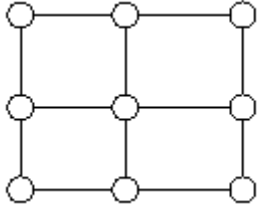


Özyineli olarak Fibonacci

- `#include <stdio.h>`
- `#include <conio.h>`
- `int fib(int);`
- `int main(){`
- `int x ;`
- `scanf("%d",&x);`
- `printf("%d ", fib(x));`
- `getch();`
- `}`
- `int fib(int n){`
- `if(n==1||n==0)`
- `return 1;`
- `return fib(n-1)+fib(n-2);`
- `}`



Örnek Problem



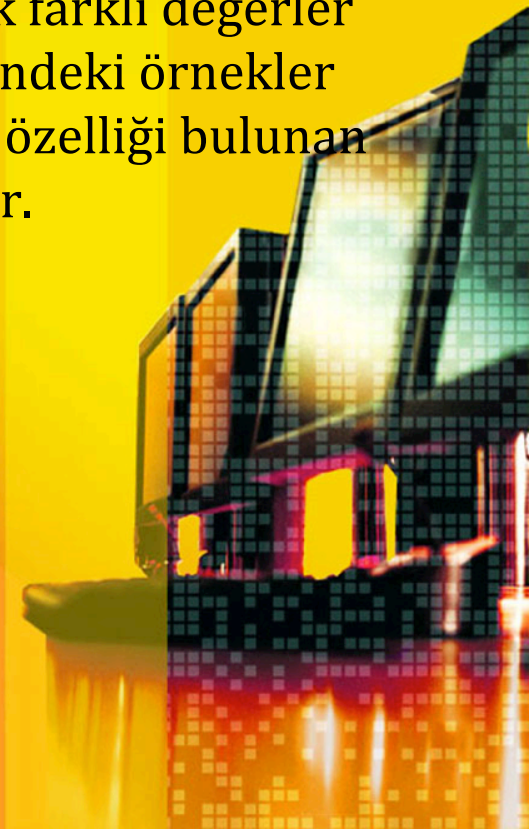
- Yukarıdaki grafikteki kare sayısı 5tir. Buna göre verilen bir kare grafik diziliminin boyutuna göre $(n \times n)$ kaç kare içereceğini hesaplayan fonksiyon yazınız.

Çözüm

- Recursive bir problemdir. Buna göre örneğin 3x3lük bir ızgarada 4 adet 2x2lik ızgaradaki kadar kare ve ilave olarak 1 kare daha bulunacaktır.
- ```
int kare_sayisi(int n){
```
- ```
    if(n==1)
```
- ```
 return 1;
```
- ```
    return kare_sayisi(n-1)*4 +1;
```
- ```
}
```

Bu derste, programlama dillerinde kullanılan fonksiyon kavramına giriş yapılmıştır. Yapısal programlama dillerinin temel üç özelliğinden birisi olan ve bir program bloğunun parametrik olarak farklı değerler için çalışabilmesini sağlayan fonksiyon kavramı, C dilindeki örnekler üzerinden anlatılmıştır. Ayrıca kendi kendini çağırma özelliği bulunan özyineli fonksiyonlar anlatılmış ve örnekler verilmiştir.

## SONUÇ





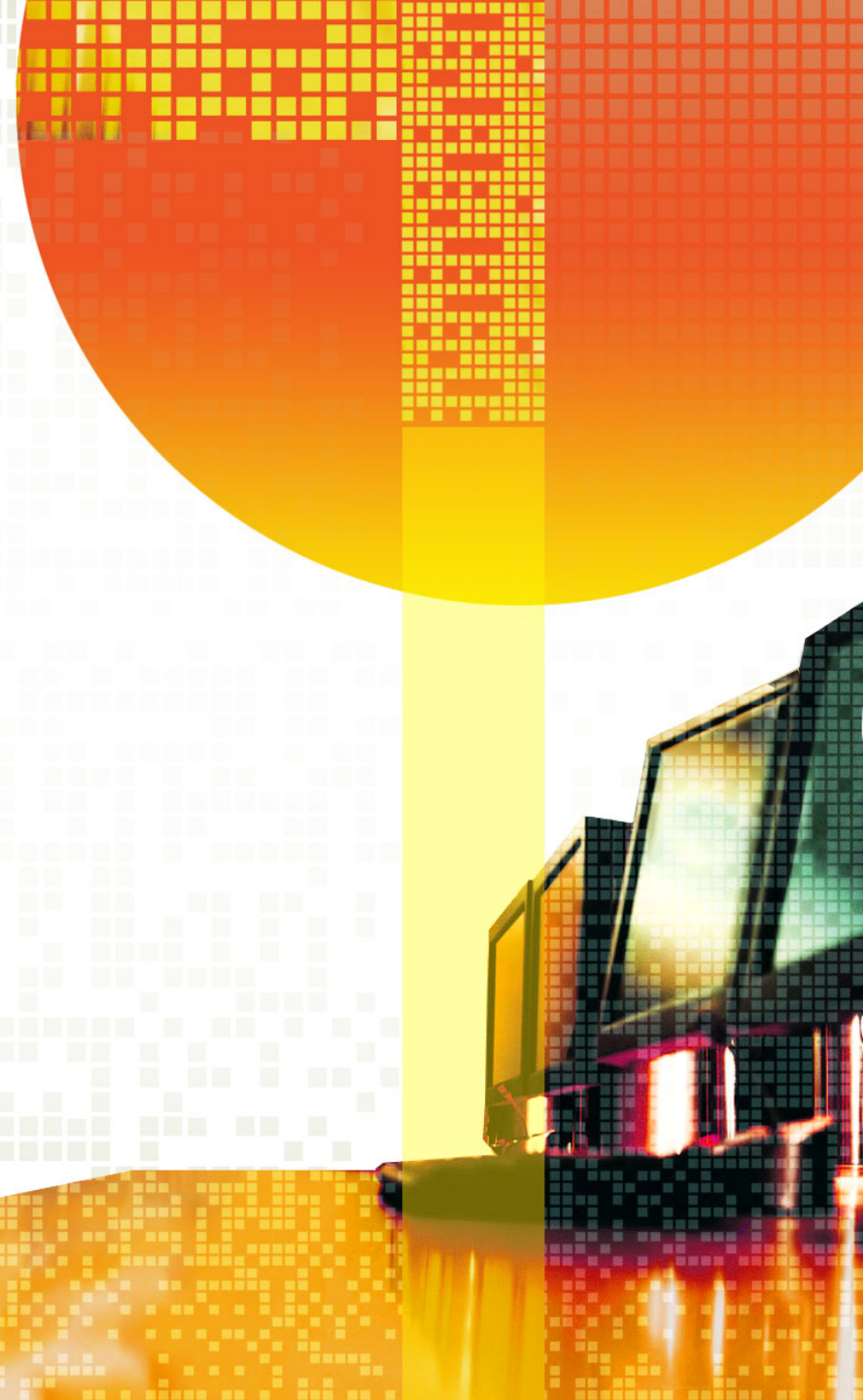
# Döngüler

Bu derste, temel olarak bir bilgisayar programının, kodun bir kısmını, koşullu olarak nasıl tekrar ettiğini öğreneceğiz. Yapısal programlama dillerinin tamamında bulunan bu özellik, programcının, aynı veya benzer işlemleri, bilgisayara yaptırmak istemesi sırasında, koddan istediği bir bloğu, parametrik olarak şarta bağlaması ile olur.



# Bölüm İçeriği

- Döngüler
  - Basit Döngüler
  - İç içe döngüler (nested loops)
  - Örnekler
- Sorular





## Örnek Döngü

- `#include <stdio.h>`
- `#include <conio.h>`
- `int main(){`
- `for(int i = 51; i <= 50; i += 2){`
- `printf(" %d ", i);`
- `}`
- `getch();`
- `}`

Ekran :

21 23 25 27  
29 31 ...  
49

`i=21`  
`i+= 2`  
`i=i+2`

Initialization  
Condition  
Iteration



## Döngünün 3 unsuru

- Başlangıç değeri (intial value)
- Adım değeri (step value, iteration value)
- Bitiş koşulu (condition)



## While Döngüsü

- `int i=11;`
- `printf(" %d ",i);`
- `i++;`
- `while(i<=10){`
- `printf(" %d ",i);`
- `i++;`
- `}`



## do..While Döngüsü

- `int i=11;`
- `do{`
- `printf(" %d ",i);`
- `i++;`
- `} while(i<=10);`





## Örnek Kod

20 ile 50 arasındaki tek sayıları ekrana bastıran kodu yazalım.

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

```
#include <conio.h>
```

```
int main(){
 for(int i =21;i<=50;i+=2){
 // if(i%2 == 1)
 printf(" %d ",i);
 }
 getch();
}
```



## Örnek Kod

100'den 20'ye kadar, 3'e ve 7'ye tam  
Bölünen sayıları, büyükten küçüğe doğru  
yazdıralım

```
#include <stdio.h>
#include <conio.h>
int main(){
 for(int i = 100; i >= 20; i--){
 if(i % 3 == 0 && i % 7 == 0)
 printf("%d ", i);
 }
 getch();
}
```





# Örnek Kod

Kullanıcıdan 3 adet sayı okuyup en  
Büyüğünü ekrana yazan kod

```
#include <stdio.h>
#include <conio.h>
int main(){
 int girilen;
 int eb = 0;
 for(int i = 0;i<5;i++){
 int x = 0;
 scanf("%d",&girilen);
 if(girilen>eb)
 eb = girilen;
 x += girilen;
 }
 printf(" eb : %d",eb);
 getch();
}
```





## Örnek Kod

Kullanıcı, -1 girene kadar yazılan sayıların en büyüğünü bulan kod

- `#include <stdio.h>`
- `#include <conio.h>`
- `int main(){`
- `int girilen=0;`
- `int eb = 0;`
- `while(girilen !=-1){`
- `scanf("%d",&girilen);`
- `if(girilen>eb)`
- `eb = girilen;`
- `}`
- `printf(" eb : %d",eb);`
- `getch();`
- `}`

# Örnek Kod

- Kullanıcı -1 girene kadar girilen sayıların ortalamasını bulan kod
- `#include <stdio.h>`
- `#include <conio.h>`
- `int main(){`
- `int girilen=0;`
- `int ort;`
- `int n = 0;`
- `int toplam = 0;`
- `while(girilen !=-1){`
- `scanf("%d",&girilen);`
- `n++;`
- `toplam = toplam + girilen;`
- `}`
- `printf(" ortalama : %d",toplam/n);`
- `getch();`
- `}`



# Örnek Kod

- Girilen sayının asal olup olmadığını bulan kod
- `#include <stdio.h>`
- `#include <conio.h>`
- `int main(){`
- `int a=1,b;`
- `scanf("%d",&b);`
- `int flag = 0;`
- `while (a<b-1){`
- `a++;`
- `if(b%a == 0){`
- `flag = 1;`
- `break;`
- `}`
- `}`
- `if(flag == 0)`
- `printf(" sayi asal");`
- `else`
- `printf(" sayi asal degil");`
- `getch();`
- `}`





# İç içe döngüler

- Örnek tablo

| 1 | 2  | 3  | 4  | 5  |
|---|----|----|----|----|
| 2 | 4  | 6  | 8  | 10 |
| 3 | 6  | 9  | 12 | 15 |
| 4 | 8  | 12 | 16 | 20 |
| 5 | 10 | 15 | 20 | 25 |

## Örnek tabloyu basan kod

- `for(int j =1;j<=5;j++){`
- `for(int i =1;i<=5;i++){`
- `printf(" %d ",i);`
- `}`
- `printf("\n ");`
- `}`



## Örnek Kod

- Ekrana \* sembolü ile dik üçgen çizdiren kod
- `#include <stdio.h>`
- `#include <conio.h>`
- `int main(){`
- `for(int n = 0;n<5;n++){`
- `for(int i = 0;i<=n;i++){`
- `printf("*");`
- `}`
- `printf("\n");`
- `}`
- `getch();`
- `}`

```
*
**


```





## Örnek Kod

- `#include <stdio.h>`
- `#include <conio.h>`
- `int main(){`
- `for(int n = 4;n>=0;n--){`
- `for(int i = 0;i<=n;i++){`
- `printf("*");`
- `}`
- `printf("\n");`
- `}`
- `getch();`
- `}`

5  
4  
3  
2  
1

```


**
*
```

## Örnek Kod

- `#include <stdio.h>`
- `#include <conio.h>`
- `int main(){`
- `for(int n = 4;n>=0;n--){`
- `// 4-n yildiz`
- `for(int i = 0;i<4-n;i++){`
- `printf(" ");`
- `}`
- `for(int i = 0;i<=n;i++){`
- `printf("*");`
- `}`
- `printf("\n");`
- `}`
- `getch();`
- `}`

5-

5  
4  
3  
2  
1  
yildiz

=

0 2  
1 3  
2 5  
3 7  
4 11  
bosluk

\* \* \* \* \*  
\* \* \* \*  
\* \* \*  
\* \*  
\*



# Sonu

Bu derste, programlama dillerinin d kontrol aıklanmıřtır. Yapısal programlama dillerinin temel  zelliğinden birisi olan ve bir program bloğunun belirli bir řart altında tekrar etmesini saėlayan dler, C dili zerinden rnekler ile anlatılmıřtır. Ayrıca i d d kavramı anlatılmıř ve birden fazla boyutu olan verilerin iřlenmesi konusunda giriř yapılmıřtır.





### 3. Ders

Bu derste, temel olarak bir bilgisayar programının, bilgisayar tarafından alıřtırılması sırasında izlenen akıř ve bu akıřın nasıl kontrol edilebileceęi, řartlara baęlı olarak nasıl ynlendirilebileceęi anlatılacaktır.



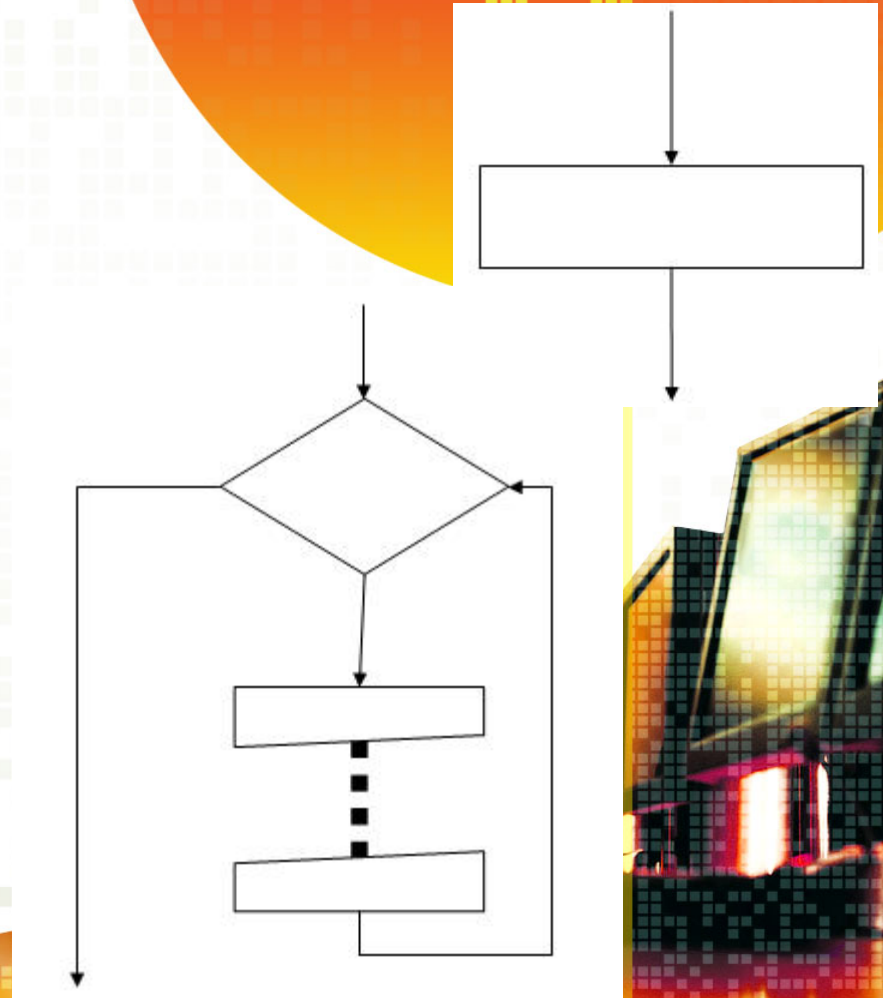
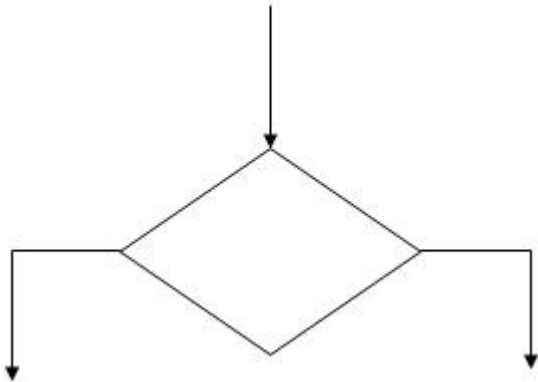
# Bölüm İçeriği

- Akış Kontrolü
  - Akış Çizelgeleri (Flow Charts)
  - C Dilinde Akış Kontrolü
- Sorular



Click to add title

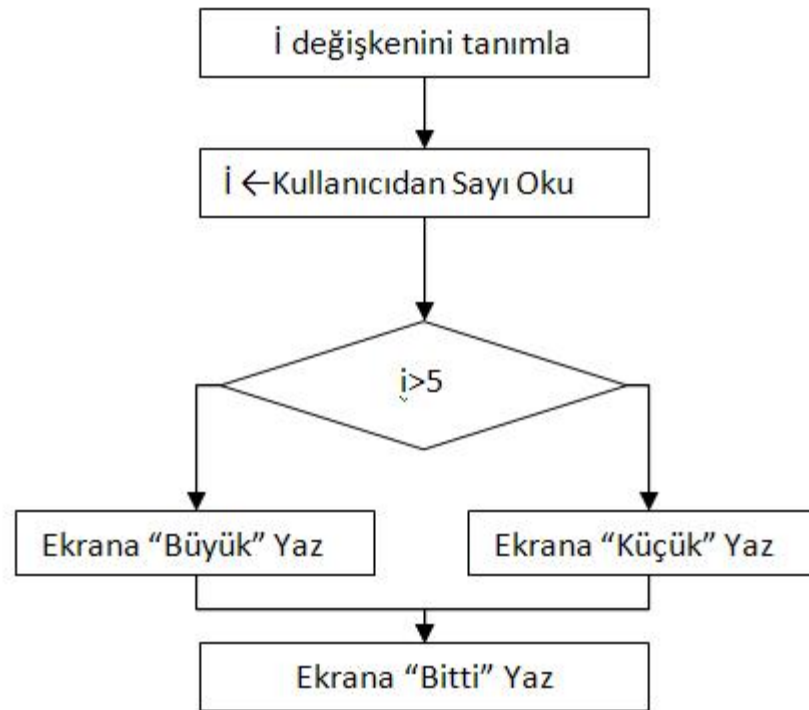
- Talimatlar (Statements)
- Döngüler (Loops)
- Koşullar (Conditions)





## Akış Kontrolü (Örnek)

- Kullanıcıdan Sayı
- Okuyarak, okunan
- Sayının 5'ten büyük
- Olması Durumunda
- Ekran «büyük»
- Aksi halde «küçük»
- Yazan kodun tasarır



# Kodlama

- Akış Kontrolünün C dilindeki karşılığı
  - *If( $i > 5$ ) // koşul kontrolü ve dallanmayı sağlayan kod*
  - *{*
  - *// koşul doğru ise büyük*
  - *} else { //koşul yanlışsa durumu kontrol ediliyor*
  - *//koşul yanlışsa küçük*
  - *}*



## Else If Kullanımı

- *if(i>5) // koşul kontrolü ve dallanmayı sağlayan kod*
- *{*
- *// koşul doğru ise büyük*
- *} else if(i==5){ // şayet büyük değilse eşit mi diye bak*
- *// eşit durumunda çalışacak kod*
- *}else{ //koşul yanlışsa durumu kontrol ediliyor*
- *//koşul yanlışsa küçük*
- *}*



## Örnek C Kodu

- *#include <stdio.h>*
- *#include <conio.h>*
- *int main(){*
- *int i;*
- *scanf("%d",&i);*
- *if(i>5) // koşul kontrolü ve dallanmayı sağlayan kod*
- *{*
- *printf("Buyuk"); // koşul doğruysa bu alt program çalışır*
- *} else { //koşul yanlışsa durumu kontrol ediliyor*
- *printf("Buyuk degil");*
- *}*
- *printf("Bitti");*
- *getch();*
- *}*

# Örnek Uygulama

- Kullanıcıdan iki sayı okuyarak büyük olanı ekrana basan kod.
- `#include <stdio.h>`
- `#include <conio.h>`
- `int main(){`
- `int a,b;`
- `printf(" iki sayı giriniz");`
- `scanf("%d%d",&a,&b);`
- `if(a>b)`
- `printf("büyük sayı: %d",a);`
- `else if(a<b)`
- `printf("büyük sayı: %d",b);`
- `else`
- `printf("sayılar eşit");`
- `getch();`
- `}`



## Örnek Uygulama

- Bir şirket, çalışanlarına, fazla mesai ücreti ödemektedir. Sizden ücreti hesaplayan bir program yazmanız isteniyor. Programın özellikleri aşağıdaki şekildedir:
- 10 saate kadar saat başına 5 lira
- 10 ile 20 saat arasında, saat başına 3 lira
- 20 saatten sonrası için, saat başına 2 lira





# Çözüm

- `#include <stdio.h>`
- `#include <conio.h>`
- `int main(){`
- `int x;`
- `scanf("%d",&x);`
- `int ucret = 0;`
- `if(x<=10){`
- `ucret = x * 5;`
- `}`
- `else if(x<=20) {`
- `ucret = (x-10)*3 + 50;`
- `}`
- `else {`
- `ucret = (x-20)*2 + 80;`
- `}`
- `printf("mesai ucreti : %d", ucret);`
- `getch();`
- `}`



## Örnek Uygulama

- Bir öğrencinin notu, 100 üzerinden girildiğinde aşağıdaki değerlere göre harf notu karşılığını hesaplayan programı yazınız:
- 90 ve üzeri AA
- 80 ile 90 arası BA
- 70 ile 80 arası BB
- 70 ve altı F





# Hatalı Kod

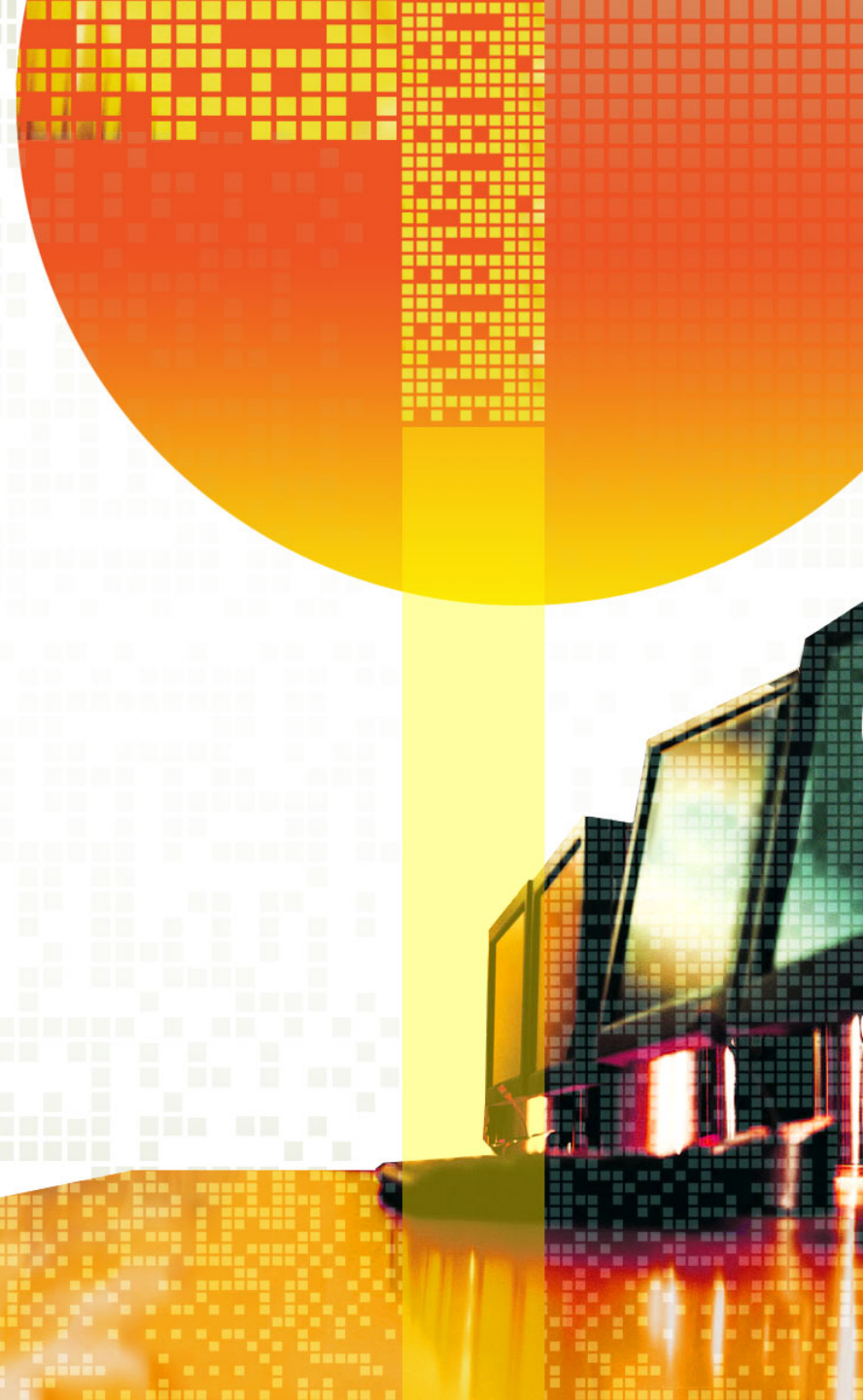
- `#include <stdio.h>`
- `#include <conio.h>`
- `int main(){`
- `printf("notu giriniz");`
- `int a;`
- `scanf("%d",&a);`
- `if(a>90){`
- `printf("AA");`
- `if(a>80 && a<90)`
- `printf("BA");`
- `if(a>70 && a<80)`
- `printf("BB");`
- `else`
- `printf("F");`
- `getch();`
- `}`





# Doğru Kod

- `#include <stdio.h>`
- `#include <conio.h>`
- `int main(){`
- `printf("notu giriniz");`
- `int a;`
- `scanf("%d",&a);`
- `if(a>=90){`
- `printf("AA");`
- `else if(a>=80)`
- `printf("BA");`
- `else if(a>=70)`
- `printf("BB");`
- `else`
- `printf("F");`
- `getch();`
- `}`



## Sonuç

Bu derste, programlama dillerinin akışını kontrol etmeye yarayan, akış kontrolü konusu anlatılmıştır. Yapısal programlama dillerinin temel üç özelliğinden birisi olan ve bir program bloğunun belirli bir şarta bağlı olarak çalışıp çalışmamasına karar veren akış kontrolleri, özel olarak C dilindeki if blokları üzerinden örnekler ile açıklanmıştır.





## 2. Ders

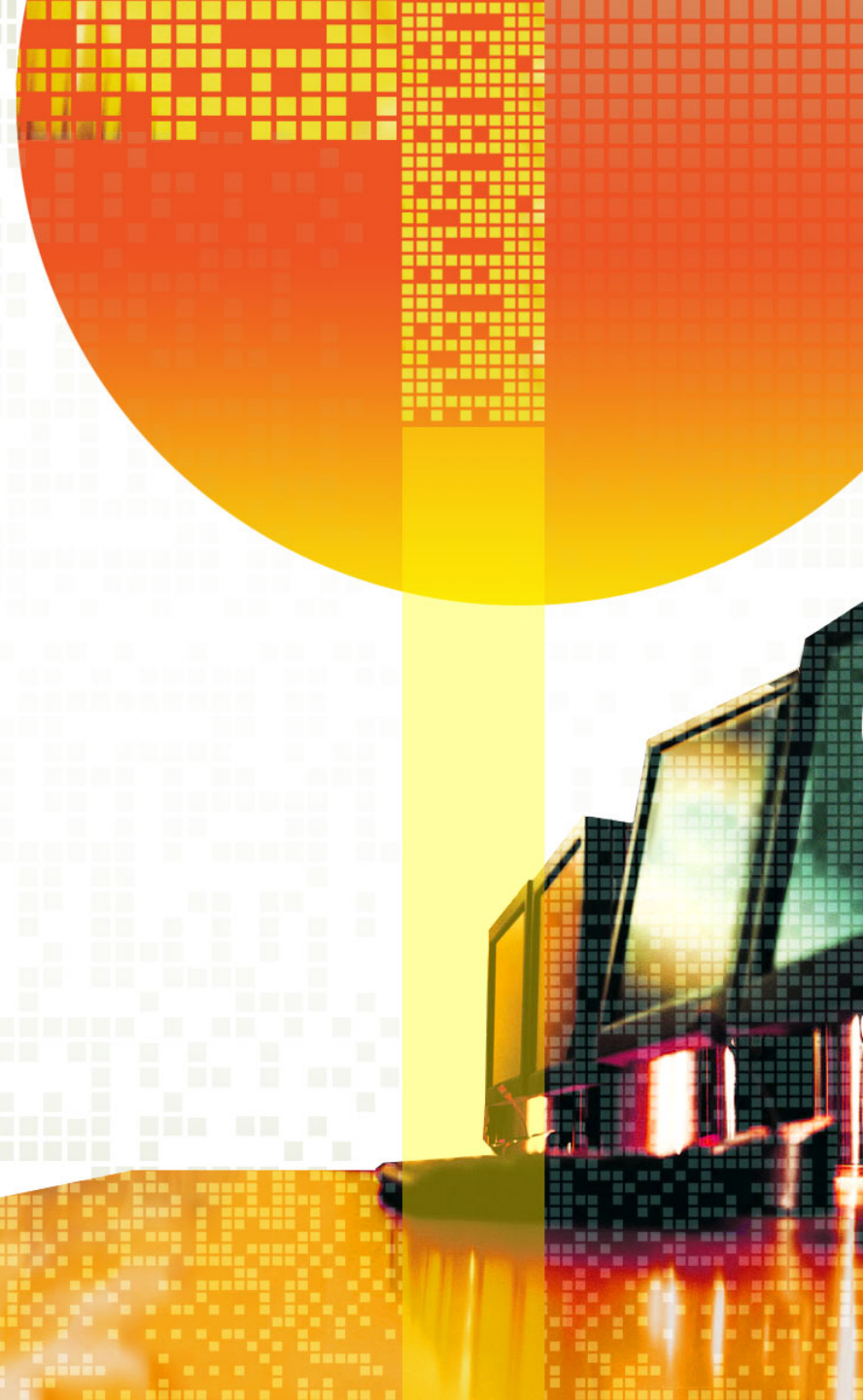
Bu derste, kullandığımız dilde, uymamız gereken kuralları ve bir kod yazarken, kodun çalışmasını doğrudan etkilemeyen fakat bize kolaylık sağlayan yazma özelliklerini öğreneceğiz. Ayrıca herhangi bir dil için en kritik konulardan birisi olan işlemleri öğrenecek ve C dilinde işlemlerin nasıl kodlandığını ve çalışmalarını öğreneceğiz.





# Bölüm İçeriği

- C Dili için Yazım Kuralları
  - Yorumlar (Comments)
  - Talimatlar (Statements)
  - Bloklar
  - C Dilinde Örnek Kod
- İşlemler (Operators)
  - C Dilinde İşlem Kullanımı
- Sorular



# Programlama Dilindeki Temel Unsurlar

- Talimatlar (Statements)
- Blocklar (blocks)
- Yapısal Programlama  
Structured Programming
  - Bloğun Koşullu olması
  - Bloğun Tekrarlanabilmesi
  - Bloğun Parametrik olması





- */\* Bu kod blok konusunu denemek için*
- *yazılmıştır.*
- *\*/*
- *#include <stdio.h>*
- *#include <conio.h>*
- *int main(){*
- *// a isminde bir değişken main fonksiyonu*
- *// içerisinde tanımlanıyor*
- *int a=10;*
- *{*
- *// küme parantezleri ile bir alt blok oluşt*
- *int a=5;*
- *printf("\n%d",a);*
- *}*
- *printf("\n%d",a);*
- *getch();return 0;*
- *}*

- Bloklar
- Yorumlar
- Değişken İsimleri



# İşlemler

- Matematiksel İşlemler
  - Tekil İşlemler ( ++ ) unary ops
  - İkili İşlemler ( + ) binary ops
- Mantıksal İşlemler ( a >= )

Logical Operators,  
boolean algebra, boolean logic

- Atama İşlemi ( = )

Assignment Operators

```
int a;
int b=20;
int c=30;
a=b+c;
b=c+a*2;
```

# Örnek Kod (Matematiksel İşlemler)

- `#include <stdio.h>`
  - `#include <conio.h>`
  - `int main(){`
  - `int a=10;`
  - `int b=3,c=7;`
  - `printf("%d",a+b);`
  - `printf("\n%d",a<<2);`      $2 \ll -> 8, 1000$
  - `printf("\n%d",a>>2);`      $0010$
  - `printf("\n%d",a+b*2);`
  - `printf("\n%d",a%b);`      $10 = 1010$
  - `printf("\n%d",a==b);`      $\ll 2 -> 0010\ 1000 = 40$
  - `printf("\n%d",a>b);`      $\gg 2 -> 001010 = 2$
  - `printf("\n%d",a++);` Post increment
  - `printf("\n%d",++a);` Pre increment
  - `getch();`
  - `return 0;`
  - `}`
- Remainder , modulo  
 $10 \% 3 = 1$   
 $10 \bmod 3 = 1$



## İkilik Tabanda İşlemler

- Kaydırma İşlemleri (  $\gg$  ,  $\ll$  )
- Ve İşlemi (  $\&$  )
- Veya İşlemi (  $|$  )

```
| &
5 = 0101
2 = 0010
 & 0000
| 0111
```

|      |      |
|------|------|
| 00 0 | 00 0 |
| 10 0 | 10 1 |
| 01 0 | 01 1 |
| 11 1 | 11 1 |

```
int a;
int b=5;
int c=2;
a=b&c; // 0
b=c/a&2; // 2
c = c << 2; // 8
```



# Mantıksal İşlemler

- Eşitlik Kontrolü ( == , != )
- Büyüklük / Küçüklük kontrolü ( > , < , <= , >= )
- İki mantıksal işlemin birleştirilmesi
  - Ve bağlacı ( && )
  - Veya bağlacı ( || )

5 == 3 -> 0

5 == 5 -> 1

a == 3

a != 3

a > 3 , a < 3

a > 3 || a < 0

a <= 3 && a >= 0

## Bazı İşlem Örnekleri

- Başlangıç durumu olarak int  
`a=10; int b=3; int c=7;`

| İşlem            | Sonuç                                                                                                         |
|------------------|---------------------------------------------------------------------------------------------------------------|
| <code>a++</code> | a değişkeninin değeri 11 olur                                                                                 |
| <code>++a</code> | a değişkeninin değeri 11 olur                                                                                 |
| <code>a+b</code> | Toplam değeri 13 olur                                                                                         |
| <code>a-b</code> | Çıkarma işleminin sonucu 7 olur                                                                               |
| <code>a*b</code> | Çarpma işleminin sonucu 30 olur                                                                               |
| <code>a/b</code> | Bölme işleminin sonucu 3 olur. (ondalıklı sayılar int tipi ile tanımsızdır dolayısıyla tam sayı kısmı alınır) |
| <code>a%b</code> | Kalan 1 'dir . 10'un 3'e bölümünden kalan.                                                                    |



# İşlem Örnekleri (Devam)

|                   |                                                                                                                                                                                                                                                                                                              |
|-------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>a&lt;&lt;2</b> | a sayısının 2 sola kayması demektir. Bunu anlamak için önce a sayısının ikilik tabandaki karşılığını bulalım. $a_2 = 00001010$ sola iki basamak kaydırılırsa 00101000 sayısı bulunur ve bu sayının onluk sistemdeki karşılığı 40 sayısıdır. Dolayısıyla bu satırın çalışması sonucu 40 değeri bulunmuş olur. |
| <b>a&gt;&gt;2</b> | a sayısının 2 sağa kaymış hali. Bunu anlamak için önce a sayısının ikilik tabandaki karşılığını bulalım. $a_2 = 00001010$ sağa iki basamak kaydırılırsa 00000010 sayısı bulunur ve onluk sistemdeki karşılığı 2'dir.                                                                                         |
| <b>a&lt;b</b>     | Mantıksal bir karşılaştırma yapılır. a'nın sayısal değeri b'den küçük mü diye kontrol edilir. Küçük olmadığı için C dilinde olumsuz anlamında 0, C++ ve JAVA dillerinde ise false değeri döner.                                                                                                              |
| <b>a&gt;b</b>     | Mantıksal bir karşılaştırma yapılır. a'nın sayısal değeri b'den büyük mü diye kontrol edilir. Büyük olduğu için C dilinde olumlu anlamında 1, C++ ve JAVA dillerinde ise true değeri döner.                                                                                                                  |



## İşlem Örnekleri (Devam)

|                               |                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|-------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>a==b</b>                   | a ile b'nin sayısal değerlerinin eşit olup olmadığı kontrol edilir. a ile b'nin sayısal değerleri aynı olmadığı için C dilinde olumsuz anlamında 0, C++ ve JAVA dillerinde ise false değeri döner.                                                                                                                                                                                                                                      |
| <b>a!=b</b>                   | a ile b'nin sayısal değerleri eşit değiller mi diye kontrol edilir. a ile b'nin sayısal değerleri aynı olmadığı için C dilinde olumlu anlamına gelen 1, C++ ve JAVA dillerinde ise true değeri döner.                                                                                                                                                                                                                                   |
| <b>a&gt;b&amp;&amp;b&gt;c</b> | İşlem iki bölümden oluşmaktadır. İlk bölümde a'nın değerinin b'den büyük olması ikinci bölümde ise b'nin değerinin c'den büyük olması kontrol edilmiştir. Bu iki işlem arasında mantıksal VE işlemi vardır. VE işleminden bilineceği üzere aynı anda iki durumda doğru olduğunda doğru sonucu çıkar. Bu satırda a, b'den büyük olacak ve b, c'den büyük değildir dolayısıyla C dilinde 0, C++ veya JAVA dillerinde false değeri alınır. |

# İşlem Örnekleri (Devam)

|                         |                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|-------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| $a > b    b > c$        | İşlem iki bölümden oluşmaktadır. İlk bölümde a'nın değerinin b'den büyük olması ikinci bölümde ise b'nin değerinin c'den büyük olması kontrol edilmiştir. Bu iki işlem arasında mantıksal VEYA işlemi vardır. VEYA işleminden bilineceği üzere taraflardan birisinin doğru olması sonucun doğru olması için yeterlidir. Bu satırda a, b'den büyüktür ancak b, c'den büyük değildir dolayısıyla C dilinde 1, C++ veya JAVA dillerinde true değeri alınır. |
| $a += 5$<br>$a = a + 5$ | Bu satırda a değişkeninin değeri 5 artırılarak kendi içerisine konulur. Bu satırın açılmış hali $a = a + 5$ olarak da yazılabilir.                                                                                                                                                                                                                                                                                                                       |
| $a -= b$<br>$a = a - b$ | Bu satırda a değişkeninin değeri b kadar azaltılarak kendi içerisine konulur. Bu satırın açılmış hali $a = a - b$ olarak da yazılabilir.                                                                                                                                                                                                                                                                                                                 |



## Sorular

- Aşağıdaki problemi çözen bir kod yazınız:
- Bir havuzu a musluğu 3, b musluğu 8 ve c musluğu 12 saatte doldurabilmektedir. Buna göre 3 musluk aynı anda açılırsa havuz kaç saatte dolar?
- Kullanıcıdan a,b ve c sayılarını okuyarak aşağıdaki işlemi yapan kodu yazınız
- $a^2+b^2+3c$
- Kullanıcıdan bugünün'ün tarihini ve kaç yaşında olduğunu alarak doğum tarihini yıl olarak bulan kod yazınız.



# Ders Sonu

Sonraki Ders: Akış Kontrolü

Youtube: Bilgisayar Kavramları

Bu derste, genel olarak programlama dillerindeki işlemlere (operators) giriş yapılmış ve bir programlama dilinde bulunan temel işlemler anlatılmıştır.

Ayrıca C dilinde işlemlere örnekler verilerek açıklanmış ve işlemlerin kullanıldığı yerler anlatılmıştır.

# Programlamaya Giriş Dersi

Bu dersin amacı, genel olarak bir program mantığının anlatılması, program geliştirme yöntemlerinin gösterilmesi ve karşılaşılan farklı durumlarda uygulanabilecek problem çözüm tekniklerinin öğretilmesidir. Bir programın, en basit seviyesinden ele alınarak, zaman ve hafıza ikilemi içerisinde, en verimli şekilde yazılmasının yollarının anlatıldığı derste, bir programın çalışma aşamalarından başlanarak, programlama dillerindeki işlemler, fonksiyonlar, değişken kavramı ve kullanımı, diziler, dizgiler ve bu konularda yazılabilecek algoritmalar işlenecektir. Derste ayrıca C dili üzerinden örnekler verilere, C yazım kurallarının öğrenilmesi hedeflenmektedir. Bu sayede C yazım kurallarının kullanıldığı JAVA, C++ veya C# gibi daha gelişmiş dillere geçiş kolaylığı hedeflenmektedir.





# 1. Hafta

Bu derste genel olarak bir “program”ın ne olduğundan bahsedip, basit bir programın nasıl yazılabileceğini anlatmaya çalışacağız. Ayrıca C dilinde basit bir program örneği verip, nasıl çalıştığını ve basit bir programın, bilgisayardaki çalıştırılma aşamalarını anlatacağız. Son olarak, ders kapsamında kullanacağımız DEV-CPP editör / derleyicisinin nasıl kullanılacağını gösterip, yazmış olduğumuz basit programı çalıştıracacağız.





# Bölüm İçeriği

- Programlamaya Giriş
  - Program Nedir?
  - Bilgisayarda Programlar Nasıl Çalışır?
  - C Diline Giriş
- Değişkenler
  - Değişkenler ve Hafıza Yönetimi
  - C Dilinde Değişken Kullanımı



## Temel Donanım (H/W)

- Klavye (Standart Giriş)
- Monitör (Standart Çıkış)
- RAM (Hafıza, Bellek, birincil hafıza)
- CPU ( İşlemci)
- HDD (Disk, ikincil hafıza)





# Temel Yazılım Öğeleri

- İşletim Sistemi (O/S)
- Kaynak Kod
- Makine Kodu
- Derleyiciler (Compilers)

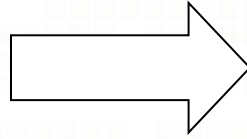




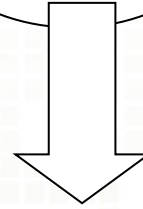
# Bir Programın Çalışması



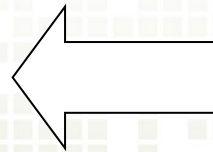
Programcı bir  
programlama dilinde  
kaynak kodu hazırlar



Derleyici kaynak  
kodu Makine  
Koduna çevirir



İşletim Sistemi  
Makine Kodunu  
hafızaya yükler



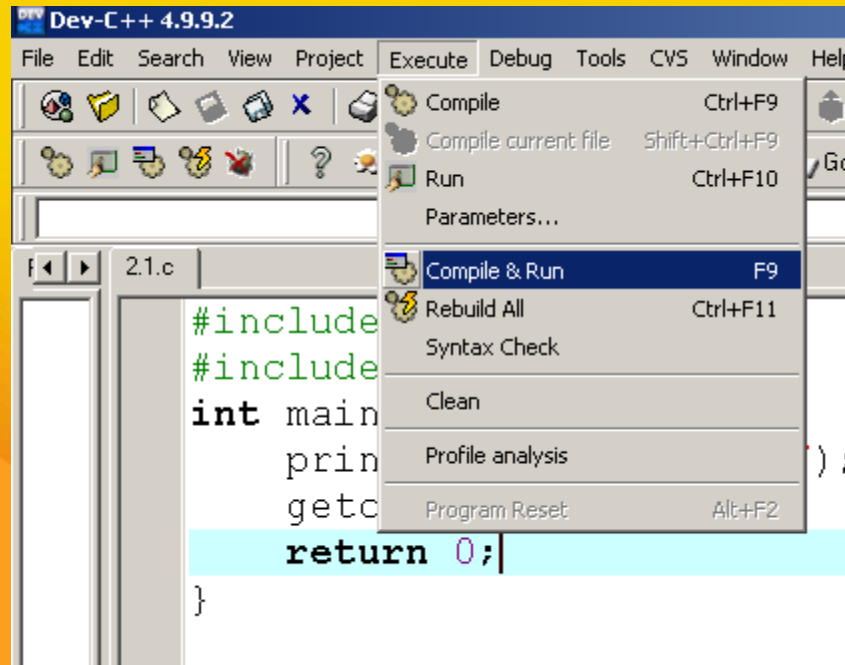
Yüklenen kod  
işletim sistemi  
tarafından CPU'da  
çalıştırılır



# Programlamaya Giriş (C Dili)

```
1. #include <stdio.h>
2. #include <conio.h>
3. int main(){
4. printf("Merhaba Dünya");
5. getch();
6. return 0;
7. }
```





Dev-CPP Programında Temel İşlemler

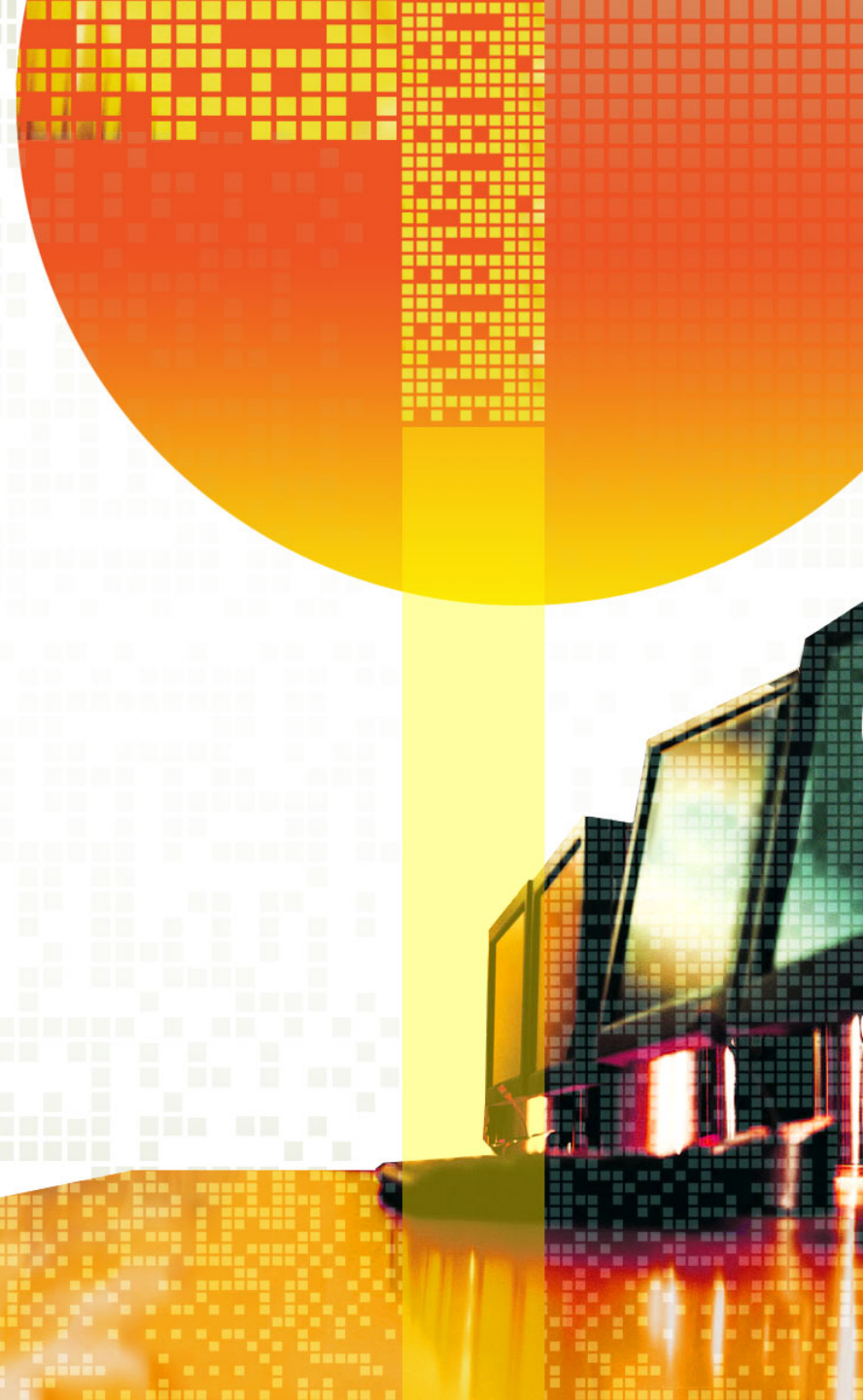
# Kodun ÇALIŞTIRILMASI





# Değişkenler

- Değişken İsmi
- Değişkenin Tipi
- Hafızadaki Yeri
- Değişkenin Değeri



# Değişken Tipleri

| Değişken tipi | Açıklama                      | Örnek |
|---------------|-------------------------------|-------|
| int           | Tam sayı değerleri tutar      | 10    |
| float         | Tek öncelikli ondalıklı sayı  | 3.14  |
| double        | Çift öncelikli ondalıklı sayı | 3.14  |
| char          | Karakter                      | 'a'   |



# C Dilinde Değişken Kullanımı

```
1. #include <stdio.h>
2. #include <conio.h>
3. int main(){
4. int a=10;
5. int b=3,c=4;
6. int d;
7. printf("a: %d, b: %d, c: %d\n",a,b,c);
8. printf("Bir sayi giriniz:");
9. scanf("%d",&d);
10. printf("\ngirilen sayi:%d",d);
11. getch();
12. return 0;
13. }
```



# C Dilinde, Değişkenlerin Standart Çıktıları

|             |                                                                                                                                                                                                                                                        |
|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>d, i</b> | İşareti bulunan onluk sistemdeki bir sayıyı ekrana basar. %d ile %i, printf için aynı anlamdadır                                                                                                                                                       |
| <b>u</b>    | Onluk tabanda işaretsiz bir tam sayıyı ekrana basar                                                                                                                                                                                                    |
| <b>f, F</b> | Ondalık sayıyı ekrana basmaya yarar. Küçük f ile büyük F harfleri arasındaki fark sayısal değerlerde anlaşılmaz. Bir değer boş olması veya sonsuz olması durumunda basılan değerlerdir.                                                                |
| <b>e, E</b> | Üstsel olarak bir sayıyı yazmaya yarar. Bilimsel olarak gösterilen sayıları 10 üzeri (exponent) şeklinde yazabiliriz. Örneğin 1234 sayısı $1.234 \times 10^3$ şeklinde yazılabilir. Bu gösterimde 10 yerine e harfi konulur ve ekrana 1.234e3 yazılır. |
| <b>x, X</b> | İşaretsiz bir tam sayı değerini onaltılık (hexadecimal) tabanda ekrana basmaya yarar. Büyük ve küçük x arasındaki fark basılan harflerin büyük veya küçük olmasıdır.                                                                                   |
| <b>o</b>    | İşaretsiz bir tam sayı değerini sekizlik (octal) tabanda ekrana basmaya yarar.                                                                                                                                                                         |
| <b>s</b>    | Ekrana bir yazıyı (string) basmaya yarar.                                                                                                                                                                                                              |
| <b>c</b>    | Ekrana bir karakter (character) basmaya yarar.                                                                                                                                                                                                         |
| <b>p</b>    | Print a void * (pointer to void) in an implementation-defined format.                                                                                                                                                                                  |
| <b>%</b>    | Ekrana sadece yüzde işareti % basmaya yarar.                                                                                                                                                                                                           |

# Özel Sembollerin Ekrana Basılması

| Sembol | Anlamı                                             |
|--------|----------------------------------------------------|
| \n     | Yeni satır                                         |
| \t     | Sekme karakteri (tab)                              |
| \b     | Geri silme karakteri (Son konulan karakteri siler) |
| \\     | Ekrana ters bölme (\) karakteri basmak için        |
| \"     | Ekrana tırnak (") karakteri basmak için            |

# Sonuç

Bu derste, genel olarak bir programın ne olduđuna ve nasıl çalıştıđına giriş yapılmıř. C dili ile basit bir uygulama geliřtirmenin adımları açıklanmıř, ayrıca C dilindeki deđiřken kavramı açıklanarak konu üzerinde örnekler verilmiřtir.

Ayrıca C dilindeki temel yazıma ve okuma işlemleri anlatılmıř ve örnekler üzerinden çalışması gösterilmiřtir.

