C’de Detaylı Üs Alma Programı

Bu uygulama yazısında C de pow() fonksiyonunu kullanmadan üs hesabı yapan programı yazmaya çalıştım. Başta tanımladığım float tipinde 2 fonksiyon sayesinde + lı ve – li üs değerleriyle işlem yapabiliyor. Gelecek yazılarda görüşmek üzere …

[crayon-5c6947e452cae924867685/]

Yazar : *Ömer Can ESKİCİOĞLU*

C Dilinde Diziler

C dilinde diziler çok kullanışlıdır. Diziler, aynı isim ve farklı indis numarası ile verileri tutabildiğimiz değişkenlerdir. Aynı tür verilerle çalışırken büyük kolaylık sağlar. Diziyi kaç elemanlı tanımlamışsak, o kadar değişken tanımlamışız gibi düşünülebilir.

Dizileri normal bir değişken gibi tanımlarız. Önce tipini, ardından ismini, son olarak da isminin yanına köşeli parantez içinde eleman sayısını yazarız.Örnek olarak:

int dizi[10]; //10 elemanlı dizi tanımladık.

int dizi[10] = {0}; //10 elemanlı dizi tanımlayıp, tüm elemanlarını "0"a eşitledik.
int dizi[] = {0, 1, 2, 3, 4}; //Dizi tanımlarken eleman sayısını belirtmedik, fakat diziye 5 adet değer atadığımız için dizi otomatik olarak beş elemanlı tanımlandı.
Dizileri tanımlarken tüm elemanlarını “0”a eşitlemek faydalı olabilir. Bunu yapmadığımız zaman RAM’de, dizinin tanımlandığı bölgede hangi bilgi varsa, ona göre işlem yapılır. Özellikle dizilerle aritmetik işlem yapıyorsak, bu durum istenmeyen sonuçlara yol açabilir. Örneğin 5 elemanlı bir dizi tanımladık ve elemanlarını sıfıra eşitlemedik. 3 elemana veri giriş yaptık, 2 elemana hiç dokunmadık. Elemanların tümünü bir for döngüsüyle topladık. Dokunmadığımız elemanlarda o an hangi değer varsa, o değerler de toplama dahil olacaktır.
Dizilerle çalışırken unutulmaması gereken; indislerin “0”dan başladığıdır. Örneğin yukarıda 10 elemanlı bir dizi tanımladık. Ancak bu dizinin sonuncu elemanının indir numarası “9”dur. Eğer “10” indis numaralı elemana erişmeye çalışırsak, RAM’de, tanımlamış olduğumuz diziden bir sonraki bölüme erişiriz ki bu da bize alakasız bir sonuç döndürecektir.
Dizileri değişkenlere benzetmiştik. Kullanırken de aynen değişkenler gibi kullanılır, yanına köşeli parantez içinde indis numarası yazılır.

Bir dizi çok sayıda değişken barındırdığından, bunları birbirinden ayırdetmek için *indis* adı verilen bir bilgiye ihtiyaç vardır. C Programlama Dili’nde, bir dizi hangi tipte tanımlanmış olursa olsun başlangıç indisi her zaman 0’dır.

Bir dizinin bildirim işleminin genel biçimi söyledir:
[crayon-5c6947e453d0c412036280/]
Örneğin, 5 elemanlı, kütle verilerini bellekte tutmak için, kutle dizisi şöyle tanımlanabilir:
[crayon-5c6947e453d16269498755/]
Bu dizinin elemanlarına bir değer atama işlemi şöyle yapılabilir:
[crayon-5c6947e453d1b097826374/]
Bildirim sırasında dizilerin eleman sayısı tamsayı türünden bir sabit ifadesiyle belirtilmesi zorunludur.

Bir diziye başlangıç değerleri aşağıdaki gibi kısa formda atanabilir:

[crayon-5c6947e453d27985886895/]

Küme parantezlerinin sonlandırıcı ; karakteri ile bittiğine dikkat ediniz.

Bir dizinin uzunluğu belirtilmeden de başlangıç değeri atamak mümkündür.

[crayon-5c6947e453d2d828075566/]

Derleyici bu şekilde bir atama ile karşılaştığında, küme parantezi içindeki eleman sayısını hesaplar ve dizinin o uzunlukta açıldığını varsayar. Yukarıdaki örnekte, a dizisinin 4, v dizisinin 5 elemanlı olduğu varsayılır.

printf ve scanf fonksiyonları bir dizinin okunması ve yazdırılması için de kullanılır. Örneğin bir A dizisinin aşağıdaki gibi bildirildiğini varsayalım:

[crayon-5c6947e453d32617321532/]

Bu dizinin elemanlarını klavyeden okumak için:

[crayon-5c6947e453d37879540392/]

daha sonra bu değerlerini ekrana yazmak için:

[crayon-5c6947e453d3b029307013/]
C Programlama Dili dizideki sayıları küçükten büyüğe sıralama işlemini en basit şekilde yapacağını program kodu. Bu kod ile kullanıcının girdiği sayıları küçükten büyüğe göre tekrardan diziye kaydedip ekrana yazdırabilirsiniz.

```
#include <stdio.h>
#include <stdlib.h>
// Dizideki sayıları sıralama (Küçükten büyüğe)
int main(){
int dizi[50], gecici, adet;

printf(“Kac adet sayi girilecek: “);
scanf(“%d”, &adet);  //Kaç adet sayı girileceğini kullanıcıdan alıyoruz

for(int i=0; i<adet; i++){  // Kullanıcıdan sayıları alıyoruz
printf(“%d)Sayi giriniz: “, i+1);
scanf(“%d”, &dizi[i]);
}

for(int i=0; i<adet–1;i++){
for(int j=i+1; j<adet; j++){
if(dizi[i] > dizi[j]){
gecici = dizi[i]; // Dizi[i] yi kaybetmemek için gecicide tutuyoruz
dizi[i] = dizi[j]; //dizi[i] yi dizi[i] den daha küçük olan dizi[j] kaydediyoruz
dizi[j] = gecici; // Dizi[j] ye ise dizi[i] değerini kaydediyoruz. Bu değeri gecicide saklamıştık
}
}
}
for(int i=0; i<adet; i++)
printf(“%d “, dizi[i]);

printf(“\n”);
system(“pause”);
return 0;
}
```

C Dilinde Yapı(Struct)

Yapı (Struct)

Birbirleriyle ilişkisi olan bir veri grubu C dilinde **yapı (structure)** adı verilen bir veri biçimi içinde saklanabilir. Yapı tanıtımı için struct kelimesi kullanılır. Yapı tek bir isim altında birbiri ile ilişkili ama farklı tiplerde olabilen verileri saklamaktadır. Yapının bileşenleri olan her biri farklı bir veri tipinde de olabilen verilere **yapının alanı (field)** ya da **üyeleri (members)** adı da verilir.

C dilinde yapı 3 farklı biçimde tanımlanabilir:

- Bir yapı grubunu tapı tip adı ile tanımlama
- Yapıları tek tek tanımlama
- Typedef yardımı ile tanımlama

Bir yapı grubunu yapı tip adı ile tanımlama

Örnek:
[crayon-5c6947e454445372019070/]
Burada personel, sicil,yas,cins,ad,soyad,brut alanlarından oluşan bir yapının tip adıdır, personel adı ile birlikte bir yer ayrılmaz personel adı sadece yukarıda belirtilen yapıyı simgeleyen bir tür veri sınıfının adıdır. Bellekte somut olarak yer ayıran değişkenler ise p1,p2,p3 adlı yapı değişkenleridir ve her biri personel tip adı ile tanımlanan yapı bileşenlerine sahiptirler.

Yapıya değer atama ve yapı elemanlarını yazdırma

Örnek:
[crayon-5c6947e45444d264735697/]
Programın Çıktısı:

Yapıları Tek Tek Tanımlama

Bir program içinde belirli bir tipten sadece bir iki tane yapı değişkeni kullanılacaksa bu durumda ayrı bir tip adı tanımlamaya gerek yoktur. Bu durumda yapı değişkenleri şu şekilde tanımlanabilir;

Örnek:
[crayon-5c6947e454453082972540/]
Burada iletken1 ve iletken2 adlı yapı değişkenleri float tipinde uyunluk,kesit, ve ozdirenc alanlarına sahip olarak tanımlanmıştır.

typedef Kullanımı ile Yapıları Tanımlama

typedef kelimesi kullanarak bir yapı tanımlanır ve sonrasında bu sınıftan yapı değişkenleri tanımlanabilir.

Örnek:
[crayon-5c6947e454457266760141/]

C Dilinde Rekürsif Fonksiyonlar

Kendisini çağırabilen işlevlere **rekürsif (özyineli)** işlev denir.

Bazı algoritmalar kendiliğinden rekürsiftir. Bunlardan en bilinenide faktöriyel algoritmasıdır. Matematikte, n sayısının faktöriyeli

n!=1.2.3……(n-1).n

şeklinde 1'den n 'e kadar tam sayıların çarpımı biçimindedir. Ayrıca 0!=1 olarak tanımlanmıştır.

Şimdi yukarıdaki ifadeyi

n!=1.2.3……n=F(n) biçiminde tanımlarsak bu durumda

(n-1)!=1.2.3..(n-1)=F(n-1) olarak yazabiliriz. Buna göre

(n-1)!=1.2.3..n=F(n)=1.2.3…(n-1).n=F(n-1).n ifadesine ulaşılacaktır. Bu durumda F(n)=F(n-1).n

İfadesi rekürsif bir ifadedir. Çünkü F(n) fonksiyonunun tanımlanması ve hesabı fonksiyonun kendisine referans verilerek gerçekleştirilmektedir.

Şimdi, yukarıdaki tanımlamaya göre hesaplamanın nasıl yapıldığına bakalım. Örnek olarak n=4 olsun.

Bu durumda, n=4 için

1. F(4)=F(3).4
2. F(3)=F(2).3
3. F(2)=F(1).2
4. F(1)=F(0).1

Adımları gerekecektir. Şimdi f(0)=0!=1 olarak tanımlı değere

ulaşıldığı için 4. Adımda F(1)=1 olarak hesaplanabilir. Sonra bir önceki adıma geçilerek F(2)=1.2=2 olarak hesaplanır. Sonra 2. Adıma dönülecek F(3)=3.2=6 olarak hesaplanacaktır. Sonuçta ise 1. Adıma dönülerek F(4)=6.4=24 olarak 4! İfadesinin sonucu bulunacaktır.

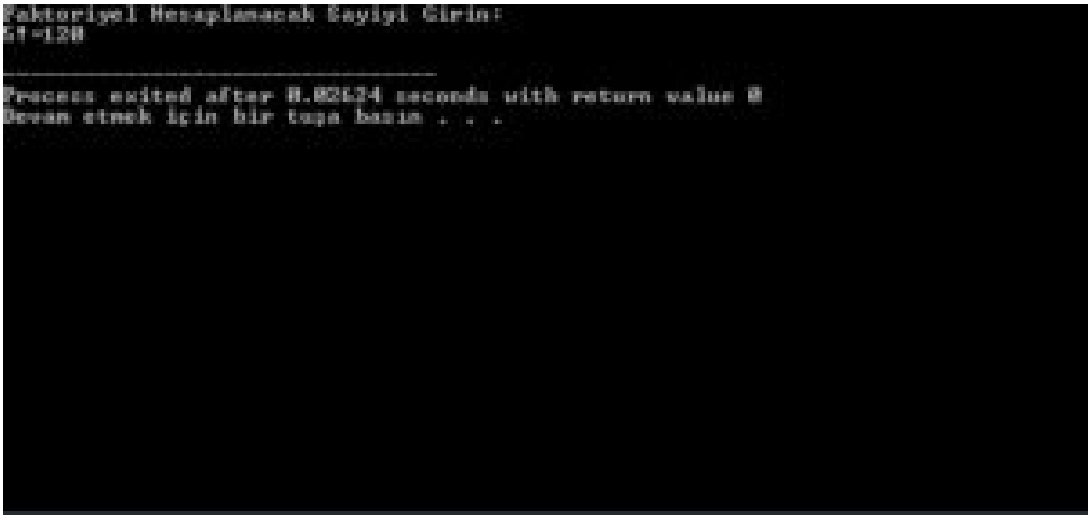
Böylelikle rekürsif bir algoritmada iki kısım mevcuttur.

1. Argümanın bir veya daha fazla değeri için fonksiyon değerinin belirlenmiş olduğu durum. Örnek olarak yukarıdaki algoritmada F(0)=0!=1 durumu. Bu durumada baz adı verilir.
2. İndüktif veya rekürsif adım. Yukarıdaki örnekte

F(n)=F(n-1).n durumu

Örnek:

C dilinde rekürsif faktöriyel program
[crayon-5c6947e4548c8883193780/]
Programın çıktısı



Fibonacci Sayılarının Üretilmesi

Fibonacci dizisi 1,1,2,3,5,8,13,21,34,55,89,144,233….. biçiminde tanımlanmış sonsuz dizidir.

F0=1 ve F1=1 alınarak

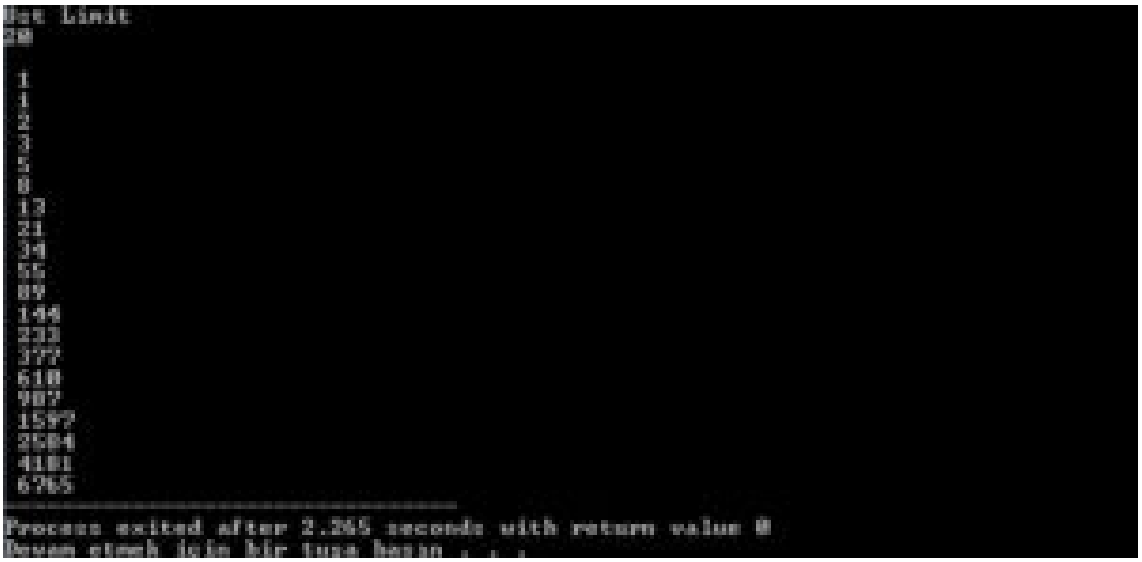
F2=F0+F1

F3=F1+F2

Ve genel olarak

Fi= Fi-1 +Fi-2 biçiminde tanımlanır. Fibonacci dizisini hesaplayan rekürsif bir C fonksiyonu örneği yapalım.

Örnek:
[crayon-5c6947e4548d0190915298/]
Programın çıktısı



C Dilinde Wordlist Oluşturma – 2

Geçen yazımda C de wordlist oluşturma programını geliştirerek bu programı oluşturdum. Wordlist havuzuna büyük harf, küçük

harf ve sayılar eklenmiştir.Oluşturulan wordlistlerde sondaki 3 sayı basamağı kullanıcının yapacağı ufak bir işleme göre değiştirilebilir.
[crayon-5c6947e454cc3962398513/]
Yazar : Ömer Can ESKİCİOĞLU

C Dilinde Wordlist Oluşturma – 1

Bu yazımda 25 elemanlı bir karakter dizisine rastgele harfler seçip istenilen harf sayısına ve istenilen kelime sayısı dahilinde wordlist oluşturmaya çalıştım. Eğer bu yazdığım kodu biraz daha geliştirebilirsem bundan sonraki c konsolunda bruteforce program denemesi yapacağım. Lütfen görüşlerinizi düşüncelerinizi yoruma eklemeyi unutmayın.
[crayon-5c6947e45544b340845414/]
Yazılan kod Windows işletim sisteminde Dev C++ editöründe denenmiştir. Eğer kodda hata devam ederse C sürümüne dikkat ediniz.

Yazar :Ömer Can ESKİCİOĞLU

e-posta:*eskiciogluomer@gmail.com*

Pointer Dönüşlü Fonksiyonlar

Bu konuyu incelemeden önce pointerlerin ne olduğunu bilmelisiniz ve Call by Value&Call by Reference konularını gözden geçirmiş olmalısınız, bu konunun anlaşılırlığı bakımından daha sağlıklı olacaktır.

Geri dönüşlü fonksiyonların parametre değerleri ve bir dönüş değeri olduğunu biliyoruz. Bu parametrelerin *pointer* bir değer olabileceği gibi dönüş tipimiz de bir *pointer* olabilir. Aşağıdaki kodumuzda bu konuyu inceleyelim.
[crayon-5c6947e455baf638139532/]
Main fonksiyonumuzda gördüğümüz gibi dizi[] arrayimizi tanımladık
[crayon-5c6947e455bb8567085289/]
Ardından *p pointeri adresDegistir(char *p, int index) fonksiyonunun (dizi,fazlalik) parametreleriyle return'u olarak tanımlanmıştır.
[crayon-5c6947e455bbd837444656/]

Fonksiyonumuzun parametrelerinin aldığı değerler *p ve index sırasıyla dizi string arrayi ve index yani 2 olmuştur.
[crayon-5c6947e455bc1854489713/]
Fonksiyonun başındaki * işareti ile de fonksiyonumuzun dönüş tipi pointer olmuştur
[crayon-5c6947e455bc5255277328/]

uzunluk ise strlen komutuyla p stringinin uzunluğu olarak tanımlanmıştır.

[crayon-5c6947e455bca257752644/]

Fonksiyonumuzun içindeki bu koşul ile stringimizin uzunluğu index(2)'ten büyük olduğu müddetçe dönüşümüz (p+index) kadar olacatır.Aksi halde NULL (boş) bir değer olacaktır. Bunun anlamı ise mesela p stringimizin adresinin 2 birim ilerlemiş olması ve main fonksiyonuna dönüşün pointer biçiminde olmasıyla main fonksiyonundaki
[crayon-5c6947e455bce150596437/]
çıktısının sonucu ise

"zilimagiris" olacaktır.

Hazırlayan Gürkan Aktaş.

C Dilinde Limit Kütüphanesi

"limits.h" başlığı "char", "short", "int", "long" gibi değişken tiplerinin en fazla ve en az değer aralıklarını belirleyen standart C++ kütüphanesidir. Örneğin işaretsiz bir karakteri 255 maksimum değere kadar saklayabilirsiniz. Aşağıdaki örneğe bakalım :
[crayon-5c6947e456125923040808/]
bu örneğin çıktısı şu şekildedir :
1 byte'ın bit sayısı = 8
Maximum SIGNED CHAR = -126
Maximum SIGNED CHAR = 127

```
Maximum UNSIGNED CHAR = 255
Maximum SHORT INT = -32766
Maximum  SHORT INT = 32767
Maximum  INT = -2147483648
Maximum  INT = 2147483647
Maximum  CHAR = -128
Maximum CHAR =127
Maximum LONG = -2147483648
Maximum LONG = 2147483647
```

C dilinde if-else yapısı

Merhaba arkadaşlar, bu başlık altında sizlere koşullu ifadelerden bahsedeceğim.

Peki nedir bu koşullu ifadeler(if, else if, else)?

Bunu bir koşulun gerçekleşmesine bağlı olarak gerçekleşen olaylar olarak açıklayabiliriz.

Örneğin; bugün arkadaşımla buluşacağım bu yüzden sinemaya gideceğiz.

Arkadaşımla buluşmam koşul, olayın gerçekleşmesi ise sinemaya gitmemiz aksi durumda sinemaya gitmeyeceğiz.

Bu ve buna benzer yapıları oluşturmak için if, else if, else komutları mevcut ve bugün bunlardan bahsedeceğim.

Öncelikle if, else kullanımından bahsedelim

1. **#include <stdio.h>**
2. **İnt main()**

```
3. {
4. if(koşul)
5. }
```

İlk olarak küme parantezi içerisinde if yazıp parantez içerisinde koşulumuzu belirtiyoruz

```
1. #include <stdio.h>
2. İnt main()
3. {
4. if(koşul)
5. { işlemler…
6.  }
7. }
```

Tekrar küme parantezi açarak gerçekleşecek işlemlerimizi yazıyoruz

Koşul gerçekleştiğinde eğer koşul gerçekleşirse, parantez içerisindeki işlemler gerçekleşecek

Aksi durumda olacak iş ise else yapısıyla sağlanır;

```
1. #include <stdio.h>
2. İnt main()
3. {
4. if(koşul)
5. { işlemler…
6.  }
```

```
7. else
8. {işlemler
9. }
10. }
```

İf, else yapısının en basit hali bu şekildedir.Koşul gerçekleşirse ilk blok, gerçekleşmezse ikinci blok çalışacaktır

Bir koşulu etkileyecek birden fazla durumda meydana gelebilir, bu esnada bu blokların nasıl kullanıldığına dair örneklerde vereceğim ama önce en basit haline bir örnek verelim.

```
1. #include<stdio.h>
2. int main( void )
3. {
4.  int s1;
5.  printf("Lütfen bir tam sayı giriniz: ");
6.  scanf("%d",&s1);
7.  if( s1 > 100 )
8.   printf("Girilen sayı 100'den büyüktür\n");
9.  else
10.   printf("Girilen sayı 100'den küçüktür\n");
11.  return 0;
12. }
```

Örnekte görüldüğü gibi, bir koşulun doğruluğunun if ile kontrolünü yaptırıp koşul karşılanıyorsa bir sonraki komut satırı devreye giriyor ve "Girilen sayı 100'den büyüktür" ekrana yazdırılıyor. Şayet verilen koşul yanlışsa, o zaman else satırı dikkate alınıyor ve ekrana "Girilen sayı 100'den

küçüktür” yazdırılıyor. Ancak ikisini de yapması gibi bir durum söz konusu değildir.

Birden fazla durum söz konusu olduğunda else if yapısını kullanabiliriz, ne kadar durum gerçekleşebilirse if yapısından sonra o kadar else if ekleyabiliriz;

Örneğin aynı boyda 4 kişiyi seçebiliriz ve bunları zayıf, normal, kilolu, aşırı kilolu olarak değerlendirebiliriz ve bu olayda 4 duruma sahip oluruz. Bunun gibi birden fazla gerçekleşebilecek durumlarda else if yapısını kullanırız

Bunu alttaki örnekte görebilirsiniz

```
1. #include<stdio.h>
2. int main( void )
3. {
4.  int k;
5.  printf(“kilonuzu giriniz: “);
6.  scanf(“%d”,&k);
7.  if( 50<k< 60 )
8.   printf(“zayifsiniz\n”);
9.  else if( 60<k<70 )
10.   printf(“normalsiniz\n”);
11.  else if( 70<k<80 )
12.   printf(“kilolusunuz\n”);
13.  else if( 80<k<90 )
14.   printf(“asiri kilolusunuz\n”);
15. return 0;
16. }
```

İç İçe if ve else deyimleri

C dilinde, bir if veya else deyimine bağlı olarak çalıştırılan işlem satırında yeni bir if deyimi yer alabilir. Yine bir if veya else deyimine bağlı olarak çalıştırılan kod bloğu içinde yer alan işlem satırlarından birisinde yeni bir if deyimi yer alabilir. Başka bir ifade ile, bir if deyimi başka bir if veya else deyiminin içinde kullanılabilir. İçte kalan if deyimi dıştaki if deyimine ait ifadenin doğru olması halinde programın çalıştırdığı tek bir satır olabileceği gibi, dıştaki if deyimine ait bir kod bloğunun işlem satırlarından biri de olabilir.

```
1. if(ifade)
2. if(ifade) işlem-satırı;
3. if(ifade) {
4. if(ifade) {      işlem-satırı;      işlem-satırı;
5. }
6. else {
7. işlem-satırı;
8. if(ifade) işlem-satırı;
9. else işlem-satırı;
10.  }
11. }
```

İf, if else , else if örnekleri;

Örnek if kullanımı :

Örnek programımızda klavyeden, bir tam sayı girilmesi istenmektedir. Ve bizde girilen sayı, 100’den büyükse koşulunu

vererek ekrana “Girilen sayı 100’den büyüktür” yazdırmaktayız.

```
1. #include<stdio.h>
2. int main( void )
3. { int s1; //girilen sayimiz//
4. printf(“Lütfen bir tam sayı giriniz: “);
5. scanf(“%d”,&s1);
6. if( s1 > 100 )
7. printf(“Girilen sayı 100’den büyüktür\n”);
8. return 0;
9. }
```

Örnek if else kullanımı :

Klavyeden girilen sayının tek mi çift mi olduğunu ekrana yazdıran program

```
1. #include <stdio.h>
2. #include <conio.h>
3. int main()
4. {
5. int a;
6. printf(“Sayiyi giriniz=”);
7. scanf(“%d”,&a);
8. if(a%2==0)
9. printf(“Girilen sayi cifttir.”);
10. else
11. printf(“Girilen sayi tektir.”);
12. getch();
13. return 0;
14. }
```

Örnek else if kullanımı;

Kullanıcıyı yasına göre çocuk, genç ve yaslı olarak sınıflandırma

1. #include <stdio.h>#include <conio.h>
   int main()

   {

   int yas;

   printf(“yasinizi giriniz:”);

   scanf(“%d”,&yas);

   if(yas<=17)

   {printf(“cocuk”);

   }

   else if(yas==18)

   {

   printf(“genc”);

   }

   else

   {

   printf(“yasli”);   }

   }

Hazırlayan Çağrı KONCA

C’de giriş-çıkış komutları

Tüm programlama dillerinde belirli bir formata göre ya da formatsız olarak programda kullanmak için giriş çıkış deyimleri kullanılır

C’de kullanılan giriş çıkış komutları şunlardır;

PRİNTF() KOMUTU

Programda kullandığımız değişken veya yazıcıya yazdırmak için kullanılan komuttur.Bu komut **stido.h** kütüphanesinde bulunmaktadır.

Kullanım Şekli;

Printf(“[mesaj] format_string”,değişken);

Mesaj: kullanılması zorunlu olmayan ama yapılan işlemle ilgili mesajları ekrana yazdıran kısımdır.

Format_string: yazdırlmak istenilen değişken ve sabitlerin veri türlerini belilemek için kullanılan karakter topluluğudur.Bilgini ekrana ne şekilde yazılacağını belirtir.

Format_string üç kısımdan oluşmaktadır:

Düz metin (literal string): yazdırılmak istenen ileti.
Örneğin: printf(“MERHABA DÜNYA”)gibi.

Konrol karakterleri (escape squence): değişkenlerin ve sabitlerin nasıl yazılacağını belirtmek veya imlecin alt satıra geçirilmesi gibi bazı işlemlerin gerçekleştirilmesi için kullanılır. Bu karakterler Tablo 4.1’de listelenmiştir.

Örneğin: printf(“MERHEBE\n DÜNYA”); gibi.

Tablo 4.1: *Kontrol karakterleri*

Karakter	Anlamı
\a	Ses üretir (alert)
\b	imleci bir sola kaydır (backspace)
\f	Sayfa atla. Bir sonraki sayfanın başına geç (formfeed)
\n	Bir alt satıra geç (newline)
\r	Satır başı yap (carriage return)
\t	Yatay TAB (horizontal TAB)
\v	Dikey TAB (vertical TAB)
\”	Çift tırnak karakterini ekrana yaz
\’	Tek tırnak karakterini ekrana yaz
\\	\ karakterini ekrana yaz
%%	% karakterini ekrana yaz

Tip belirleyici (conversion specifier): % işareti ile başlar ve bir veya iki karakterden oluşur (%d gibi). Ekrana yazdırılmak istenen değişkenin tipi, % işaretinden sonra belirtilir

Örneğin: printf(“x in değeri %d dir”); gibi.

Tablo 4.2: *Tip karakterleri*

Tip Karakteri	Anlamı	Yazdırılacak veri tipi
%c	tek bir karakter	Char
%s	karakter dizisi (string)	Char

%d	işaretli ondalık tamsayı	int, short
%ld	uzun işaretli ondalık tamsayı	Long
%u	işaretsiz ondalık tamsayı	unsigned int, unsigned short
%lu	işaretsiz uzun tamsayı	unsigned long
%f	Gerçel sayı	Float
%lf	Çift duayarlı gerçel sayı	Double

Scanf() KOMUTU

Değişkenlerin bellekteki alanlarına klavyeden değer girmek için kullanılır.stdio.h kütüphanesinde bulunur.

Kullanım şekli;

Scanf(“format_string”,değişken);

Örnekler:,

```
01:
02:
03:
04:
05:
06:
07:
08:
09:
10:
11:
12:
13:
14:
15:
16:
17:
18:
19:
20:
```

```
/* 04prg03.c
scanf() fonksiyonu ile int ve float
  tipindeki verilerin okunması */

#include <stdio.h>

main()
   {
int    t;
float  g;

printf("Bir gercel sayi girin: ");
scanf("%f",&g);
printf("Bir tamsayi girin    : ");
scanf("%d",&t);

printf("\n");

printf("\t %f * %f = %f\n",g,g,g*g);
printf("\t %d * %d = %d\n",t,t,t*t);

return 0;
   }
```

ÇIKTI

```
Bir gercel sayi girin: 1.34
 Bir tamsayi girin    : 12

1.340000 * 1.340000 = 1.795600
        12 * 12 = 144
```