C Programlama Dili

Dr. Öğr. Üye. Zafer Civelek

C Programlamaya Giriş

- C dili, bilgisayar programı tasarımına yapısal ve disiplinli bir yaklaşım için zemin hazırlamıştır. Bu bölümde, C programlamaya giriş yapılacak ve C'nin bir çok önemli özelliğini gösteren birkaç örnek sunulacak.

Bir Satır Yazı Yazma

- Basit bir C programı ile başlayalım. İlk örneğimiz bir satır yazıyı yazar. Program şu şekildedir.

```
1   // C'de ilk program
2   #include<stdio.h>
3
4   //main fonksiyonu çalışmaya başlar
5   int main(void)
6   {
7       printf("Merhaba!\n");
8   }//main fonksiyonu sonu
```

Ekran çıktısı ise şu şekildedir.

Bir Satır Yazı Yazma

- Bu program basit olduğu halde, C dilinin birkaç önemli özelliğini gösterir. 1. satır açıklama satırıdır.

```
// C'de ilk program
```

- Bu satırlar // ile başlar ve // satırların açıklama olduğunu gösterir. Program dosyasına açıklama ekleyerek, programın okunabilirliğini iyileştirmiş oluruz. Açıklamalar, program çalışırken dikkate alınmaz ve bilgisayarın bir şey yapmasına sebeb olmaz. Açıklamalar, C derleyicisi tarafından göz ardı edilir ve herhangi bir makine dili nesne kodu üretilmesine neden olmaz.

Bir Satır Yazı Yazma

- İlk açıklama programın amacını açıklıyor. Açıklamalar aynı zamanda başkalarının sizin programınızı okuyup anlamasına yardımcı olur.
- Ayrıca ilk satırdaki /*'dan son satırdaki */'ye kadar içerisinde bulundurduğu her şey açıklamadır. /*.....*/ çok satırlı açıklamalar için kullanılabilir.

#include Önişlemci Talimatı

2. satır

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

C önişlemcisine bir talimattır. # ile başlayan satırlar

Bir Satır Yazı Yazma

derlemeden önce önişlemci tarafından işlenir. 2. satır önişlemciye standart giriş/çıkış başlık (<stdio.h>) içeriğini programa eklemesini söyler. Derleme, printf (7. satır) gibi standart giriş/çıkış kütüphane fonksiyonlarını çağırdığı zaman, derleyici tarafından kullanılan bilgileri bu başlık ihtiva eder.

Boş Satırlar ve Boşluklar

3. satır sadece boş bir satırdır. Programı daha kolay okumak için boş satırlar, boşluk karakterleri ve sekme karakterleri kullanılır. Bu karakterlerin hepsi beyaz boşluk olarak bilinir.

Beyaz boşluk karakterleri normalde derleyici tarafından göz ardı edilir.

main Fonksiyonu

5. satır

```
int main(void)
```

main fonksiyonu, her C programının bir parçasıdır. main'den sonraki parantez main'in fonksiyon olarak adlandırılan bir program yapı taşı olduğuna işaret eder. C programları, bir veya daha fazla fonksiyon içerir. Bu fonksiyonların içerisinden bir tanesi mutlaka main

Bir Satır Yazı Yazma

olmak zorundadır. Her C programı yürütülmeye main fonksiyonundan başlar. Fonksiyonlar bilgi gönderebilir. main'in solundaki int anahtar kelimesi main'in bir tamsayı değer gönderdiğini gösterir. Fonksiyonlar yürütülmek için çağrıldığında aynı zamanda veri alabilirler. Parantez içindeki void kelimesi main'in herhangi bir bilgi almadığını gösterir.

iyi programlama uygulaması 2.1

Her fonksiyon, kendisinden önce fonksiyonun amacını tanımlayan bir açıklamaya sahip olmalıdır.

Bir Satır Yazı Yazma

Sol küme parantezi, {, bir fonksiyonun gövdesini başlatır (6. satır). Karşılık gelen sağ küme parantezi bir fonksiyonu sonlandırır (8. satır). Bu parantez çifti ve parantezler arasındaki program bir blok olarak isimlendirilir. Blok, C'de önemli bir program birimidir.

Bir Çıkış İfadesi

7. satır

```
printf("Merhaba!\n");
```

ifadesi, bilgisayara ünlem işareti ile sonlanan bir karakter dizisini ekrana bastırmak için , yani bir eylem

Bir Satır Yazı Yazma

yapması için bilgisayara talimat verir. Bir karakter dizini bazen bir karakter katarı, bazen mesaj veya bir söz olarak adlandırılır. printf fonksiyonu («f» «biçimlendirilmiş»'e karşılık gelir), parantez içindeki argümanı ve noktalı virgül (;) dahil tüm satıra ifade denir. Her ifade noktalı virgül ile (aynı zamanda ifade sonlandırıcı olarak da bilinir) sonlanmak zorundadır. printf ifadesi yürütüldüğü zaman, ekrana Merhaba! mesajı yazılır. Karakterler normalde printf ifadesindeki çift tırnaklar arasında tam olarak göründükleri gibi basılır.

Bir Satır Yazı Yazma

Kaçış Dizinleri

\n karakteri ekrana basılmamıştır. Ters bölü (\) kaçış karakteri olarak adlandırılır. printf'in sıra dışı bazı şeyler yapabileceğini belirtir. Bir karakter dizisinde ters bölü ile karşılaşıldığında, derleyici devam ederek sonraki karaktere bakar ve ters bölü ile birleştirerek bir kaçış dizisi haline getirir. \n kaçış dizisi yeni satır anlamındadır. printf karakter katarında yeni satır varsa, yeni satır imlecin ekranda sonraki satır başına konumlanmasına neden olur.

Bir Satır Yazı Yazma

Bazı yaygın kullanılan kaçış dizinleri şekilde gösterilmiştir.

Kaçış dizisi	Tanım
\n	Yeni satır. İmleci sonraki satır başına konumlandırır.
\t	Yatay sekme. İmleci sonraki sekme noktasına kaydırır.
\a	Uyarı. İmlecin o andaki konumunu değiştirmeden bir ses veya görülebilir uyarı üretir.
\\	Ters bölü. Dizi içine ters bölü karakterini yerleştirir.
\”	Çift tırnak. Dizi içine çift tırnak karakteri yerleştirir.

Bir Satır Yazı Yazma

Bir karakter dizininde ters bölünün özel bir anlamı olduğundan, derleyici onu kaçış karakteri olarak algılar. Bu yüzden bir karakter katarına ters bölü ilave etmek için çift ters bölü (\\) kullanırız. Çift tırnak bir karakter dizisinin sınırlarını belirlediğinden çift tırnak basmak da aynı zamanda sorun oluşturur. printf çıkışı için bir karakter dizisinde \" kaçış dizisi kullanarak, printf'in bir çift tırnak göstermesi gerektiğini belirtiriz. Sağ küme parantezi, }, (8.satır) main'in sonuna ulaşıldığını ifade eder.

Bir Satır Yazı Yazma

iyi programlama uygulaması 2.2

main dahil her fonksiyonu kapatan sağ küme parantez, }, olan satıra açıklama ekleyin.

Bağlayıcı ve Yürütülebilirler

printf ve scanf gibi standart kütüphane fonksiyonları C programlama dilinin parçaları değildir. Mesela, derleyici printf ve scanf'deki yazım hatalarını bulamaz. Derleyici printf ifadesini derlerken, yalnızca kütüphane fonksiyonuna «çağrı» yapmak için nesne programında boşluk bırakır. Ancak derleyici kütüphane fonksiyonlarının nerede olduğunu bilmez, bağlayıcı bilir.

Bir Satır Yazı Yazma

Bağlayıcı çalıştığında kütüphane fonksiyonlarını konumlandırır ve uygun çağrıları nesne programındaki bu kütüphane fonksiyonlarına yerleştirir. Nesne programı bu durumda tamamlanmış durumdadır ve yürütülmeye hazırdır. Bu nedenle bağlanmış programa yürütülebilir denir.

Yaygın Programlama Hatası 2.1

printf çıkış fonksiyonunun ismini bir program içerisinde örneğin print gibi yanlış yazmak.

Bir Satır Yazı Yazma

İyi programlama Uygulaması 2.3

Fonksiyonun gövdesini tanımlayan küme parantezleri arasındaki tüm gövdeyi bir sekme içeriye kaydırın (yaklaşık 3 boşluk karakteri). Bu boşluklar programın fonksiyonel yapısını vurgular ve programların daha kolay okunmasına yardımcı olur.

Bir Satır Yazı Yazma

Çoklu printf'ler kullanma

Printf fonksiyonu Merhaba!'yı birkaç farklı şekilde yazabilir. Mesela şu program;

```
1  // İki printf ifadesiyle bir satırda yazı yazma.
2  #include<stdio.h>
3
4  //main fonksiyonu çalışmaya başlar
5  int main(void)
6  {
7      printf("Mer");
8      printf("haba!\n");
9  }//main fonksiyonunun sonu
```

Bir önceki programla aynı çıktıyı üretir.

Bir Satır Yazı Yazma

Her bir printf bir önceki printf'in yazmayı bırakdığı yerden devam ettiği için bu program çalışır. İlk printf (7. satır) Mer yazar ikinci printf (8. satır) aynı satırdan haba! yazmaya başlar.

Tek bir printf yeni satır karakteri ilavesiyle birkaç satır yazabilir. \n kaçış dizisiyle her karşılaşıldığında, çıktı sonraki satırın başlangıcından devam eder.

```
1  //Tek bir printf ile çoklu satır yazma
2  #include<stdio.h>
3
4  //main fonksiyonu çalışmaya başlar
5  int main(void)
6  {
7      printf("Merhaba\narkadaslar\nnasilsiniz?");
8  }//main fonksiyonu sonu
```

Bir Satır Yazı Yazma

```
Merhaba
arkadaslar
nasilsiniz?
```

İki tam sayıyı toplama programı

Bu programımız kullanıcının klavyeden yazdığı iki tam sayıyı alan standart kütüphane fonksiyonu scanf'i kullanır, bu değerlerin toplamını hesaplar ve printf kullanarak sonucu yazdırır. Program şu şekildedir;

İki Tam Sayıyı Toplama

```c
//Toplama programı
#include<stdio.h>

//main fonksiyonu çalışmaya başlar
int main(void)
{
int tamsayi1;//kullanıcı tarafından girilecek ilk sayı
int tamsayi2;//kullanıcı tarafından girilecek ikinci sayı
int toplam;//toplamın saklanacağı değişken

printf("ilk tam sayiyi girin\n");//istek
scanf("%d",&tamsayi1);//bir tamsayı oku

printf("ikinci tam sayiyi girin\n");//istek
scanf("%d",&tamsayi2);//bir tamsayı oku

toplam=tamsayi1+tamsayi2;//toplamı aktar

printf("Toplam=%d\n",toplam);//toplamı yaz
}//main fonksiyonu sonu
```

```
ilk tam sayiyi girin
45
ikinci tam sayiyi girin
72
Toplam=117
```

İki Tam Sayıyı Toplama

Değişkenler ve değişken tanımlamaları

7-9. satırlar

```
7   int tamsayi1;//kullanıcı tarafından girilecek ilk sayı
8   int tamsayi2;//kullanıcı tarafından girilecek ikinci sayı
9   int toplam;//toplamın saklanacağı değişken
```

tanımlamalardır. tamsayi1, tamsayi2 ve toplam değişkenlerin isimleridir. Bu tanımlamalar tamsayi1, tamsayi2 ve toplam değişkenlerinin, tam sayı değer alacağı anlamına gelen **int** türü olduklarını belirtir. Bir programda kullanılmadan önce tüm değişkenler bir isim ve bir veri türü ile tanımlanmak zorundadır.

İki Tam Sayıyı Toplama

Değişken tanımlamaları tek bir tanım ifadesinde birleştirilebilir. Mesela

int tamsayi1,tamsayi2,toplam;

Ancak bu tek satırlık ifade değişkenlerin yanlarına yaptığımız açıklama ifadelerini zorlaştırır.

Kimlikler ve Büyük Küçük Harf Duyarlılığı

C'de bir değişken ismi her hangi geçerli bir kimliktir. Bir kimlik, bir sayı ile başlamayan, harf, sayı ve alt çizgiden (_) oluşan karakter dizisidir. C büyük küçük harf duyarlıdır. Yani büyük ve küçük harfler C'de farklıdır.

İki Tam Sayıyı Toplama

Yaygın programlama hatası 2.2

Küçük harf kullanılması gereken yerde büyük harf kullanma (mesela main yerine Main yazma)

Hata önleme önerisi 2.1

Derleyici tarafından oluşturulan kimlikler ve standart kütüphane kimlikleri ile çakışmayı önlemek için alt çizgi karakteri (_) ile başlayan kimliklerden sakının.

İyi programlama uygulaması 2.4

Anlamlı değişken isimleri seçmek bir programın kendi kendine belgelemesine yardımcı olur, daha az açıklamaya ihtiyaç duyulur.

İki Tam Sayıyı Toplama

İyi programlama uygulaması 2.5

Basit bir değişken ismi olarak kullanılan bir kimliğin ilk harfi küçük olmalıdır.

İyi programlama uygulaması 2.6

Çok kelimeli değişken isimleri programın daha iyi okunabilir olmasına yardımcı olur. toplam_deger'deki gibi kelimeleri alt çizgi ile ayırın veya, kelimeleri yan yana koyarsanız, toplamDeger'deki gibi birinciden sonraki her kelimeyi büyük harf ile başlatın.

İki Tam Sayıyı Toplama

scanf fonksiyonu ve biçimlendirilmiş girişler

scanf(“%d”,&tamsayii);//bir tam sayı oku

ifadesi kullanıcıdan bir değer almak için scanf kullanır. Fonksiyon genelde standart giriş birimi olan klavyeden okur. Burada scanf’in “%d” ve &tamsayii şeklinde iki argümanı vardır. Birincisi, biçim kontrol karakter katarı, kullanıcı tarafından girilecek veri türünü belirler. %d çevrim belirteci verinin bir tam sayı olması gerektiğini belirtir. scanf’in ikinci argümanı değişken isminin önünde “ve işareti” (&) ile başlar, adres işlemi denir.

İki Tam Sayıyı Toplama

Değişken ismi ile birlikte &, tamsayıı değişkeninin saklandığı hafızadaki yerini scanf'e bildirir. Sonra bilgisayar kullanıcısının tamsayıı için girdiği değeri burada saklar.

İyi programlama uygulaması 2.7

Programları daha okunaklı yapmak için her virgülden sonra boşluk bırakın.

Bilgisayar scanf'i çalıştırdığı zaman, kullanıcıdan tamsayıı değişkenine bir değer girmesi için bekler. Kullanıcı bir tam sayı yazar, sonra da sayıyı bilgisayara göndermek için Enter tuşuna basarak karşılık verir.

İki Tam Sayıyı Toplama

Bilgisayar daha sonra bu sayıyı tamsayi1 değişkenine aktarır. Bu programda bundan sonra tamsayi1 değişkeninin geçtiği her yerde aynı değer kullanılacaktır.

Atama ifadeleri

17. satırdaki atama ifadesi

toplam=tamsayi1+tamsayi2;//toplamı aktar

tamsayi1 ve tamsayi2 değişkenlerinin toplamını hesaplar ve atama = işlemi kullanarak sonucu toplam değişkenine aktarır.

İki Tam Sayıyı Toplama

Biçim kontrol karakter katarı ile yazma

19. satır

printf(''Toplam=%d\n'', toplam);//toplamı yaz

Toplam karakter dizinini takip eden toplam değişkeninin sayısal değerini ekrana yazmak için printf fonksiyonunu çağırır. Bu printf'in ''Toplam=%d\n'' ve toplam olmak üzere iki argümanı vardır. İlk argüman biçim kontrol karakter katarıdır. Gösterilecek bazı sözel karakterleri ve bir tam sayı yazılacağını belirten %d çevrim belirtecini içerir.

İki Tam Sayıyı Toplama

printf ifadelerindeki hesaplamalar

Hesaplamalar aynı zamanda printf ifadesi içerisinde de gerçekleştirilebilir. Mesela;

printf("Toplam=%d\n", tamsayi1 + tamsayi2);

Hafıza Kavramları

tamsayi1, tamsayi2 ve toplam gibi değişken isimleri aslında bilgisayar hafızasındaki yerlere karşılık gelir. Her değişkenin bir ismi, bir türü ve bir değeri vardır.

toplama programında,

scanf("%d" , &tamsayi1);

ifadesi yürütüldüğünde, kullanıcı tarafından girilen değer tamsayi1 isminin atanmış olduğu hafıza konumuna yerleştirilir. Kullanıcının tamsayi1 değeri olarak 45 sayısını girdiğini farz edelim. Bilgisayar 45'i tamsayi1 konumuna yerleştirir.

Hafıza Kavramları

Hafıza konumuna ne zaman bir değer yerleştirilirse, değer konumun önceki değeri ile yer değiştirir.

scanf("%d", tamsayi2);

ifadesi yürütüldüğünde , kullanıcının 72 değerini girdiğini farz edelim. Bu değer hafızada tamsayi2 konumuna yerleştirilir. Bu konumlar hafızada ardışık olmak zorunda değildir.

Program tamsayi1 ve tamsayi2 değerlerini elde ettikten sonra bu değerleri toplar ve toplam değişkenine aktarır. Toplamayı yapan

Hafıza Kavramları

toplam=tamsayi1+tamsayi2;

ifadesi aynı zamanda toplam'da saklanan değeri değiştirir.

C'de Aritmetik

Çoğu C programı hesaplamaları C aritmetik işlemlerini kullanarak yapar.

C işlemi	Aritmetik işlem	Cebirsel ifade	C ifadesi
Toplama	+	f+7	f + 7
Çıkarma	-	p - c	p - c
Çarpma	*	bm	b*m
Bölme	/	x/y	x/y
Kalan	%	r mod s	r % s

C’de Aritmetik

Tabloda gösterildiği gibi, asteriks işareti (*) çarpmayı, yüzde işareti (%) bölmede kalanı gösterir.

Tam sayı bölme ve kalan işlemi

Tam sayı bölme tam sayı bir sonuç üretir. Mesela 7/4 ifadesi 1 değerine, 17/5 ifadesi 3 değerine sahiptir. C’de tam sayı bölme işleminden sonra kalanı gösteren kalan işlemi (%) vardır. x % y ifadesi, x, y’ye bölündükten sonra kalanı sonuç olarak gönderir. Bu durumda 7 % 4, 3 ve 17 % 5, 2 değerine sahip olur.

C'de Aritmetik

Yaygın programlama hatası 2.3

Sıfıra bölmeye çalışmak normalde bilgisayar sistemlerinde tanımsızdır ve genelde sonlandırıcı hata ile sonuçlanır. Yani programın görevini başarılı bir şekilde bitirmeden durmasına neden olan bir hata. Sonlandırıcı olmayan hatalar programın genelde doğru olmayan sonuç üreterek tamamlanmasına izin verirler.

Düz satır halinde aritmetik ifadeler

C'de aritmetik ifadelerin programlarını bilgisayara girmeyi sağlamak için düz satır halinde yazmak gerekir.

C’de Aritmetik

Bu nedenle, a bölü b gibi ifadeler a/b gibi yazılmak zorundadır. Böylece tüm işlemler ve terimler düz bir satır içinde gözükür.

Alt ifadeleri birleştirmek için parantezler

Parantezler, cebrik ifadelerde nasıl kullanılıyorsa C ifadelerinde de aynı şekilde kullanılır. Mesela a ile b + c ifadesini çarpmak için a * (b + c) yazılır.

İşlem önceliği kuralları

C aritmetik ifadelerdeki işlemleri genelde cebirdeki ile aynı olan işlem önceliği kuralları tarafından belirlenmiş kesin bir sıra içerisinde yapar.

C’de Aritmetik

1. Önce parantez içerisindeki işlemler yapılır. Parantezler en yüksek önceliktedir. ((a+b)+c) gibi iç içe parantezler durumunda , en içteki parantez çifti içerisindeki işlemler önce yapılır.

2. Sonra çarpma, bölme ve kalan işlemleri yapılır. Bir ifade birden fazla çarpma, bölme ve kalan işlemi içeriyorsa, işlem soldan sağa doğru yapılır. Çarpma, bölme ve kalan aynı öncelik seviyesindedirler.

3. Sonra toplama ve çıkarma işlemleri yapılır. Bir ifade birden fazla toplama ve çıkarma işlemi içeriyorsa, işlem soldan sağa doğru yapılır.

C'de Aritmetik

4. Son olarak atama (=) işlemi yapılır.

İşlem	İşlem	İşlem sırası (öncelik)
()	Parantezler	İlk önce yapılır. İç içe parantezler durumunda , en içteki parantez çifti içerisindeki işlemler önce yapılır.
*	Çarpma	İkinci olarak yapılır. Bir ifade birden fazla çarpma, bölme ve kalan işlemi içeriyorsa, işlem soldan sağa doğru yapılır.
/	Bölme	
%	Kalan	
+	Toplama	Üçüncü olarak yapılır. Bir ifade birden fazla toplama ve çıkarma işlemi içeriyorsa, işlem soldan sağa doğru yapılır.
-	Çıkarma	
=	Atama	Son olarak yapılır.

C'de Aritmetik

Örnek cebir ve C ifadeleri

Cebir: $m = \frac{a+b+c+d+e}{5}$

C: m=(a+b+c+d+e)/5;

Bölme, toplamadan daha yüksek önceliğe sahip olduğu için toplamayı gruplandırmak için parantezler gereklidir. Tüm (a+b+c+d+e) ifadesi 5'e bölünmeli. Eğer parantezler yanlışlıkla unutulursa a+b+c+d+e/5 ifadesi elde edilir ki bu da $a + b + c + d + \frac{e}{5}$ şeklinde yanlış işlem yapar.

Cebir: y=mx+b

C: y=m*x+b;

Burada parantez gerekli değildir. Çarpma toplamadan daha yüksek önceliğe sahip olduğu için önce çarpma yapılır.

Aşağıdaki ifade kalan, çarpma, bölme, toplama, çıkarma ve atama işlemlerini içerir.

Cebir: z=pr%q+w/x-y

C: z = p * r % q + w / x - y

6 1 2 4 3 5

C’de Aritmetik

Alt satırdaki sayılar C’nin yaptığı işlemlerin sırasını gösterir. Önce çarpma, bölme ve kalan soldan sağa doğru yapılır. Sonra toplama ve çıkarma soldan sağa doğru yapılır. En son atama işlemi yapılır.

İkinci derece polinomların hesaplanması

```
y = a * x * x + b * x + c;
  6   1   2   4   3   5
```

İfadenin altındaki sayılar C’nin işlemleri hangi sırada yaptığını gösterir.

C'de Aritmetik

İkinci derece polinomda a, b, c ve x değişkenlerinin şu değerlere sahip olduğunu farz edelim. a=2, b=3, c=7 ve x=5 olsun. Bu durumda hesaplama sırası şöyle olur.

1. adım y = 2 * 5 * 5 + 3 * 5 + 7;

2 * 5

2. adım y = 10 * 5 + 3 * 5 + 7;

10 * 5

3. adım y = 50 + 3 * 5 + 7;

3 * 5

C'de Aritmetik

4. adım y = 50 + 15 + 7;

50 + 15

5. adım y = 65 + 7;

65 + 7

6. adım y=72

Karar Verme: Eşitlik ve İlişkisel İşlemler

Karar verme: Eşitlik ve ilişkisel işlemler

Yürütülebilir ifadeler ya eylem yapar (hesaplamalar veya veri giriş çıkışı gibi) yada karar verir. Bir programda karar verebiliriz. Mesela bir kişinin bir imtihandan aldığı notun 50'den büyük olup olmadığını ve programın ''Dersten geçtiniz'' mesajı yazıp yazmayacağına karar vermek gibi. Koşul olarak isimlendirilen ifadelerin doğruluk veya yanlışlığına dayanarak karar vermeye yarayan if ifadeleri vardır. Eğer koşul doğruysa if'in gövdesindeki ifade yürütülür. Eğer koşul yanlışsa gövde ifadesi yürütülmez.

Karar Verme: Eşitlik ve İlişkisel İşlemler

Gövde ifadesi yürütülse de yürütülmese de, if ifadesi tamamlandıktan sonra yürütme if ifadesinden sonraki ifade ile devam eder.

İf ifadesindeki koşullar eşitlik işlemi ve ilişkisel işlemlerdir. İlişkisel işlemlerin hepsi aynı öncelik seviyesine sahiptir ve soldan sağa doğru işlenir. Eşitlik işlemi ilişkisel işlemlerden daha düşük önceliğe sahiptir ve soldan sağa doğru işlenir. C'de bir koşul sıfır (yanlış) veya sıfırdan farklı (doğru) bir değer üreten herhangi bir ifade olabilir.

Karar Verme: Eşitlik ve İlişkisel İşlemler

Cebrik eşitlik veya ilişkisel işlem	C eşitlik veya ilişkisel işlem	C koşul örneği	C koşulunun anlamı
Eşitlik işlemleri			
=	==	x == y	x, y'ye eşittir
≠	!=	x != y	x, y'ye eşit değildir
İlişkisel işlemler			
>	>	x > y	x, y'den büyüktür
<	<	x < y	x, y'den küçüktür
≥	>=	x >= y	x, y'den büyüktür veya eşittir
≤	<=	x <= y	x, y'den küçüktür veya eşittir

Karar Verme: Eşitlik ve İlişkisel İşlemler

Yaygın programlama hatası 2.4

==, !=, >= ve <= işlemlerinden herhangi birindeki iki sembol boşlukla birbirinden ayrılırsa, bir söz dizim hatası meydana gelir.

Yaygın programlama hatası 2.5

== işlemini atama işlemi ile karıştırmak.

Kullanıcı tarafından girilen iki sayıyı karşılaştırmak için if ifadesinin kullanımına örnek:

Karar Verme: Eşitlik ve İlişkisel İşlemler

```
1   // if ifadelerini, ilişkisel
2   // işlemleri ve eşitlik işlemlerini kullanma
3   #include<stdio.h>
4
5   //main fonksiyonu program yürütmesine başlar
6   int main (void)
7   {
8       int num1;//kullanıcıdan okunacak ilk sayı
9       int num2;//kullanıcıdan okunacak ikinci sayı
10
11      printf("iki tam sayi girin..\n");
12      printf("size sayilar hakkinda bilgi verecegim..:");
13
14      scanf("%d%d",&num1,&num2);//iki tam sayı oku
15
16      if(num1==num2){
17          printf("%d sayisi %d sayisina esittir\n",num1,num2);
18      }//if sonu
19
20      if(num1!=num2){
21          printf("%d sayisi %d sayisina esitDegildir\n",num1,num2);
22      }//if sonu
23
```

Karar Verme: Eşitlik ve İlişkisel İşlemler

```
24      if(num1<num2){
25          printf("%d sayisi %d sayisindan kucuktur\n",num1,num2);
26      }//if sonu
27
28      if(num1>num2){
29          printf("%d sayisi %d sayisindan buyuktur\n",num1,num2);
30      }//if sonu
31
32      if(num1<=num2){
33          printf("%d sayisi %d sayisindan kucuk yada esittir\n",num1,num2);
34      }//if sonu
35
36       if(num1>=num2){
37          printf("%d sayisi %d sayisindan buyuk yada esittir\n",num1,num2);
38      }//if sonu
39
40      }//main fonksiyonu sonu
```

Karar Verme: Eşitlik ve İlişkisel İşlemler

```
iki tam sayi girin..
size sayilar hakkinda bilgi verecegim..:3 7
3 sayisi 7 sayisina esitDegildir
3 sayisi 7 sayisindan kucuktur
3 sayisi 7 sayisindan kucuk yada esittir
```

```
iki tam sayi girin..
size sayilar hakkinda bilgi verecegim..:22 12
22 sayisi 12 sayisina esitDegildir
22 sayisi 12 sayisindan buyuktur
22 sayisi 12 sayisindan buyuk yada esittir
```

```
iki tam sayi girin..
size sayilar hakkinda bilgi verecegim..:7 7
7 sayisi 7 sayisina esittir
7 sayisi 7 sayisindan kucuk yada esittir
7 sayisi 7 sayisindan buyuk yada esittir
```

Karar Verme: Eşitlik ve İlişkisel İşlemler

Program iki sayıyı girmek için scanf kullanır. Her bir çevrim belirtecinin bir değerin saklandığı karşılığı olan bir argümanı vardır. İlk %d numı değişkeninde saklanan değeri çevirir ve ikinci %d num2 değişkeninde saklanan değeri çevirir.

İyi programlama uygulaması 2.8

İzin verilmesine rağmen, programın bir satırında bir ifadeden daha fazlası olmamalı.

Yaygın programlama hatası 2.6

Bir scanf ifadesindeki biçim kontrol karakter katarındaki çevrim belirteçleri arasına virgül koymak.

Karar Verme: Eşitlik ve İlişkisel İşlemler

Sayıları karşılaştırmak

16-18 satırlardaki if ifadesi

```
if(num1==num2){
   printf("%d sayisi %d sayisina esittir\n",num1, num2);
        }//if sonu
```

Eşitliği test etmek için numı ve num2 değişkenlerinin değerlerini karşılaştırır. Değerler eşitse 17. satırdaki ifade, sayıların eşit olduğunu söyleyen yazıyı gösterir. Diğer if ifadeleri de bu şekilde icra edilir. Her bir if ifadesinin gövdesinde boşluk bırakma ve her bir if ifadesinin üstünde ve altında bir satır boşluk bırakma

Karar Verme: Eşitlik ve İlişkisel İşlemler

programın okunabilirliğini iyileştirir.

Yaygın programlama hatası 2.7

Bir if ifadesindeki koşuldan sonraki sağ parantezden hemen sonra noktalı virgül koymak.

Sol küme parantezi, {, her if ifadesinin gövde başlangıcıdır. Karşılık gelen sağ küme parantezi, }, if ifadesinin gövde sonudur. Bir if ifadesinin gövdesine yerleştirilecek ifade sayısı için sınırlama yoktur.

Karar Verme: Eşitlik ve İlişkisel İşlemler

İşlemler	İşlem yönü
()	Soldan sağa
* / %	Soldan sağa
+ -	Soldan sağa
< <= > >=	Soldan sağa
== !=	Soldan sağa
=	Sağdan sola

Tablo işlemlerin önceliğini en yüksekten en düşüğe listeler. İşlemler yukarıdan aşağıya azalan öncelik sırasında gösterilmiştir. Eşit işareti aynı zamanda atama işlemidir. = atama işlemi dışındaki tüm işlemler soldan sağa doğru yapılır. Atama işlemi sağdan sola işler.

Karar Verme: Eşitlik ve İlişkisel İşlemler

İyi programlama uygulaması 2.9

Bir çok işlem içeren ifade yazarken işlem önceliği tablosuna bakın. İfadedeki işlemlerin uygun sırada uygulandığından emin olun. Eğer emin değilseniz, ifadeleri gruplamak veya birkaç basit ifadeye ayırmak için parantezler kullanın.

Bazı kelimeler int ve if gibi dilin anahtar kelimeleri veya ayrılmış kelimelerdir. Bunları değişken isimleri gibi kimlik olarak kullanmamak gerektir. Bu kelimelerin bazıları tabloda gösterilmiştir.

Anahtar kelimeler

Anahtar kelimeler			
auto	double	int	struct
break	else	long	switch
case	enum	register	typedef
char	extern	return	union
const	float	short	unsigned
continue	for	signed	void
default	goto	sizeof	volatile
do	if	static	while
C99 standardına ilave edilen anahtar kelimeler			
_Bool restrict	_Complex	_Imaginary	inline
C11 taslak standardına ilave edilen anahtar kelimeler			
_Alignas _Alignof _Atomic _Generic _Noreturn _Static_assert _Thread_local			