Bilişim Teknolojileri ve Yazılım Dersi

2. Dönem Çalışma Notları- Ünite 3

Operatör Türleri

- **Aritmetik Operatörler:**
    - Toplama (+): İki sayıyı toplar.
    - Çıkarma (-): Bir sayıdan diğerini çıkarır.
    - Çarpma (*): İki sayıyı çarpar.
    - Bölme (/): Bir sayıyı diğerine böler.
    - Mod alma (%): Bölme işleminin kalanını verir.
    - Üs alma (**): Bir sayının belirtilen kuvvetini hesaplar.
    - Tam sayıya yuvarlama (//): Bölme işleminin sonucunu tam sayıya yuvarlar.
- **Karşılaştırma Operatörleri:**
    - Eşittir (==): İki değerin eşit olup olmadığını kontrol eder.
    - Eşit değildir (!=): İki değerin eşit olmadığını kontrol eder.
    - Büyüktür (>): Bir değerin diğerinden büyük olup olmadığını kontrol eder.
    - Küçüktür (<;): Bir değerin diğerinden [1] küçük olup olmadığını kontrol eder.
    - Büyük eşittir (>=): Bir değerin diğerinden büyük veya eşit olup olmadığını kontrol eder.
    - Küçük eşittir (<=): Bir değerin diğerinden küçük veya eşit olup olmadığını kontrol eder.
- **Mantıksal Operatörler:**
    - Ve (and): İki koşulun da doğru olması durumunda True döner.
    - Veya (or): Koşullardan en az biri doğruysa True döner.
    - Değil (not): Koşulun tersini alır.
- **Atama Operatörleri:**
    - Eşittir (=): Bir değeri bir değişkene atar.
    - Artırma (+=): Bir değişkenin değerini belirli bir miktar artırır.
    - Azaltma (-=): Bir değişkenin değerini belirli bir miktar azaltır.

Farklı Programlama Dillerindeki Operatörler

Her programlama dilinin kendine özgü operatörleri ve kullanımları olabilir. Ancak temel mantık genellikle aynıdır. Örneğin, C, Java, JavaScript gibi birçok dilde yukarıda verilen operatörlerin benzerleri bulunur.

Operatörler, programlama dillerinin temel yapı taşlarından biridir. Bu nedenle, hangi programlama dilini kullanırsanız kullanın, operatörleri iyi anlamak önemlidir.

Programlamada Değişkenler

Değişkenler, bilgisayar programlarında belirli bir değeri tutmak için kullanılan adlandırılmış hafıza konumlarıdır. Bu değerler programın çalışması sırasında değişebilir veya sabit kalabilir. Değişkenler, verileri saklamak ve bu verilere programın farklı yerlerinden erişmek için kullanılır.

Neden Değişkenlere İhtiyaç Duyarız?

- **Veri Saklama:** Programın içindeki verileri (sayılar, metinler, vb.) saklamak için değişkenler kullanılır.
- **Verilere Erişmek:** Programın farklı yerlerinden bu verilere kolayca erişmek için değişken isimleri kullanılır.
- **Hesaplamalar Yapmak:** Değişkenler üzerinde matematiksel işlemler yapılabilir.
- **Karar Verme:** Değişkenlerin değerlerine göre farklı kod bloklarının çalıştırılması sağlanır.

Değişken Tanımlama

Bir değişkeni tanımlamak için genellikle şu yapı kullanılır:

```
veri_tipi değişken_adı = değer;
```

- **veri_tipi:** Değişkenin hangi tür veriyi tutacağını belirtir (örneğin, int, float, string).
- **değişken_adı:** Değişkene verilen isimdir.
- **değer:** Değişkene atanan ilk değerdir.

Örnek (Python):

Python
```
# Tam sayı değişkeni
sayi = 10

# Ondalıklı sayı değişkeni
pi = 3.14

# Metin değişkeni
isim = "Ali"

# Mantıksal değişken
durum = True
```

Örnek C++
```
int yas = 30; // Tam sayı bir değişken tanımladık ve 30 değerini atadık
double pi = 3.14; // Ondalıklı sayı bir değişken tanımladık
char ilkHarf = 'A'; // Karakter bir değişken tanımladık
bool evliMi = false; // Mantıksal bir değişken tanımladık
```

Java'da Değişken Tanımlama

Örnek:

Java
```
int yas = 30; // Tam sayı bir değişken tanımladık ve 30 değerini atadık
double pi = 3.14; // Ondalıklı sayı bir değişken tanımladık
char ilkHarf = 'A'; // Karakter bir değişken tanımladık
boolean evliMi = false; // Mantıksal bir değişken tanımladık
```

PHP değişken tanımlama

$sayi = 10; // Tam sayı bir değişken

$isim = "Ali"; // Metin bir değişken

$pi = 3.14; // Ondalıklı sayı bir değişken

$dogruMu = true; // Mantıksal bir değişken

Değişken Türleri

- **Sayısal Değişkenler:** Tam sayılar (int), ondalıklı sayılar (float), karmaşık sayılar (complex) gibi sayısal değerleri tutar.
- **Metin Değişkenleri:** Metinleri (string) tutar.
- **Mantıksal Değişkenler:** Doğru (True) veya yanlış (False) değerlerini tutar.oluşan bir koleksiyondur.

Değişkenlerin Önemi

Değişkenler, programlamada temel yapı taşlarından biridir. Programın akışını kontrol etmek, hesaplamalar yapmak ve verileri işlemek için kullanılırlar. Değişkenleri doğru bir şekilde kullanmak, daha okunaklı, anlaşılır ve hata ayıklaması kolay programlar yazmanıza yardımcı olur.

Örnek Kullanım Alanları:

- **Hesaplamalar:** Matematiksel işlemler yapmak için değişkenler kullanılır.
- **Veri Saklama:** Kullanıcıdan alınan bilgiler (isim, yaş, vb.) değişkenlerde saklanır.
- **Karar Verme:** Değişkenlerin değerlerine göre farklı kod bloklarının çalıştırılması sağlanır (örneğin, if-else yapısı).
- **Döngüler:** Belirli bir koşul sağlandığı sürece bir kod bloğunun tekrar tekrar çalıştırılması için değişkenler kullanılır (örneğin, for döngüsü).

Örnek 1 : Bir işyerindeki çalışanların bilgisini tutmak için; Yaş , maaş, kullanıcı adı, cinsiyet, telno, kilo değerlerini uygun veri tipiyle ilk değerlerini kendiniz vererek tanımlayınız?

Veri türü değişken adı = değer // Örnek değişken tanımlama şekli

İnteger yas=20 // İnteger yas;

İnteger maaş=55000

String kullanıcı adı=”FSM”

Boolean cinsiyet=bay

String telno=”5058088898”

Float kilo=50.5

Örnek 2: Kütüphane programı Kitap Bilgileri İçin uygun veri türleriyle ilk değerlerini vererek uygun şekilde tanımlayınız

// Kitap adı // Yazar adı // Yayın yılı // ISBN numarası // Mevcut kitap sayısı

String isim=”Kör duman”

String yazar=”Kemal Tahir”

int yayinYili=1995

String ISBN=12356

int adet=100

Örnek 3: Kütüphane programı Üye Bilgileri İçin Değişken tanımlama örneği ilk değerlerini vermeden.

Üye adı//Üye numarası//Üye telefon numarası// Üye eposta

String ad;

int uyeNo;

String telefon;

String eposta;

Byte, Integer, Long, Float, String, Boolean, Char Kavramları

- **-byte**: 0-255 arası küçük sayılar için.
- **-int:** Tamsayılar.
- **-long:** Büyük tamsayılar.
- **-float**: Ondalıklı sayılar.
- **-string:** Metinler.
- **-boolean**: True/False değerleri.
- **-char:** Tek karakterler

Simge	İşlev
	Başla/Bitir
	Giriş
	Atama/İşlem
	Denetim (Karar)
	Çıkış
	Döngü
	Akış Yönü
	Bağlaç
	Önceden Tanımlı İşlem/Fonksiyon

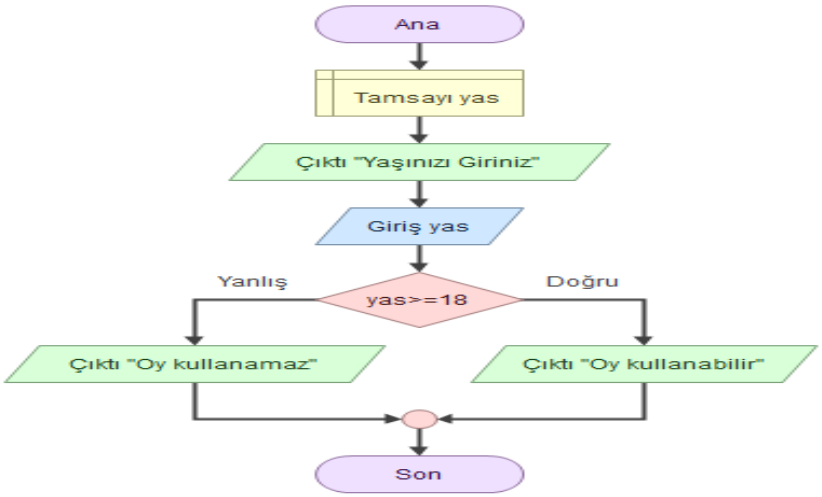
Karar- Döngü Algoritma ve Akış Diyagramları Örnekleri

Örnek1: Kullanıcının yaşına göre oy hakkı olup olmadığını kontrol eden bir program

Algoritma

1. Başla
2. Yaşı gir, yaş
3. Eğer yaş>=18
4. Oy kullanabilir 6. Adıma git
5. Oy kullanamaz
6. bitir

Akış diyagramı

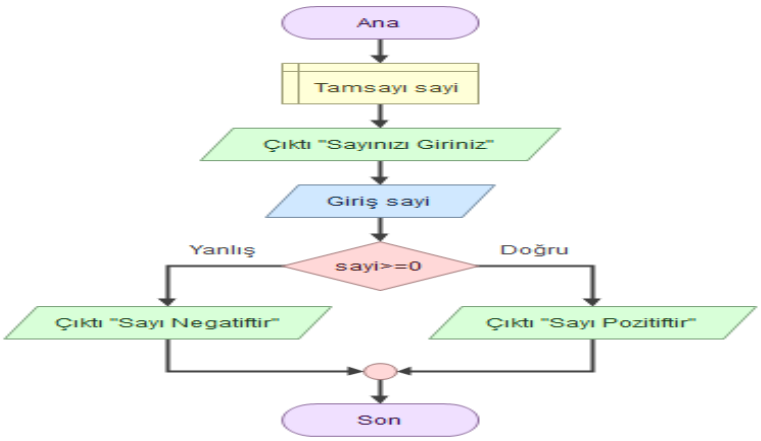


Örnek2: Kullanıcıdan bir sayı alıp pozitif mi, negatif mi olduğunu söyleyen bir programın algoritmasını ve akış diyagramını yazın.

Algoritma

1. Başla
2. Kullanıcıdan sayı al x
3. Eğer x>0 5. Adıma git
4. Sayı negatiftir. 6. Adıma git
   Sayı pozitiftir
6. Bitir.

Akış diyagramı

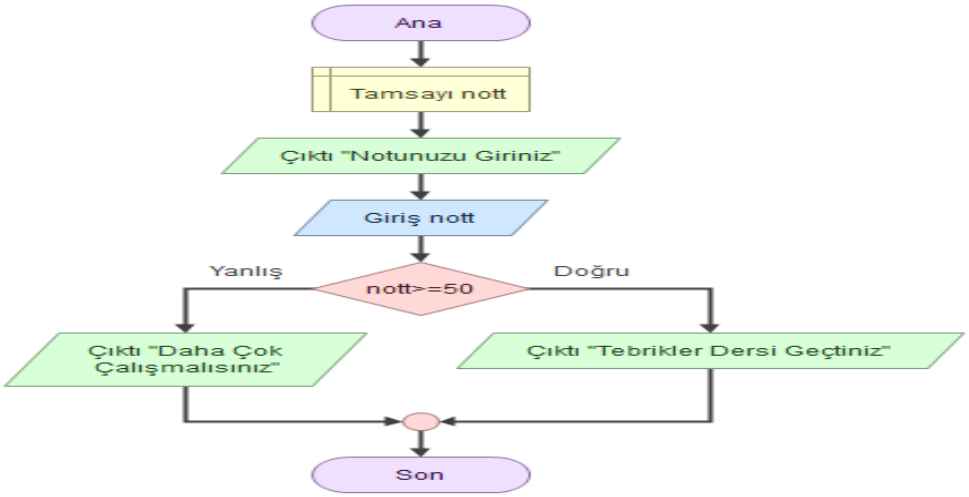


Örnek3: Klavyeden girilen nota göre öğrencinin geçti /kaldı bilgisini veren programın algoritma ve akış diyagramını oluşturunuz?(öğrenci notu 50 ve üstündeyse geçti)

Algoritma

1. Başla
2. Notu gir x
3. X>=50 5. Adıma git
4. Kaldı 6. Adıma git
5. Geçti
6. Bitir

Akış Diyagramı

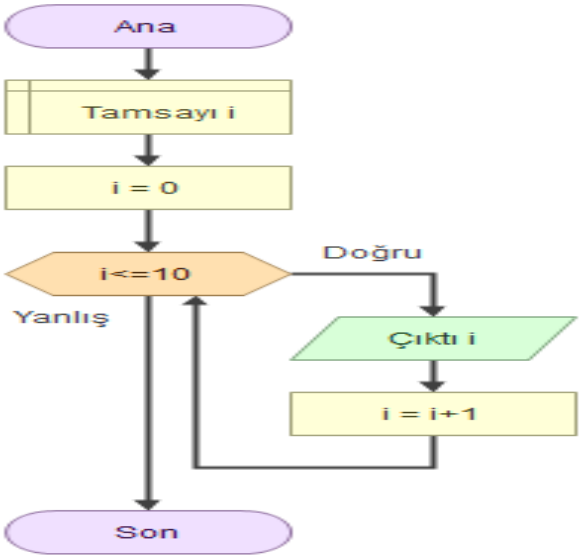


Örnek 4 : 1'den 10'a kadar sayıları yazdırmak için döngü kullanmak

Algoritma

1. **Başla**
2. İ değişkenini 1 olarak başlat (ata)
3. **i <= 10** olduğu sürece:
    a. i'yi ekrana yazdır
    b. i'yi 1 artır
4. **Bitti**

Akış diyagramı (while döngüsü ile)

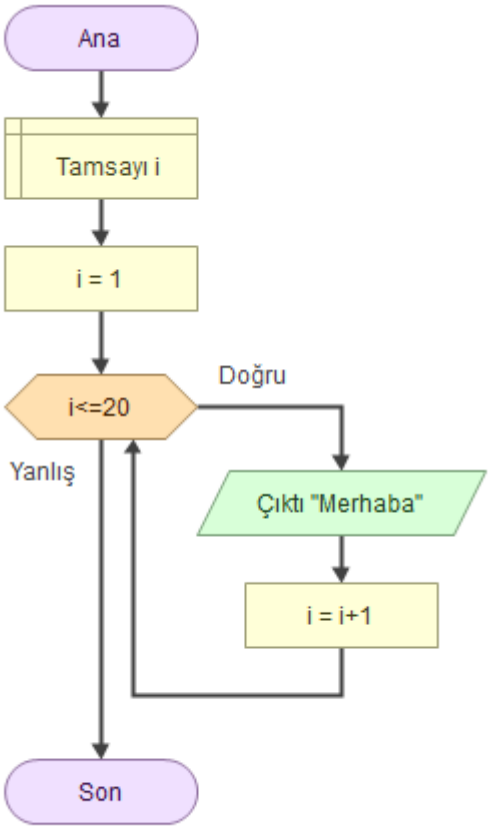


Örnek 5: Ekrana 20 defa “Merhaba” yazdıran programı while döngüsü kullanarak yazdırın.

Algoritma

1. Başla
2. i’yi tanımla
3. i=1 ata
4. i<=20
    a. “Merhaba” yi yaz
    b. i=i+1
5. Bitir.

Akış Diyagramı

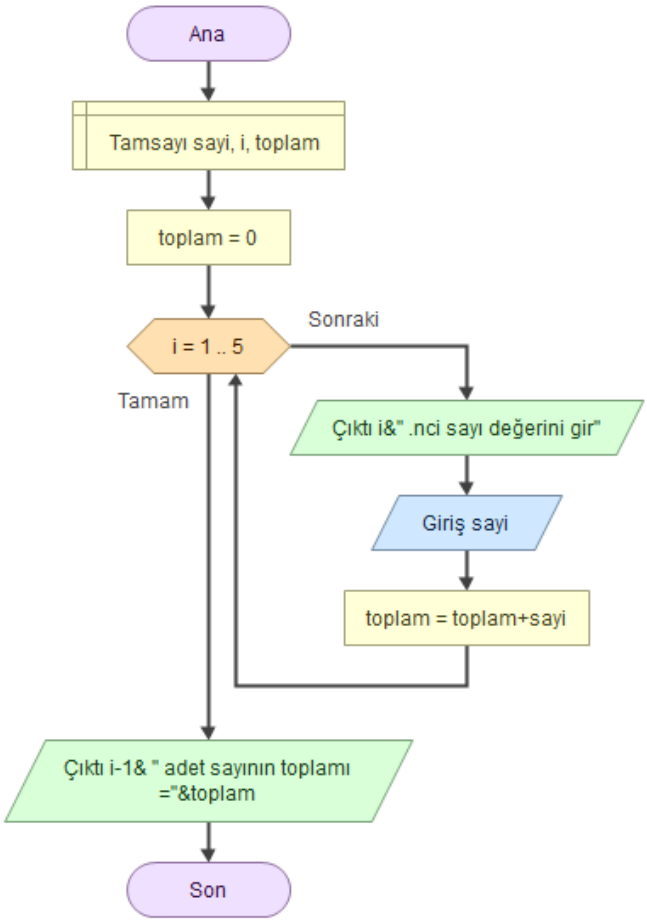


Örnek6: Klavyeden girilen 5 adet sayının toplamını veren programın algoritmasını ve akış diyagramını oluşturunuz. (for döngüsü kullanarak)

Algoritma

1. Başla
2. Toplam=0
3. İ<5 olduğu sürece
    a. Sayi değerini gir , sayi
    b. Toplam=toplam+sayi
4. Toplamı göster, toplam
5. Bitir.

Akış Diyagramı



Örnek7: Kullanıcının klavyeden girdiği sayı sıfırdan büyükse 0 dan girdiği sayıya kadar sayıları 1'er artırarak listeleyen, girdiği sayı 0'dan küçükse "lütfen pozitif sayı giriniz" uyarısı veren programı karar-döngü yapısını kullanarak Algoritmasını ve akış diyagramını yazınız.

Algoritma

1. Başla
2. Sayı tanımla , x,i
3. i=1 ata
4. Sayıyı gir ,x
5. Eğer x>0 7.adıma git
6. i<x
    a. i değerini yazdır
    b. i=i+1
7. "Lütfen pozitif sayı giriniz"
8. Bitir.

Akış Diyagramı