3- KONTROL AKIŞI

İÇERİK

- Kontrol akışı nedir?
- Kontrol akışı türleri nelerdir?
- If ve If-else deyimi
- Karşılaştırma operatörleri
- Switch-case
- Döngüler

KONTROL AKIŞI

- Kontrol yapıları programlama dillerinin en önemli parçasıdır.
- Bir program herhangi bir kontrol yapısı içermiyorsa komutlar bilgisayar tarafından **tekdüze** biçimde art arda çalıştırılır.
- Karmaşık süreçleri modelleyebilmek için;
  - belirli koşullarda programın **yalnızca belirli bir kısmının çalışması** ve diğer kısımların çalışmaması
  - belirli koşullarda programın belirli bir kısmının **tekrar etmesi** gerekebilir.

KONTROL AKIŞI

- Bir bilgisayar programında akış bazı komutlar atlanarak sürdürülebilir.
- Bu atlama işlemine **dallanma** (branching) adı verilir.

KONTROL AKIŞI

- İki tür kontrol yapısı vardır:
  - **Seçme (selection):** Bu yapılarda koşul ya da koşulların gerçekleşmesine bağlı olarak bir komut dizisi yerine başka bir komut dizisi tercih edilmiş olur. If-else, switch-case gibi yapılarla kullanılır.
  - **Tekrarlama/Döngü (loop):** Bu yapılarda koşul ya da koşulların gerçekleşmesine bağlı olarak bir komut dizisi baştan sona tekrar eder. For, while, do-while gibi yapılarla kullanılır.

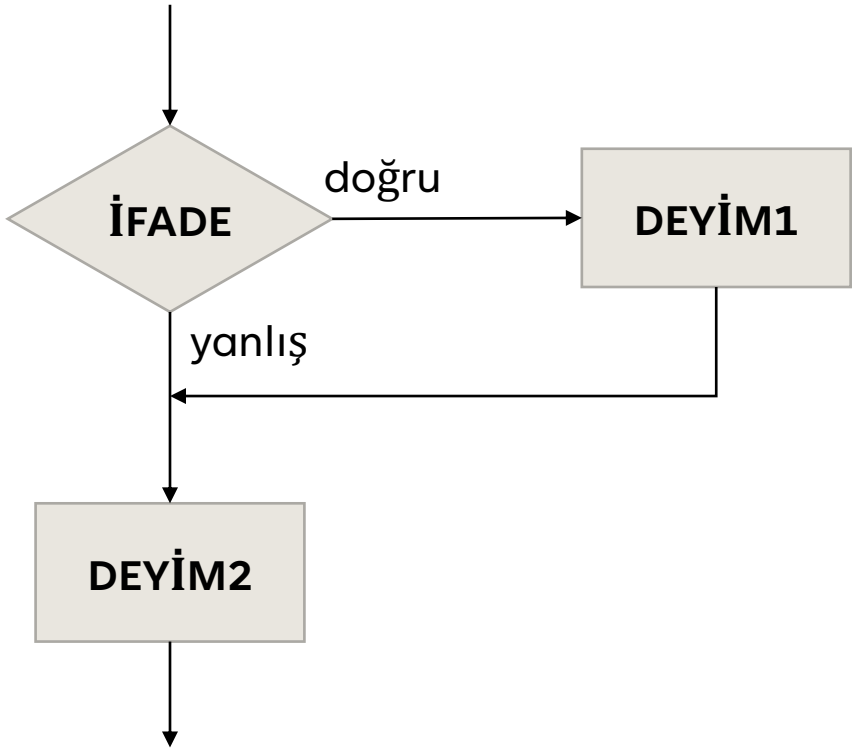
SEÇME (SELECTION)

If Deyimi

- C dilinde şartlı dallanma (conditional branching) adı verilen işlemi gerçekleştiren bir deyimdir.
- Şartlı dallanma sayesinde bir ifadenin sonucuna göre bir komutlar dizisinin icra edilip edilmeyeceğine karar verilir.

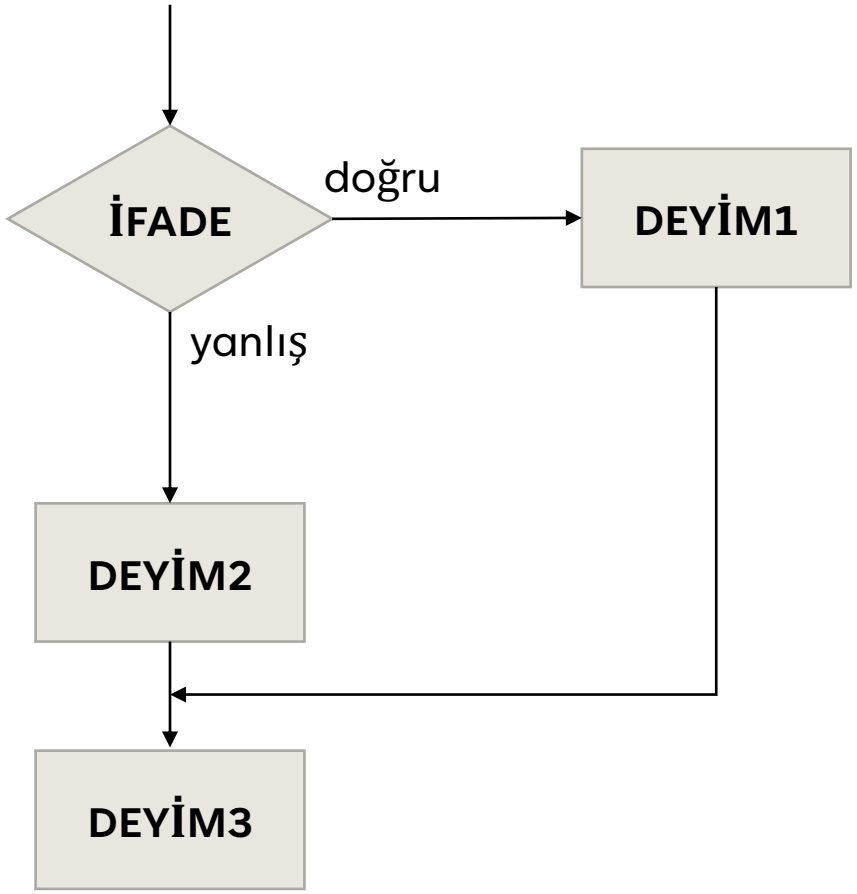
SEÇME (SELECTION)

```
if (ifade) deyim1;
deyim2;
```



SEÇME (SELECTION)

```
if (ifade) deyim1;
else deyim2;
deyim3;
```



SEÇME (SELECTION)

If Deyimi

- Kontrol edilen ifadenin değeri doğru olduğunda Deyim1 gibi tek bir komut değil, bir komutlar dizisinin çalışması isteniyorsa deyimler grubu {} sembolleri arasına alınarak bir blok oluşturulmalıdır.

```
if (ifade) {
        deyim1;
        deyim2;
        ...
        deyimN;
}
```

SEÇME (SELECTION)

```
if (ifade) {
        deyim1;
        deyim2;
        ...
        deyimN;
} else {
        deyim1;
        deyim2;
        ...
        deyimN;
}
```

KARŞILAŞTIRMA OPERATÖRLERİ

Karşılaştırma Operatörü	İşlevi
<	-den daha küçük
>	-den daha büyük
<=	küçük veya eşit
>=	büyük veya eşit
==	-e eşit
!=	eşit değil

İÇ İÇE IF DEYİMLERİ

```
if (ifade1) {
        deyim1;
} else if (ifade2) {
        deyim2;
} else if (ifade3) {
        deyim3;
} else {
        deyim4;
}
```

SWITCH CASE YAPISI

```
switch (degisken) {
        case deger1:deyim1;break;
        case deger2:deyim2;break;
        ...
        case degerN:deyimN;break;
        default:deyim;
}
```

DÖNGÜ

- Çevrimsel veya tekrarlı işlemler yapmak için 3 farklı döngü kullanılabilir:
  - while döngüsü
  - do-while döngüsü
  - for döngüsü

WHILE DÖNGÜSÜ

```
while (ifade) deyim1;
```

veya

```
while (ifade) {
    deyim1;
    deyim2;
    ...
    deyimN;
}
```

DO-WHILE DÖNGÜSÜ

```
do {
    deyim1;
    deyim2;
    ...
    deyimN;
} while (ifade);
```

FOR DÖNGÜSÜ

```
for (atama;koşul;adımlama) {
    deyim1;
    deyim2;
    ...
    deyimN;
}
```

atama ifadesinde, döngü değişkenine ilk değer ataması yapılır.

koşul ifadesinde, döngünün çalışma şartı kontrol edilir.

adımlama ifadesinde, koşul “doğru” ise döngü değişkeninin her çevrimde nasıl değişeceği (artma, azalma) belirlenir.

FOR DÖNGÜSÜ

```
for (i = 0;i <= n; i = i + 1) {
    deyim1;
    deyim2;
    ...
    deyimN;
}
```

ÖDEV

Aşağıda verilen deyimleri araştırınız. Bu ifadeler hangi kontrol deyimleriyle birlikte kullanılır? Ne işe yararlar?

- break deyimi
- continue deyimi
- goto deyimi