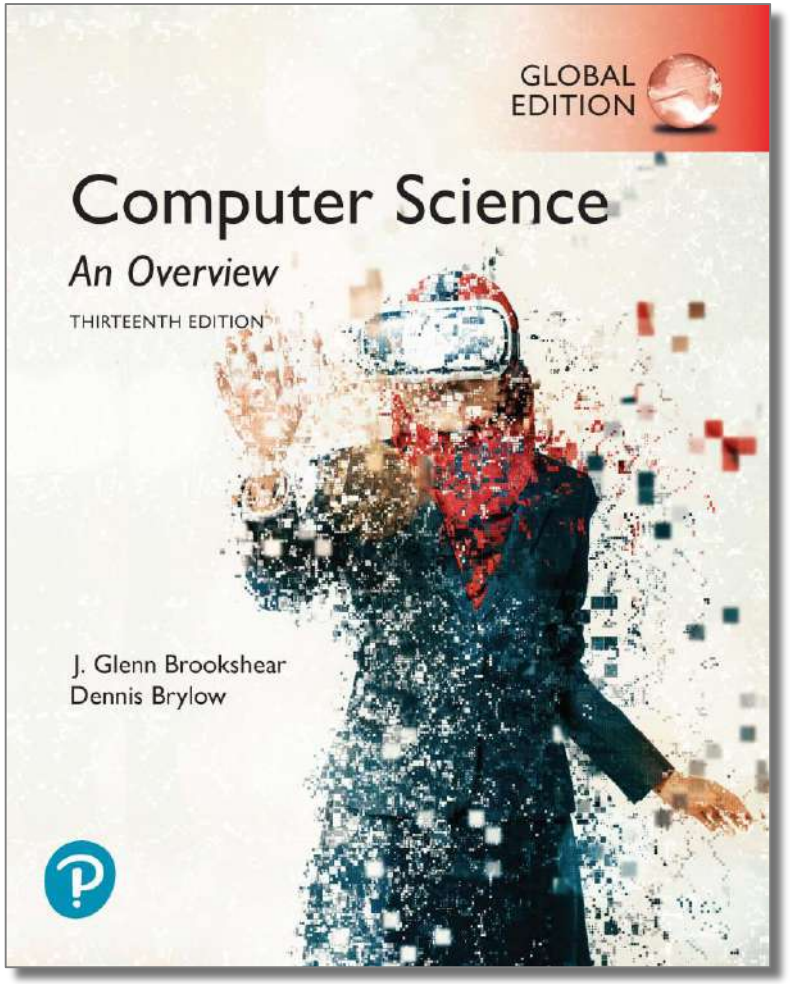
Bilgisayar Bilimine Giriş

13. Baskı, Global Edition



Bölüm 6

Programlama Dilleri

6.1 Tarihi Açıdan Bakış

- Birinci Nesil
  - Makine Dili
  - Assembly Dili

İkinci Nesil: Assembly Dili

- Makine komutlarının gösteriminin anımsatıcı bir temsili
  - Op-kodlar için anımsatıcı isimler
  - Program **değişkenleri** yada **tanımlayıcıları**: Programlayıcı tarafından seçilen bellek yerleri için açıklayıcı isimler

Assembly Dili Karakteristik Özellikleri

- Makine komutları ve assembly komutları arasında birebir haberleşme
  - Programlayıcı makine gibi düşünmeli
- Doğal olarak makineye bağımlıdır
- **Assembler** adlı bir programla makine diline çevrilir.

Program Örneği

Makine dili

```
156C
166D
5056
30CE
C000
```

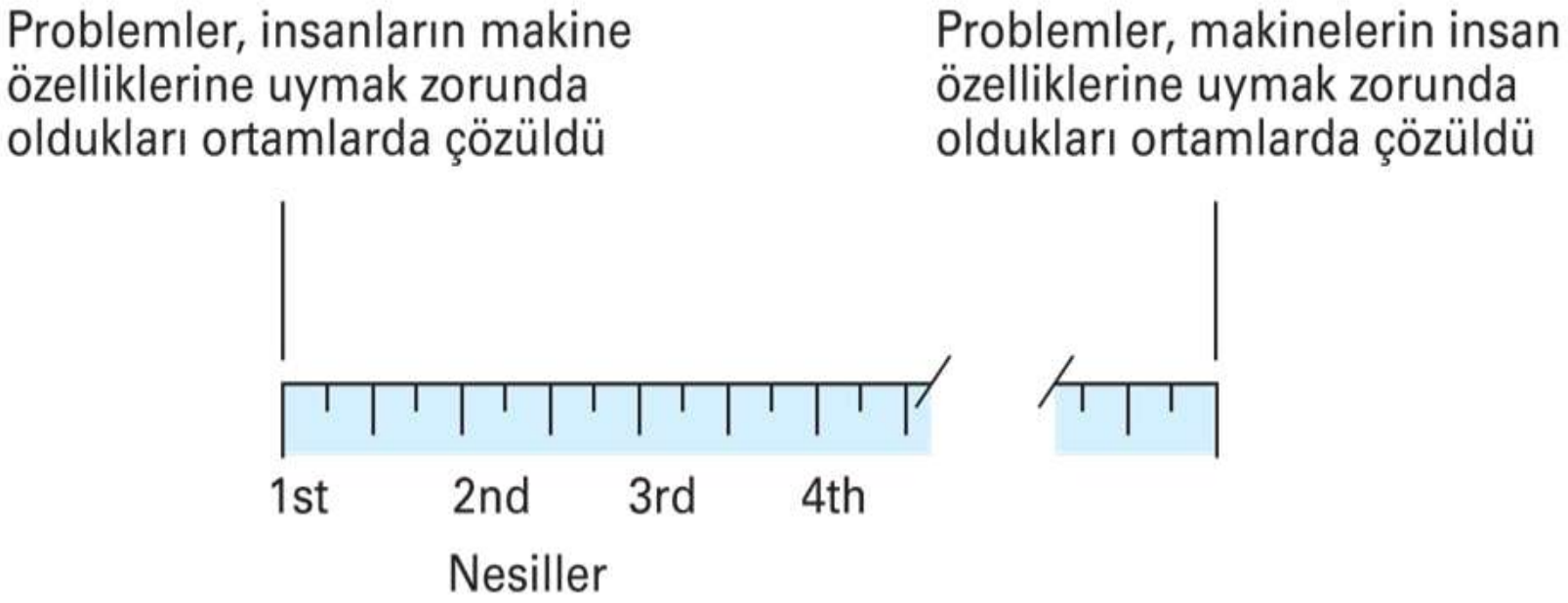
Assembly dili

```
LD    R5, Fiyat
LD    R6, KargoFiyatı
ADD R0, R5 R6
ST    R0, ToplamMaliyet
HLT
```

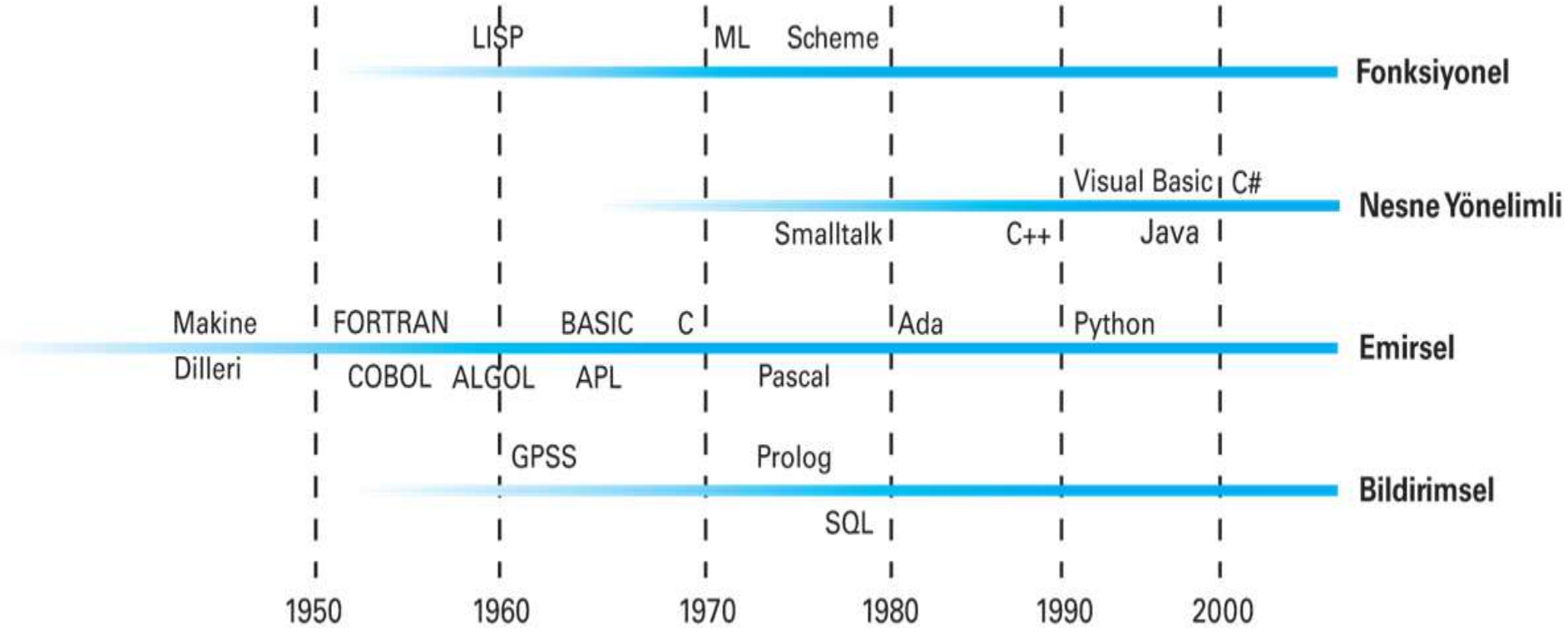
Üçüncü Nesil Diller

- Üst seviye primitifler kullanırlar
  - Bölüm 5'teki psudocode'larımıza benzerler
- Makineden bağımsız (çoğunlukla)
- Örnekler: FORTRAN(FORmula TRANslater), COBOL(COmmon Businness-Oriented Language)
- Her primitif, makine dilinde komutlar dizisine denk gelir
- Makine diline **compiler** (derleyici) adı verilen bir programla çevirilir

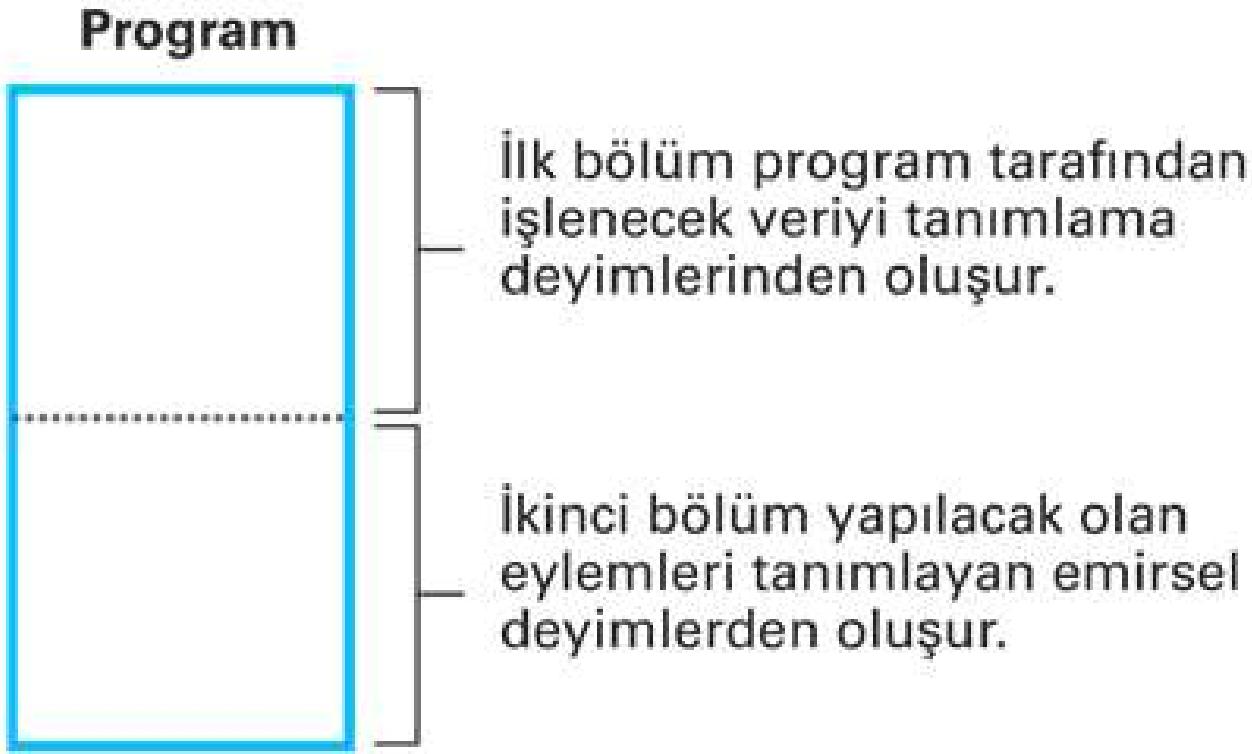
Şekil 6.1 Programlama dillerinin nesilleri



Şekil 6.2 Programlama Paradigmalarının Evrimi



Şekil 6.4 Tipik bir emirsel program ya da program biriminin yapısı



Veri Tipleri

- Integer: Tamsayı
- Real (Gerçek): Kesirli sayılar
- Character(Karakter): Semboller ve harfler
- Boolean: Doğru/Yanlış

Değişkenler ve Veri Tipleri

```
float Uzunluk, Genişlik;
int   Fiyat, Toplam, Vergi;
char  Harf;

int   AğırlıkSınırı = 100;
```

Veri Yapısı

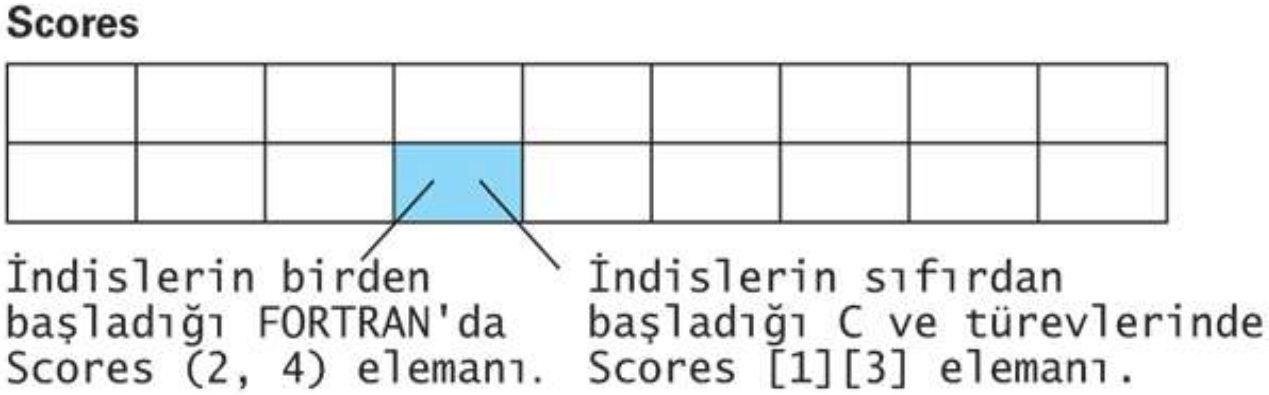
- Verinin kavramsal şekli ya da biçimi
- **Array (Dizi)** adlı yaygın bir veri yapısı
  - C dilinde

```
int Skorlar[2][9];
```

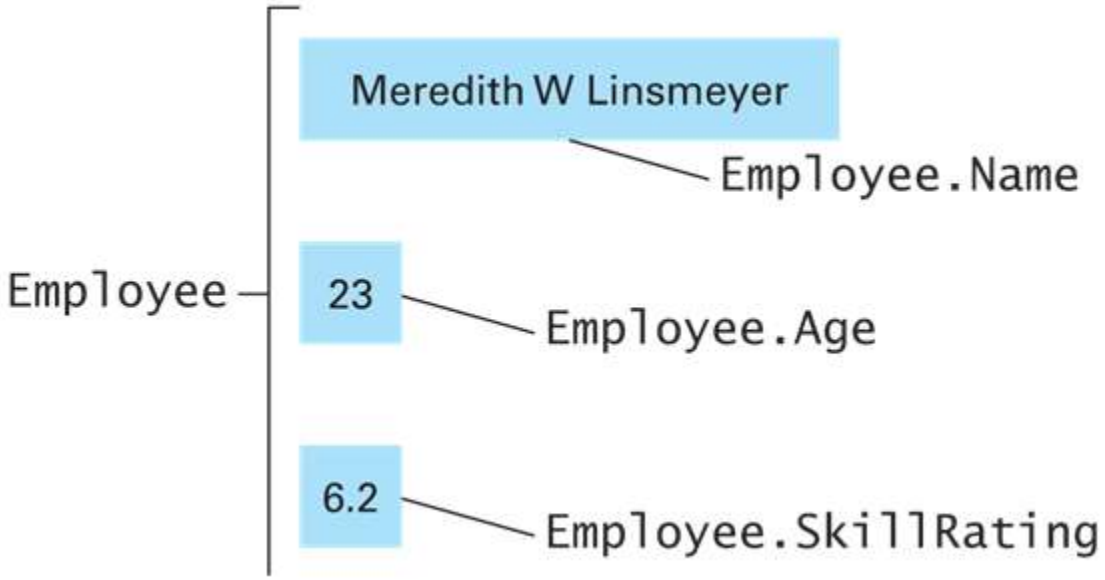
  - FORTRAN dilinde

```
INTEGER Skorlar(2,9)
```

Şekil 6.5 Dokuz sütun ve iki satırdan oluşan iki boyutlu bir dizi



Şekil 6.6 Employee yapısının kavramsal görünümü




```
struct {    char  Isim[25];
            int   Yas;
            float YetenekPuanı;
          } Employee;
```

Atama Deyimleri

- C, C++, C#, Java dillerinde

```
Z = X + y;
```

- Ada dilinde

```
Z := X + y;
```

- APL (A Programming Language)'de

```
Z ← X + y
```

Kontrol Deyimleri

- Bir **kontrol deyimi** programın çalışma gidişatını değiştirir. Bütün kontrol deyimleri içinde en dikkat çekici olanı basit *goto* deyimidir.

```
        goto 40
   20   Evade()
        goto 70
   40   if (KryptoniteLevel < LethalDose) then goto
60
        goto 20
   60   RescueDamsel()
   70   ...
```

Kontrol Deyimleri(devamı)

- Python'da

```
if (koşul):
    durumA
else:
    durumB
```

- C, C++, C#, ve Java dillerinde

```
if (koşul) durumA; else durumB;
```

- Ada dilinde

```
IF koşul THEN
    durumA;
ELSE
    durumB;
END IF;
```

Kontrol Deyimleri(devamı)

- Python dilinde While

```
while (koşul):
    gövde
```

- C, C++, C#, ve Java dillerinde

```
while (koşul)
{ gövde }
```

- Ada dilinde

```
WHILE koşul LOOP
    gövde
END LOOP;
```

Kontrol Deyimleri(devamı)

- C, C++, C#, ve Java dillerinde Switch

```
switch (değişken) {
    case 'A': durumA; break;
    case 'B': durumB; break;
    case 'C': durumC; break;
    default: durumD; }
```

- Ada dilinde

```
CASE değişken IS
    WHEN 'A'=> durum A;
    WHEN 'B'=> durum B;
    WHEN 'C'=> durum C;
    WHEN OTHERS=> durum D;
END CASE;
```

Açıklama/Yorum satırları

- Programın içinde açıklama amaçlı kullanılır
- Programı yazan insanın dışında birinin programı okurken anlaması için yazılırlar
- Derleyici(compiler) tarafından görmezden gelinirler okunmazlar

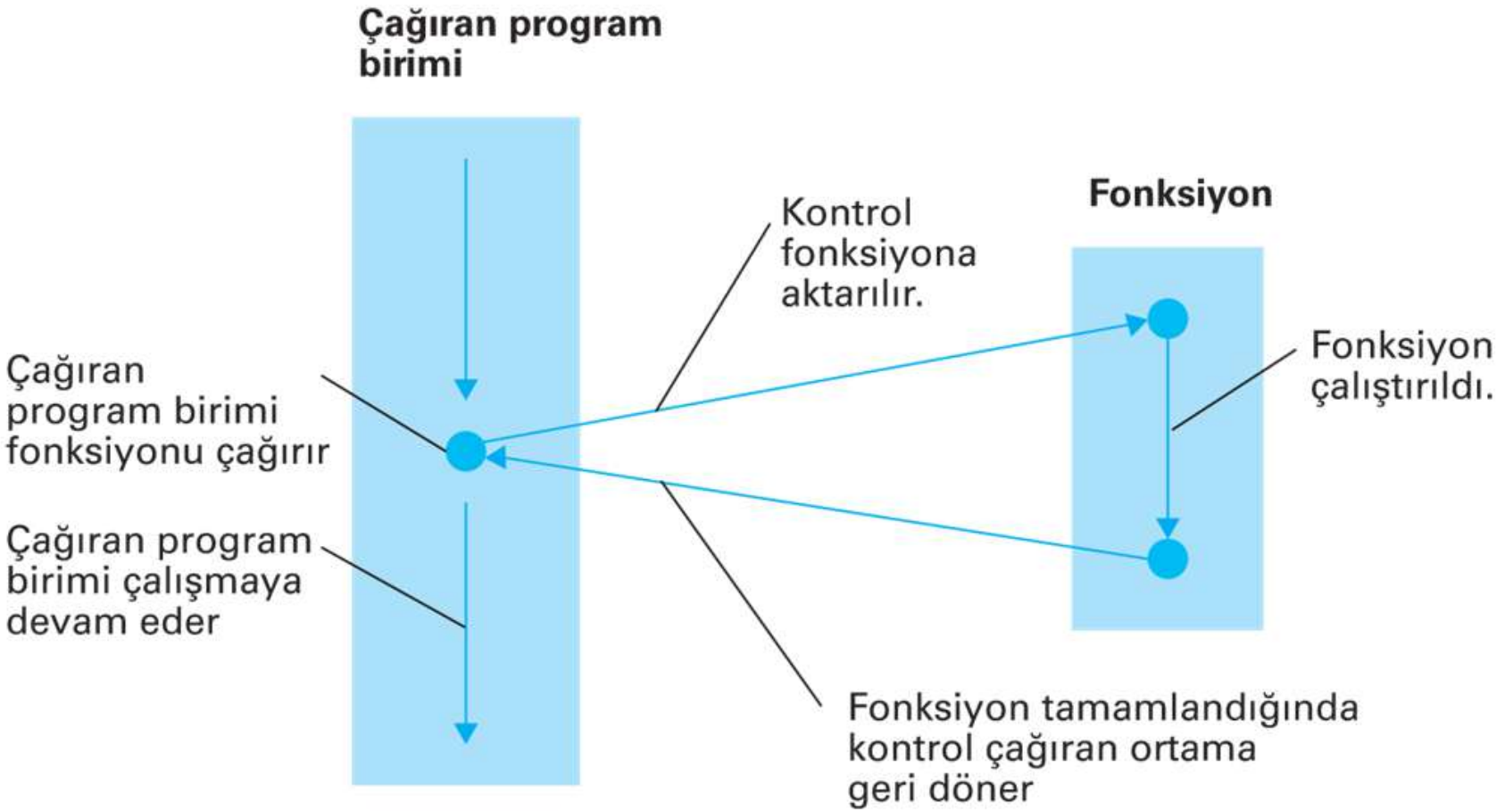
```
/* Bu C/C++/Java dillerinde bir yorum satırıdır.
*/
```

```
// Bu C/C++/Java bir yorum satırıdır.
```

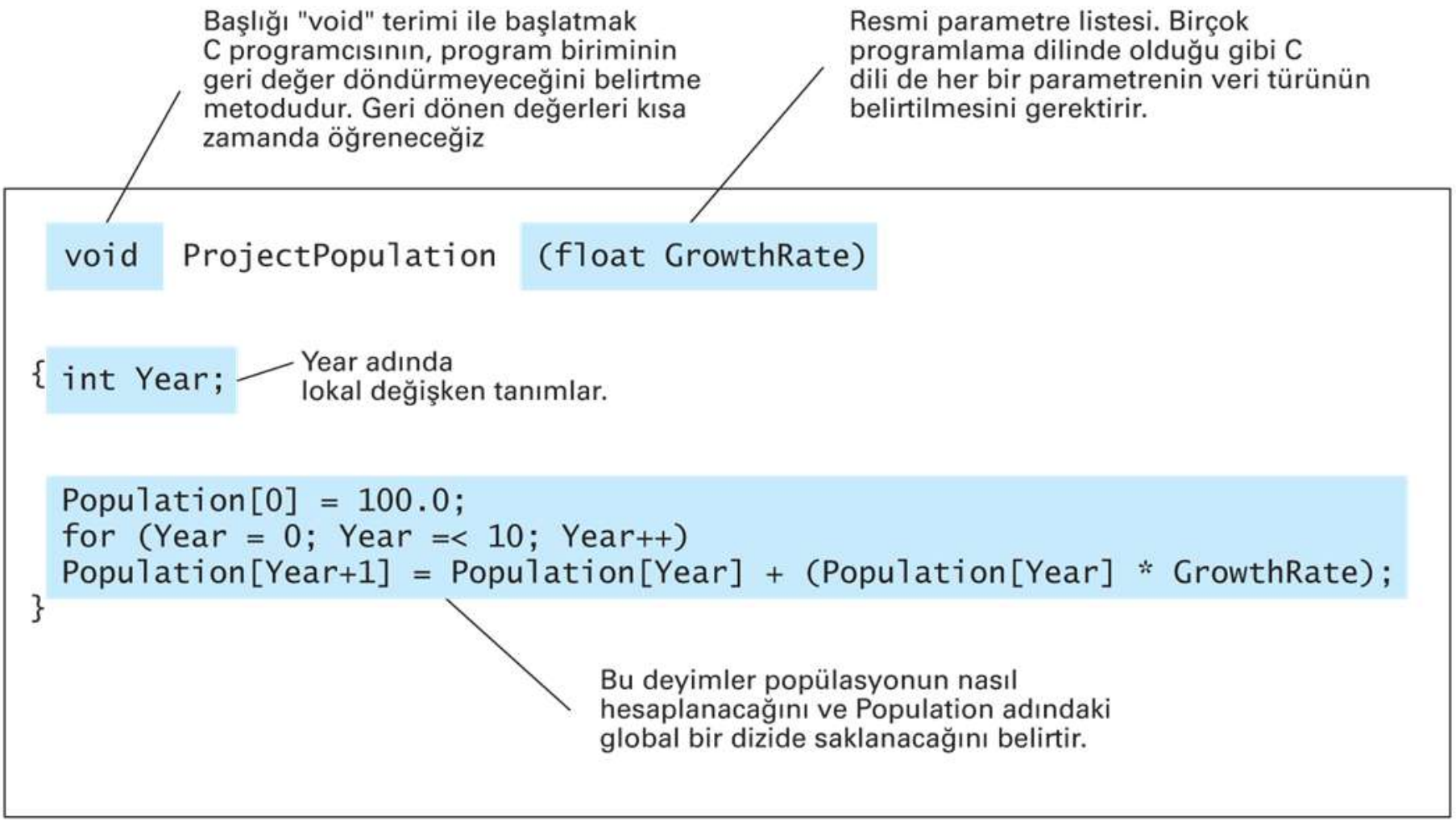
6.3 Yordamsal Birimler

- Bu kavram için bir çok terim vardır:
  - Altprogram,
  - Altrutin,
  - prosedür,
  - metot,
  - fonksiyon

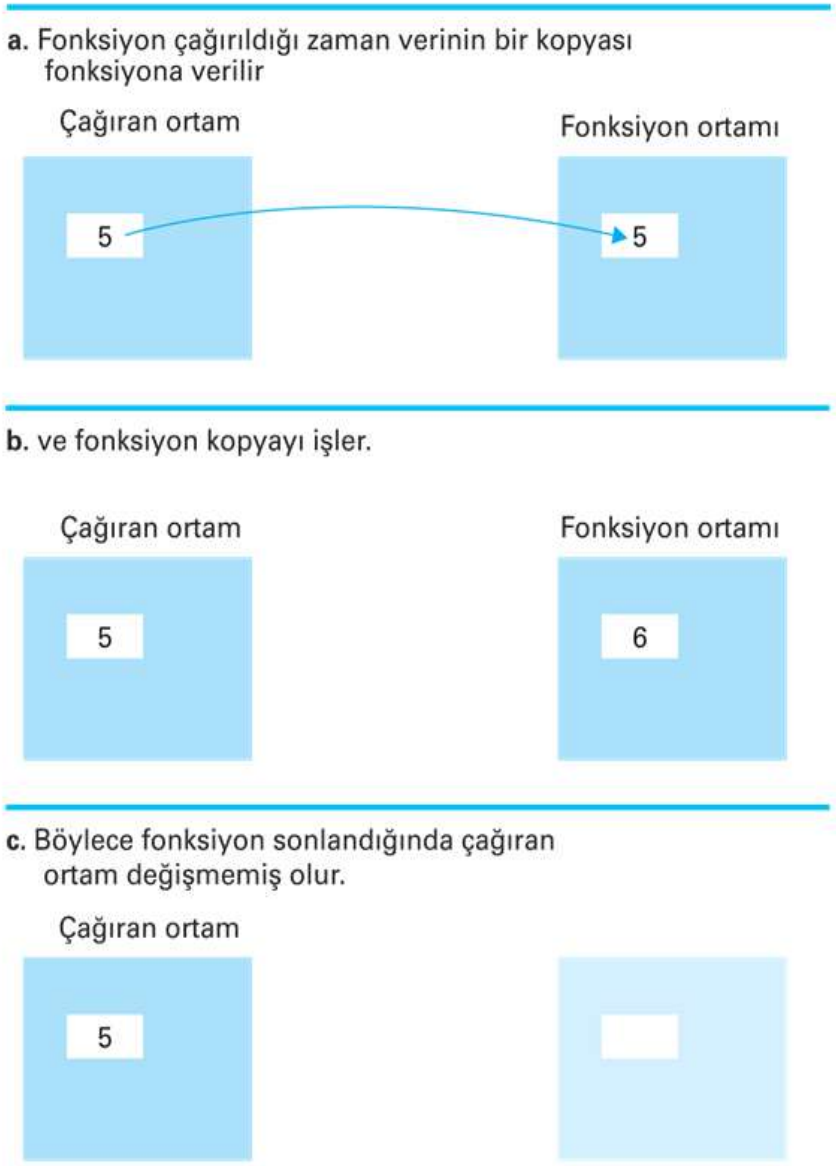
Şekil 6.8 Bir fonksiyon içeren kontrol akışı



Şekil 6.9 C dilinde yazılmış ProjectPopulation fonksiyonu

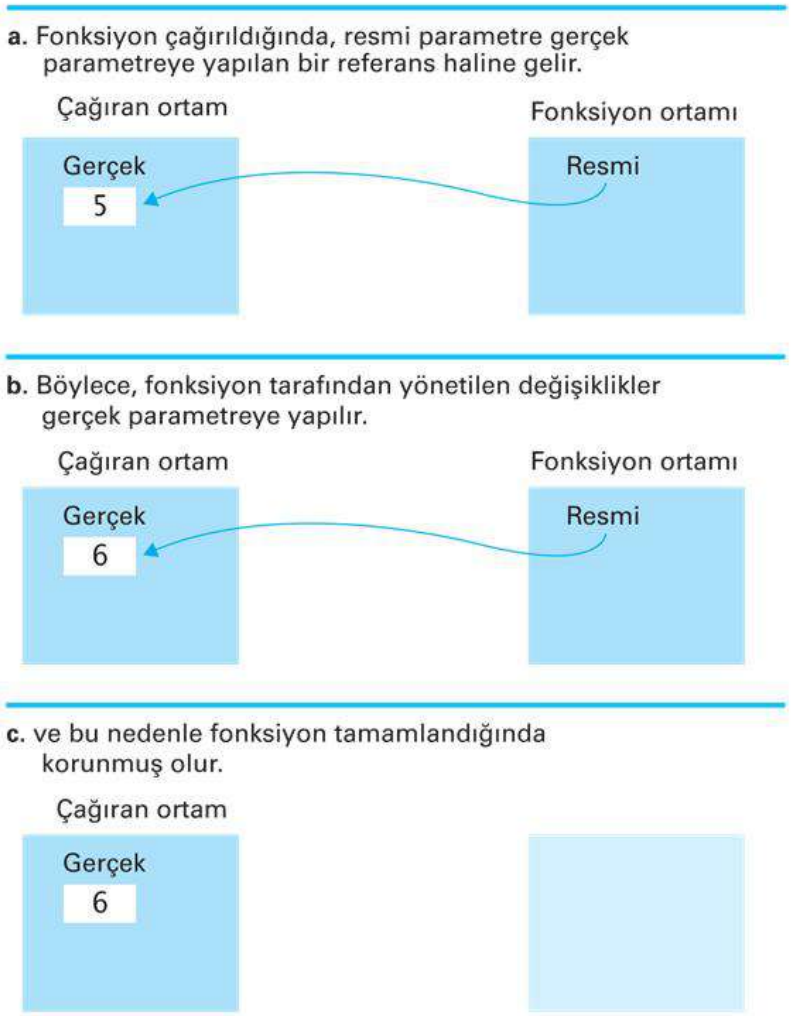


Şekil 6.10 Demo fonksiyonunun çalıştırılması ve parametreleri değer ile aktarma



Şekil 6.11 Demo fonksiyonunun çalıştırılması ve referans ile parametre aktarma

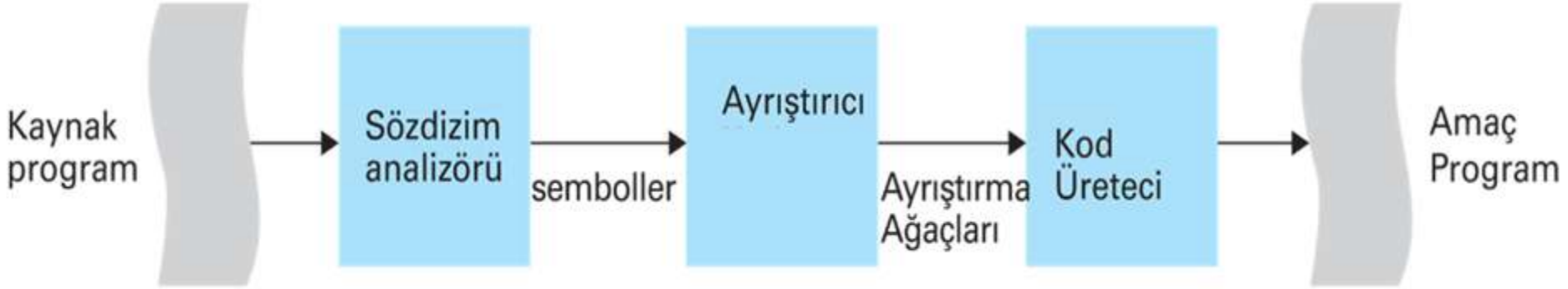
C örneği:
http://tpcg.io/ILIoSy7M



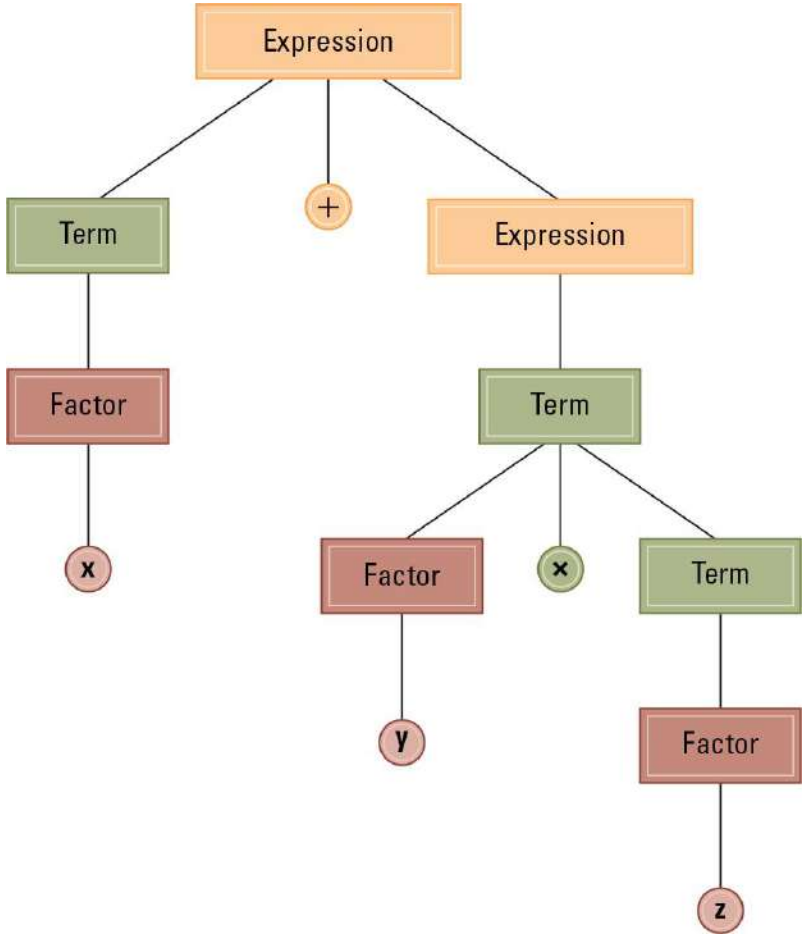
6.4 Programlama Dili Tasarlanması

- Yüksek seviye bir dille yazılmış bir programın makine tarafından çalıştırılabilir forma çevrilmesi işidir.
  - Sözdizim analizörü, tek bir işaret ya da varlıkla gösterilen sembol dizisini anımsar.
  - Ayrıştırıcı, işaretleri bir araya getirip durumlara dönüştürür.Bunu ayrıştırma ağaçları üretmek için sözdizimi diyagramlarıyla yapar.
  - Kod üreteci, durumların uygulanması için makine dili komutları üretir.

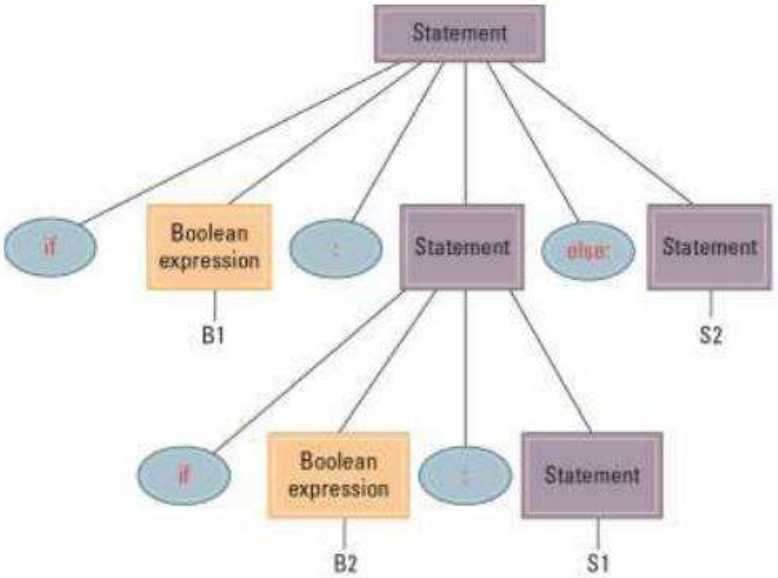
Şekil 6.13 Çevrim süreci

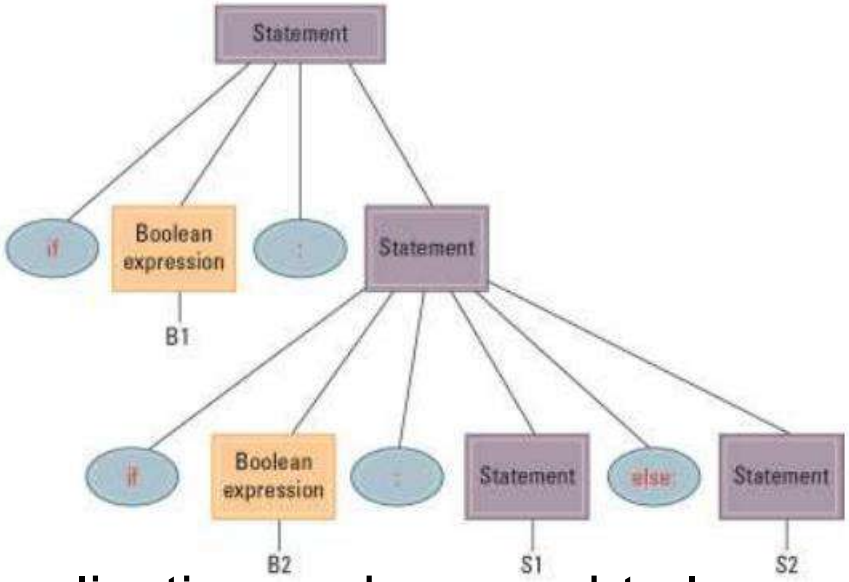


Şekil 6.16 Şekil 6.17'teki sözdizimi şemalarını temel alan x+y*z ifadesi için bir ayrıştırma ağacı



Şekil 6.17 Basit cebirsel bir ifadeyi tanımlayan sözdizimi şeması





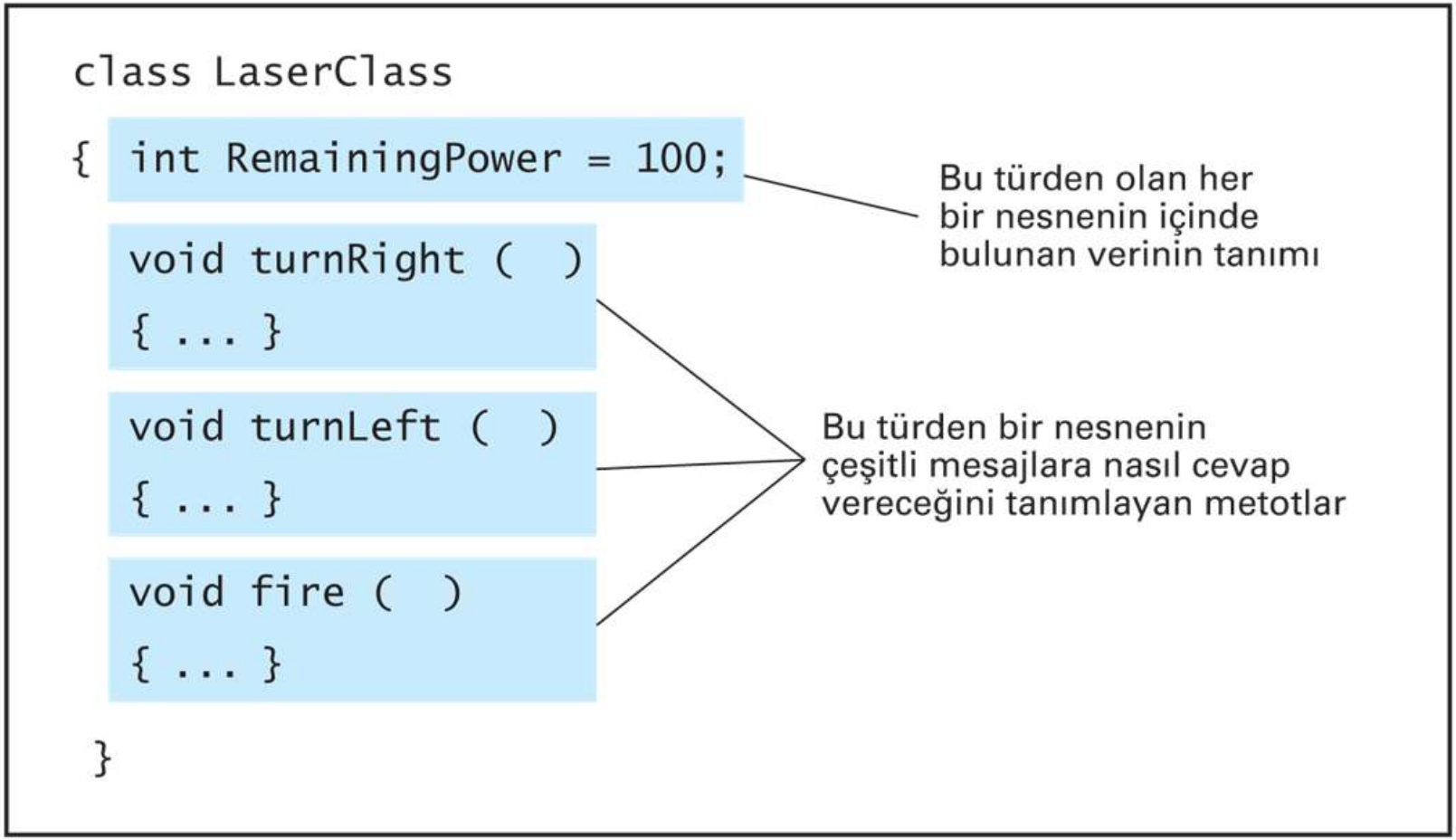
Örnek:
http://nhiro.org/learn_language/AST-Visualization-on-browser.html

6.5 Nesneye yönelik programlama

- **Nesne:** Hem veri hem prosedürleri içeren aktif program birimidir
- **Sınıf:** Nesne topluluğundan oluşan bir şablondur

Nesneye bir sınıfın **örneği** denir.

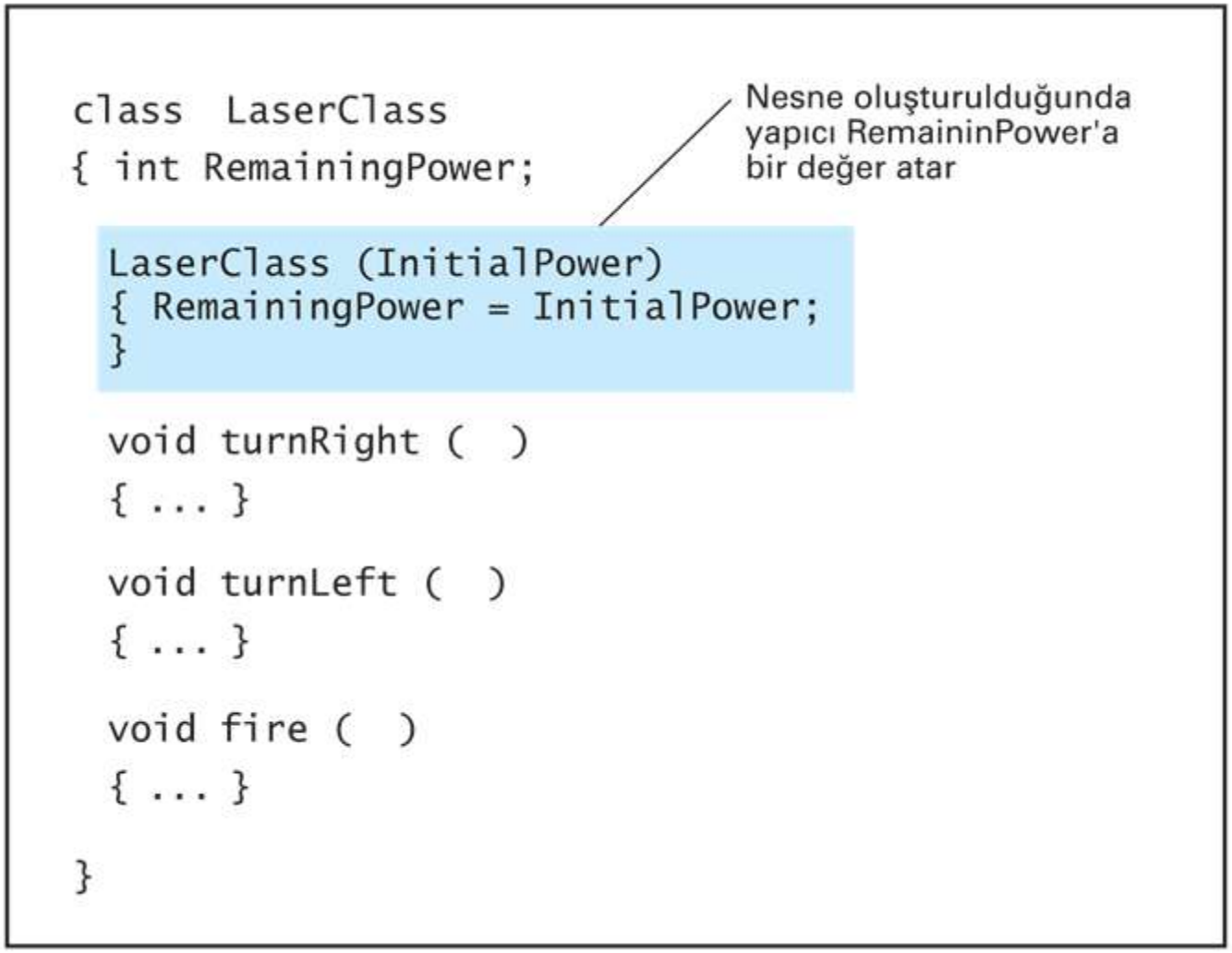
Şekil 6.19 Bir bilgisayar oyunundaki lazer silahını tanımlayan bir sınıfın yapısı



Nesnenin Bileşenleri

- **Değişkenler:** Bir nesnenin içindeki değişken
  - Nesnenin içindeki bilgiyi tutar
- **Metotlar:** Nesnenin prosedürü
  - Nesnenin yapabileceği şeyleri açıklar
- **Yapıcılar:** Yeni bir nesneye ilk inşa edildiğinde başlamak için kullanılan özel metottur

Şekil 6.21 Yapıcı fonksiyonlu bir sınıf

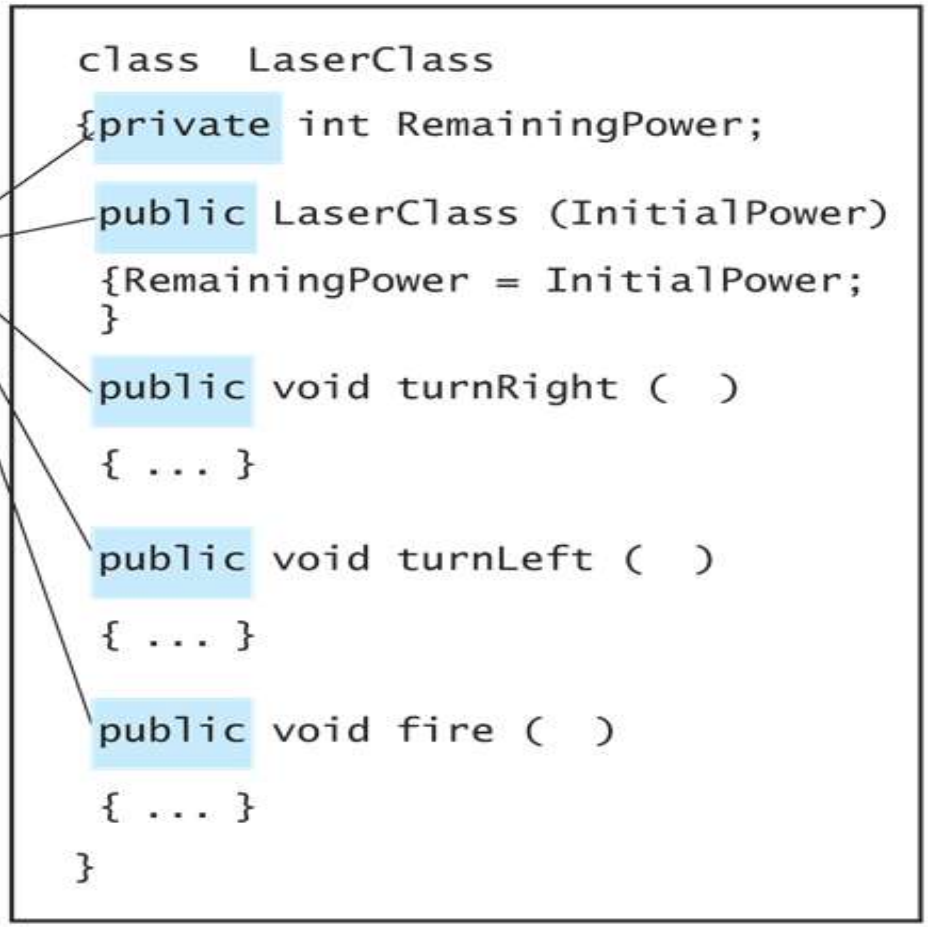


Nesne Bütünlüğü

- **Kapsülleme:** Nesnenin iç bileşenlerine erişimi kontrol etmenin bir yolu
  - Public
  - Private

Şekil 6.22 Bir Java ya da C# programında olduğu gibi kapsülleme kullanan LaserClass tanımlaması

Sınıftaki bileşenler diğer program birimlerinden erişilebilir olup olmamalarına göre public ya da private olarak tasarlanmıştır.



```
class  LaserClass
{private int RemainingPower;

 public LaserClass (InitialPower)
 {RemainingPower = InitialPower;
 }
 public void turnRight (  )
 { ... }

 public void turnLeft (  )
 { ... }

 public void fire (  )
 { ... }
}
```

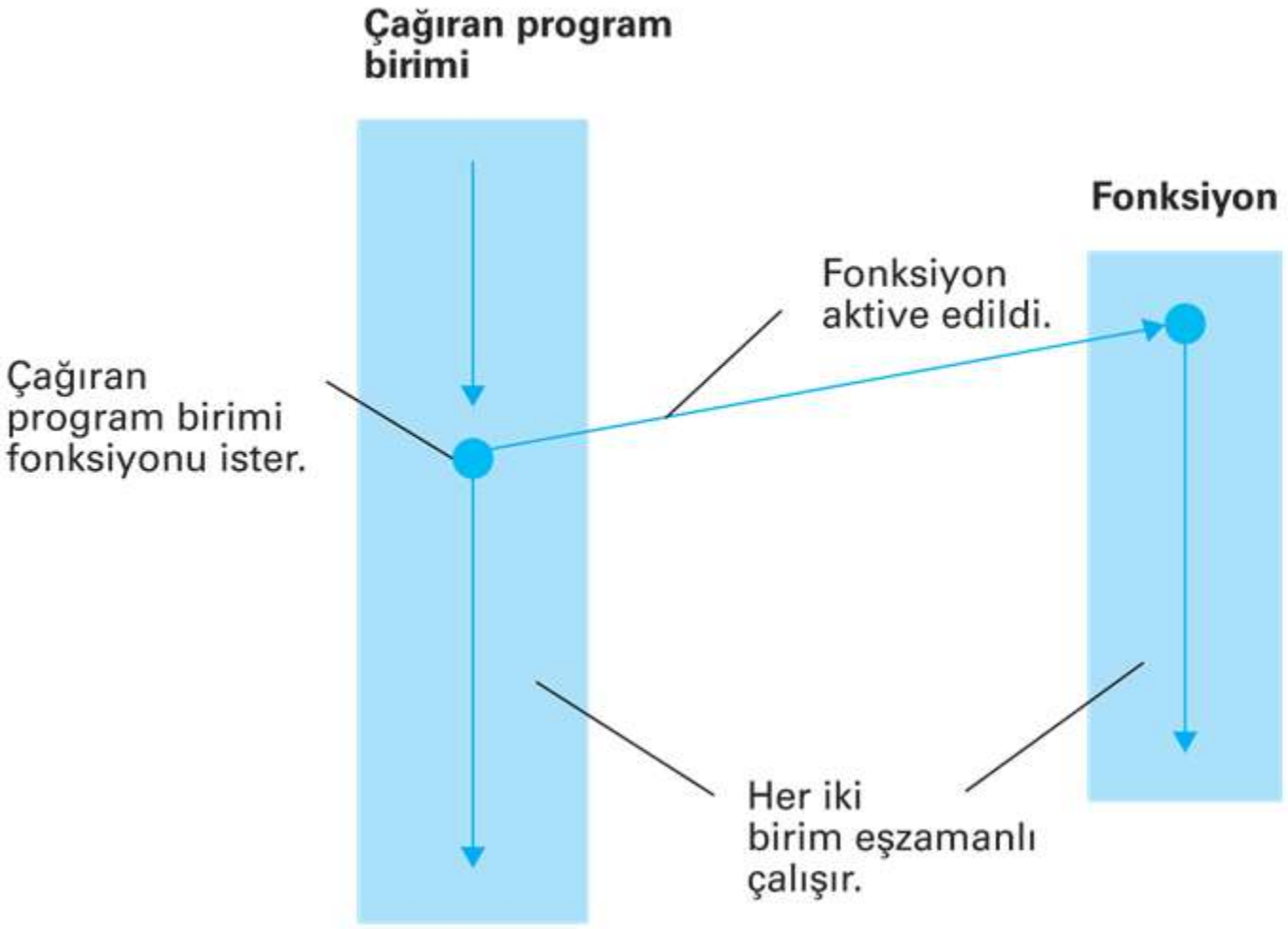
Diğer Nesneye Yönelik Programlama Kavramları

- **Miras Alma (Inheritance):** Önceden tanımlanmış sınıfların terimlerini yeni sınıflarda kullanmaya olanak sağlar
- **Çok Biçimlilik (Polymorphism):**
  - Bir türün bir başka tür gibi davranabilme ve bu tür gibi kullanılabilme özelliğidir.
  - Nesne yönelimli programlama dillerinde çok biçimlilik özelliği ise; Nesnenin davranışı çalışma anında belirlendiği için programcılar, çok biçimlilik özelliği sayesinde nesnelerin türünü önceden bilmek zorunda kalmaz.

6.6 Eş zamanlı eylemleri programlama

- **Paralel işlem:** birden fazla işlemin aynı anda çalıştırılması
  - Paralel işlem için çoklu CPU gerekir
  - Zaman paylaşımlı CPU'lar yardımıyla simüle edilebilir

Şekil 6.23 İş parçacığı (thread) oluşturulması

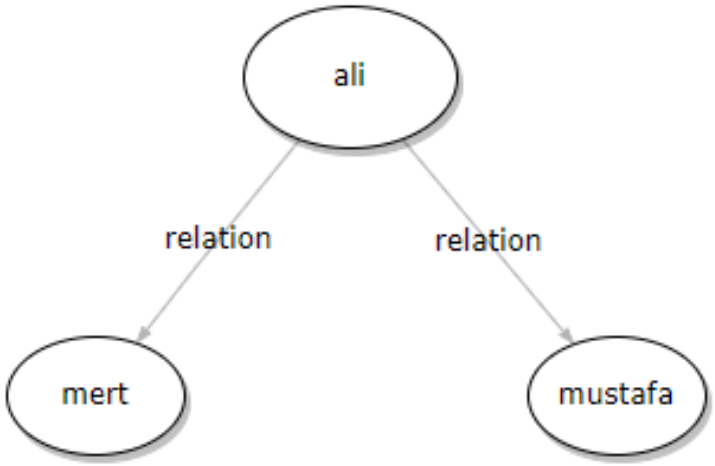


6.7 Bildirimsel Programlama

- **Çözüm:** İki veya daha fazla durumu bir araya getirerek tek bir durum ortaya çıkarmak (bu orjinalinin mantıksal bir sonucudur).
  - Örnek: (P OR Q) AND (R OR ¬Q)
    (P OR R) sonucuna tekabül eder

- Bildirimsel Programlama örneği: PROLOG

Prolog Örneği




```
ogul(mert).
ogul(mustafa).
baba(ali).
evlat(ali,mert).
evlat(ali,mustafa).
kardes(X,Y):-evlat(A,X),evlat(A,Y).
```