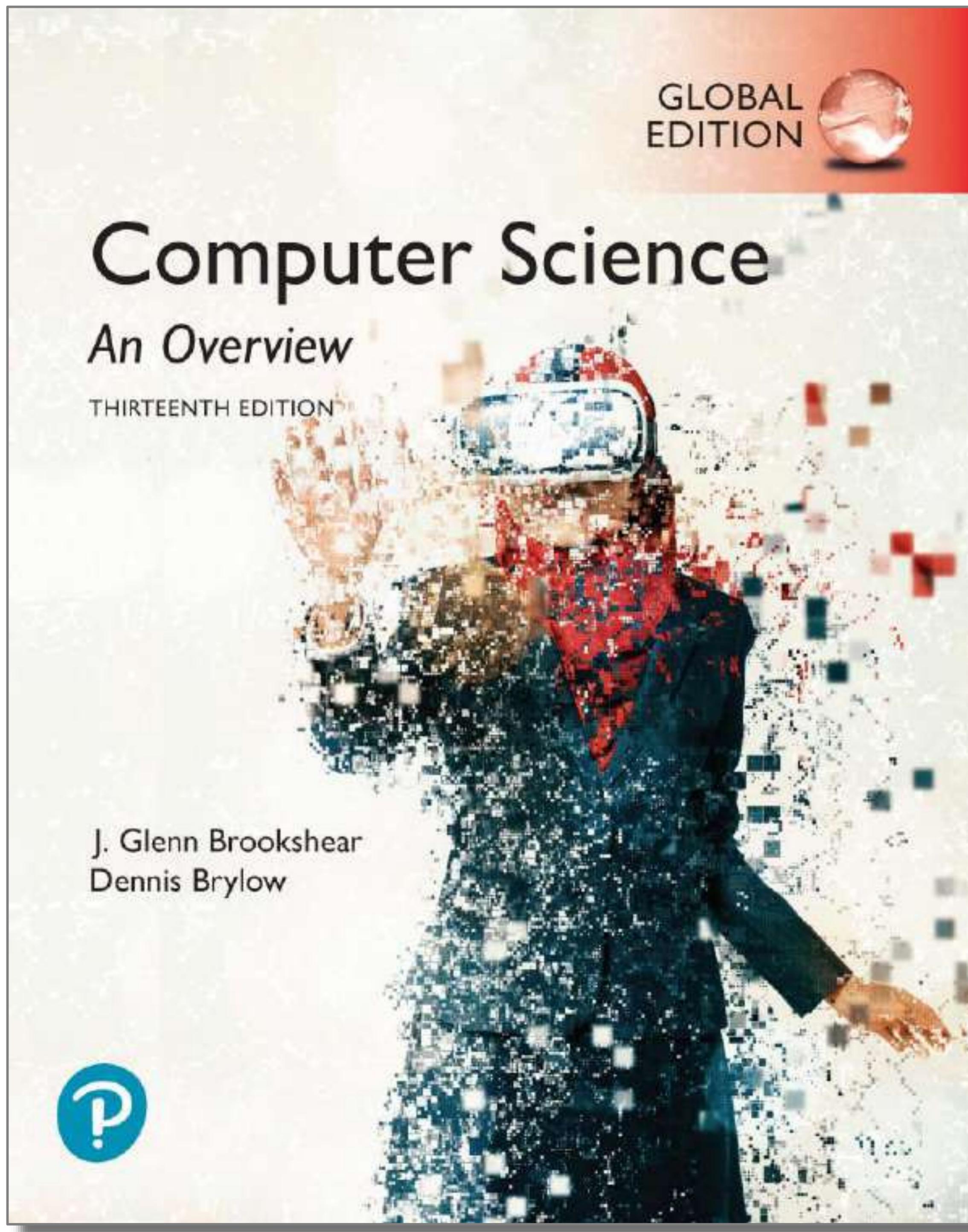


# Bilgisayar Bilimine Giriş

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 bağlıdır
- **Assembler** adlı bir programla makine diline çevrilir.

# Program Örneği

Makine dili

156C  
166D  
5056  
30CE  
C000

flex olmak yarlımız  
iye o'lu

Assembly dili

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

int : → bilgi· bellek  
bölgesine  
isim takteli.

exe → derlenmiş bir  
exe c ye getirilebilir  
ama Assembly dili ile dönüştürülebilir bilg. crane ile aynı crane  
yapılabilir

Makine dili  
komutları C  
de ki İmzalar

Linux  
nesela  
assembly  
komutları  
icinde  
kullanılır

ARM  
intel  
de ASSAHL  
ARM da  
nata  
verir

teçhizat

kod değişimi  
 -64R  
 ↑

~~python py tem - cce  
exe ye komut~~  

---

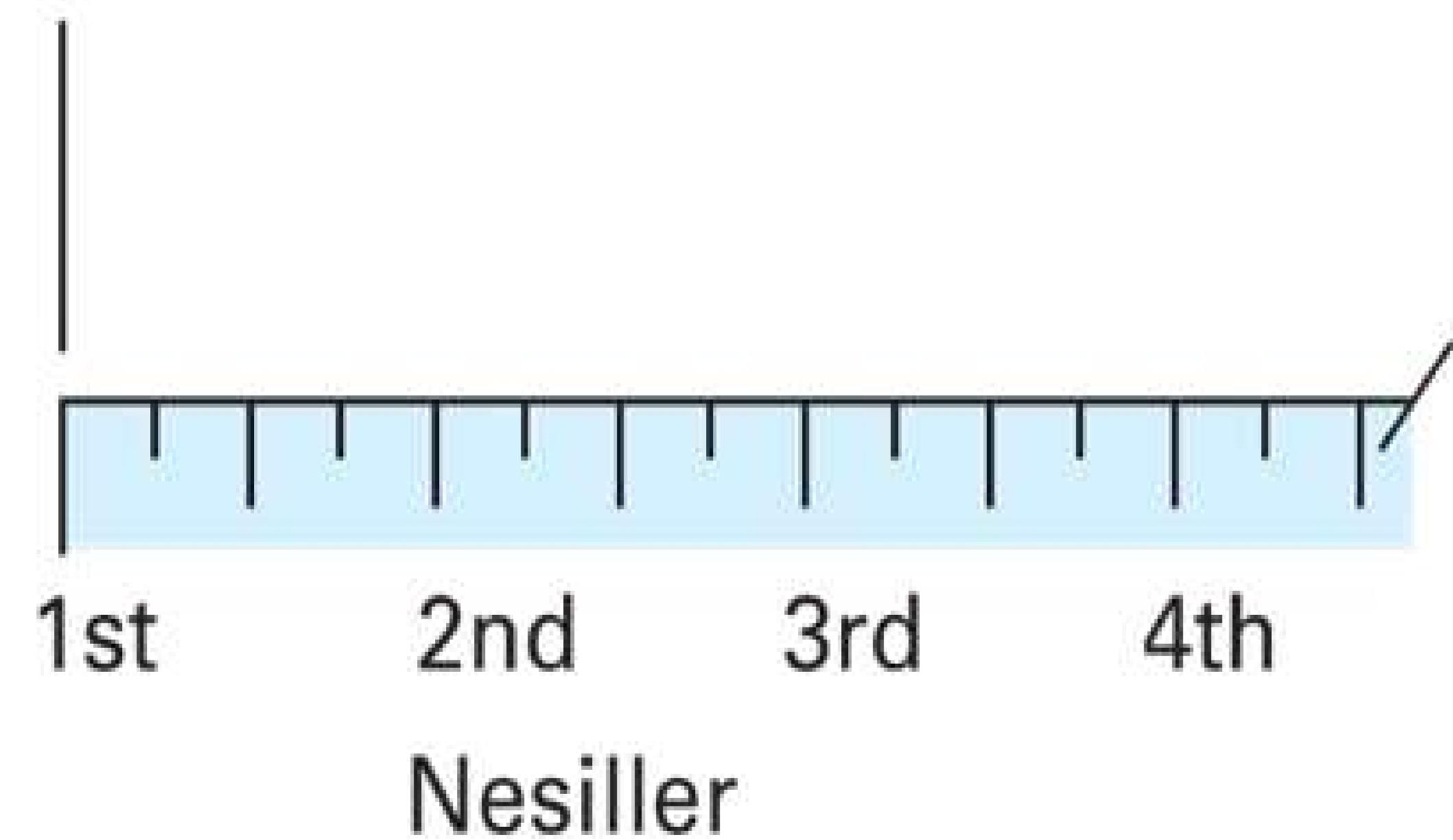
~~cde uretten exe  
zaten makine diline dili~~

# Üçüncü Nesil Diller

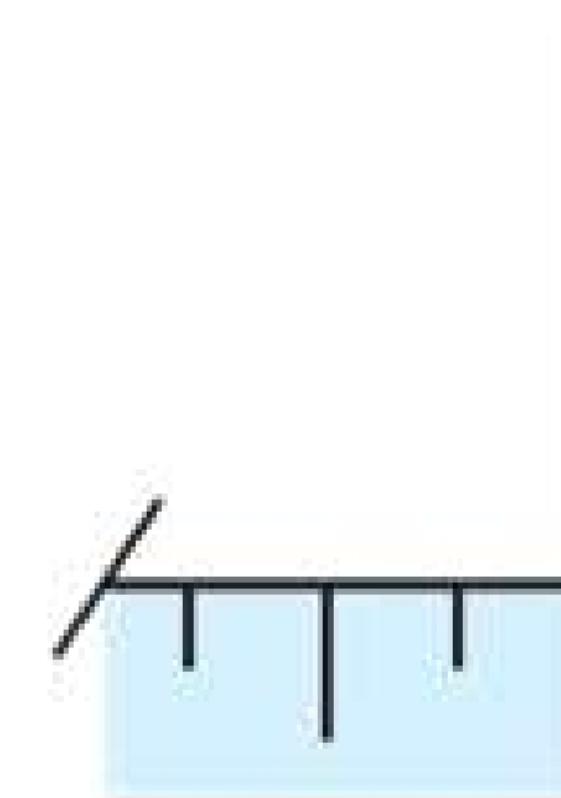
- Üst seviye primitifler kullanırlar
  - Bölüm 5'teki psudocode'larımıza benzerler
- Makineden bağımsız (çoğunlukla)
- Örnekler: FORTRAN(FORmula TRANslater), COBOL(COmmon Business-Oriented Language)
- Her primitif, makine dilinde komutlar dizisine denk gelir
- Makine diline **compiler** (derleyici) adı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

# Şekil 6.1 Programlama dillerinin nesilleri

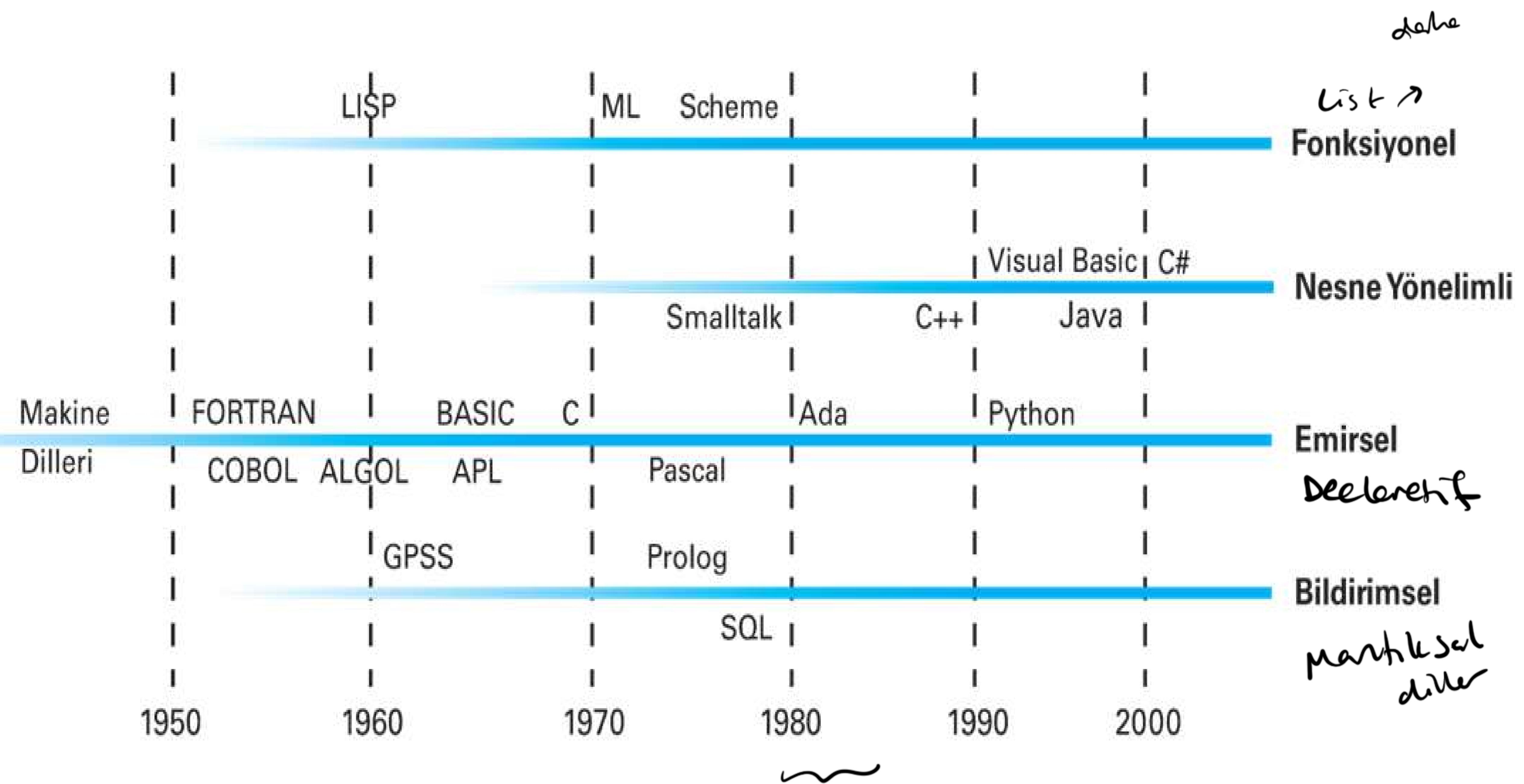
Problemler, insanların makine özelliklerine uymak zorunda oldukları ortamlarda çözüldü



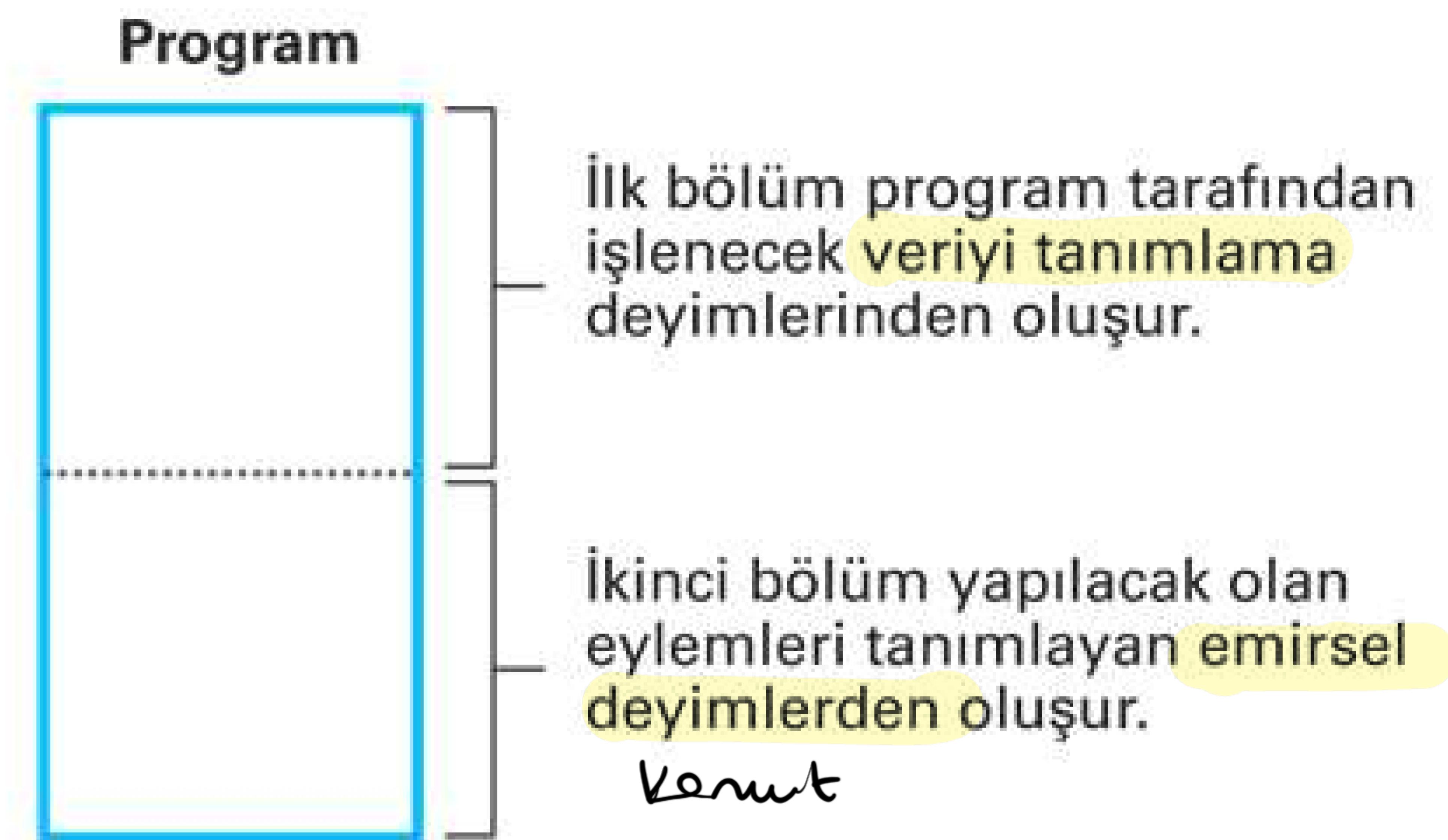
Problemler, makinelerin insan özelliklerine uymak zorunda oldukları ortamlarda çözüldü



# Şekil 6.2 Programlama Paradigmalarının Evrimi



# Şekil 6.4 Tipik bir emirsel program ya da program biriminin yapısı



# Veri Tipleri

- Integer: Tamsayı (her programda döner verdir)
- 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 biçimini
- **Array (Dizi)** adlı yaygın bir veri yapısı

– C dilinde

`int Skorlar[2][9];` → dünün içindeki  
eleman diziidir.

– FORTRAN dilinde

`INTEGER Skorlar(2,9)`

her dilde  
dizi yapısı  
muhlaea bulur.

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

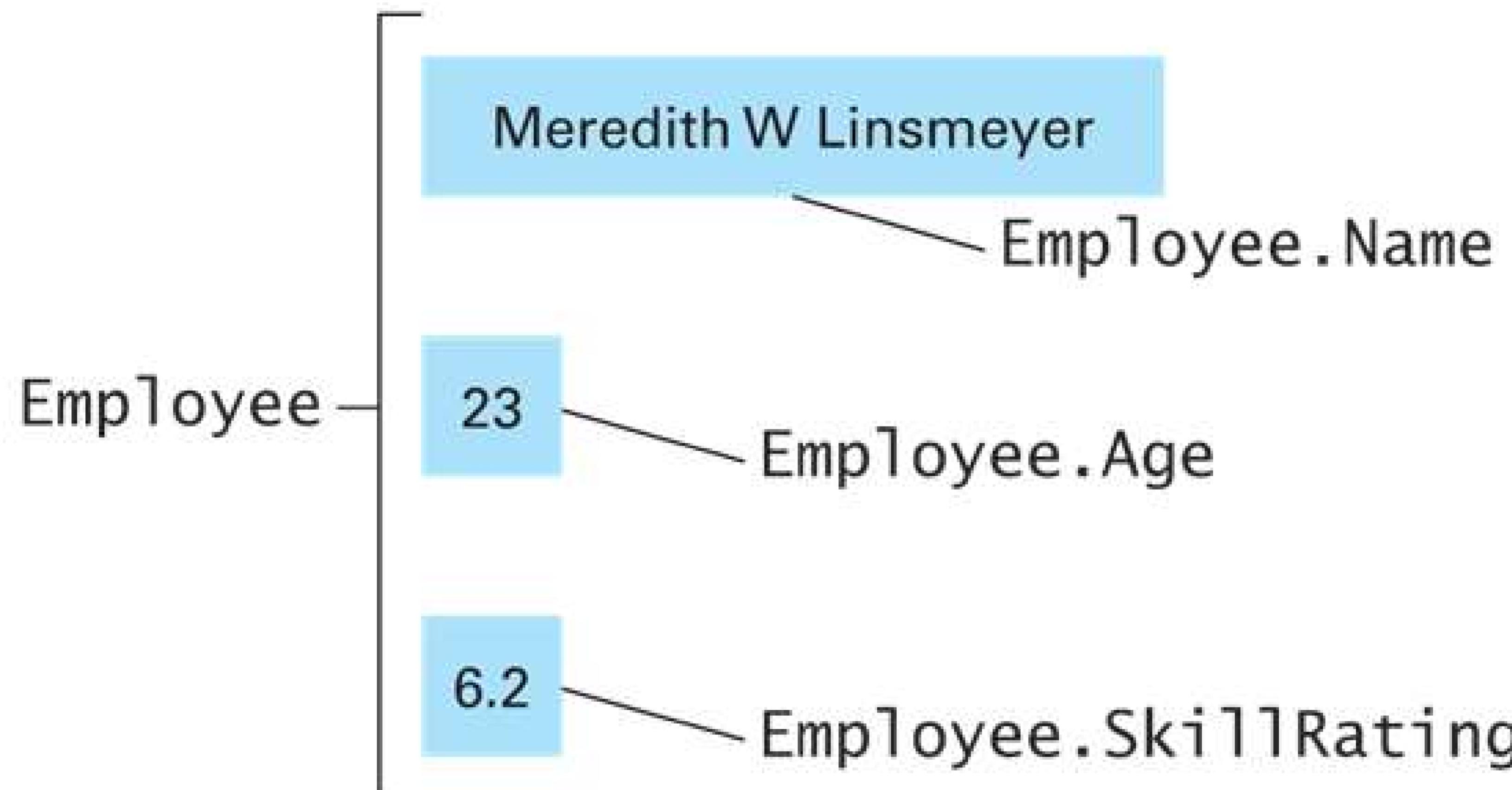
drinde C dairesi 0 den başlıyor

**Scores**


İndislerin birden başladığı FORTRAN'da Scores (2, 4) elemanı.

İndislerin sıfırdan başladığı C ve türevlerinde Scores [1][3] elemanı.

## Şekil 6.6 Employee yapısının kavramsal görünümü



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

complex  
veri tipleri

# Atama Deyimleri

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

$\text{Z} = \text{X} + \text{y};$

- Ada dilinde

$\text{Z} := \text{X} + \text{y};$

- APL (A Programming Language)'de

$\text{Z} \leftarrow \text{X} + \text{y}$

# Program akışının başka bir yere yönlendirilmesi

## Kontrol Deyimleri

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

```
      goto 40          Günümüzde Fort. program  
20   Evade()         if else  
      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

fazla  
den yorum  
satırı koymak programın  
calışma hizini yavaşlatmaz

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

kod  
yorum  
dosyası  
oluştur  
şopay  
relake

```
/* 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:

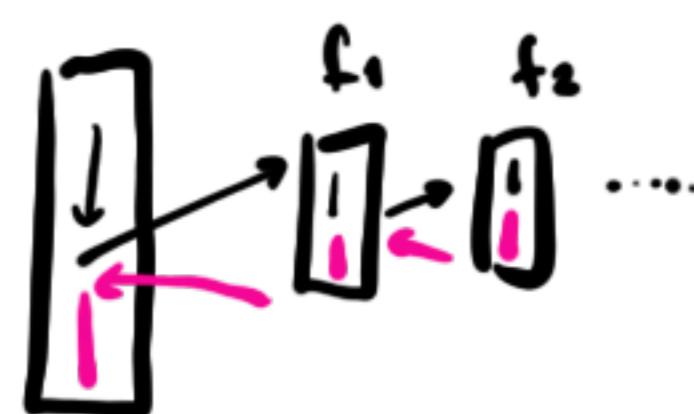
- Altprogram, → Tekrarlı olarak çağırılabilen kod blokları
- Alrutin, → Altprograma benzer data ile sırayla işlem yapma amacıyla kullanılır. Kontrol programın ana akışına döner.
- prosedür,
- metot, → Class içindeki fonksiyonlar. Nesneye özgüdür.
- fonksiyon → Bir değer döndüren altprogramlardır.

Deger  
döndürmez

Sonuç üretmek  
yerine bir görevi yerine getirir.

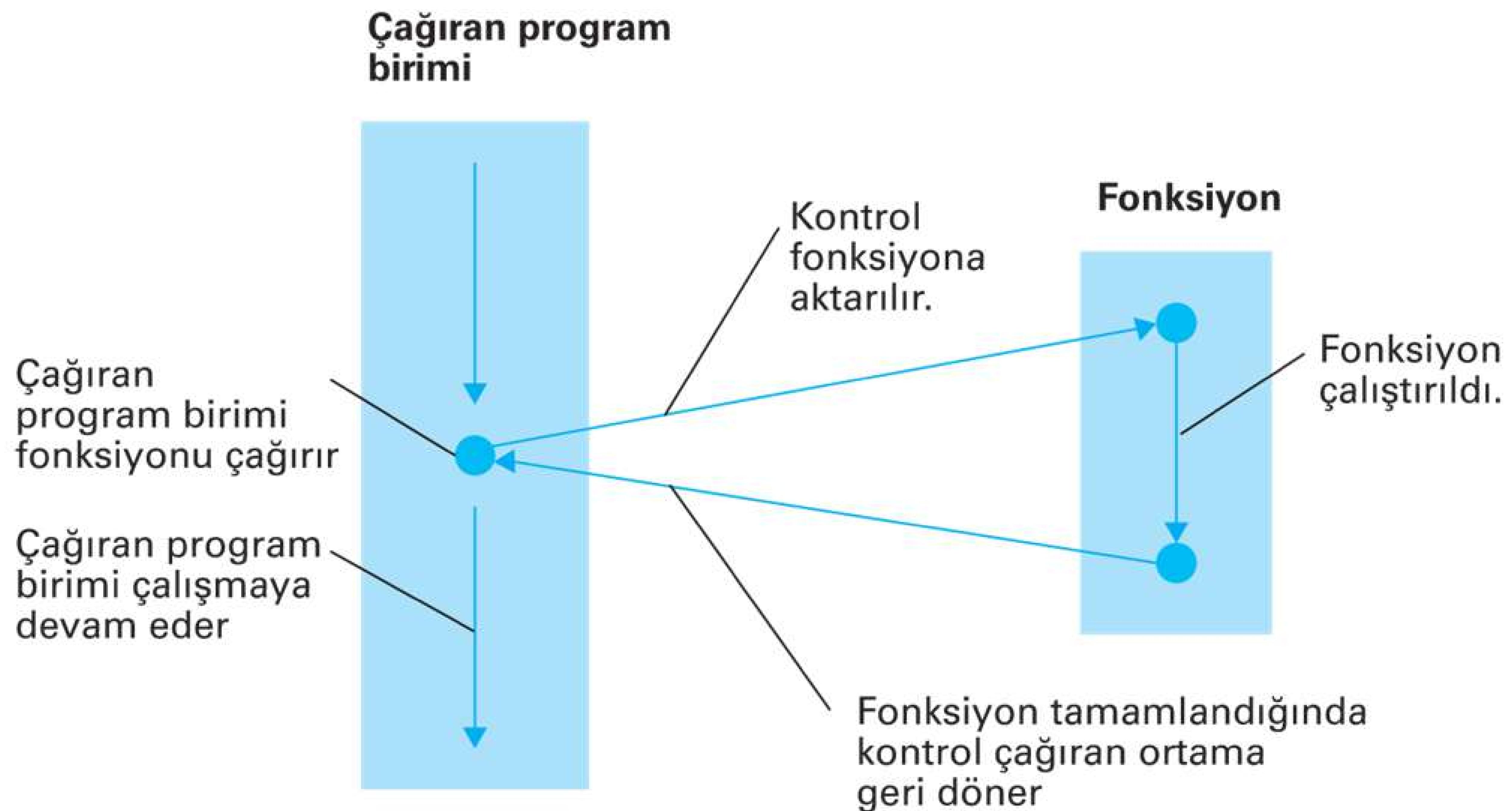
Belli bir girdiyi alır ve bir sonuc üretir.

Sınıfın özelliklerini ve davranışlarını kullanabilir.



bellalle kopiler  
olusur

## Şekil 6.8 Bir fonksiyon içeren kontrol akışı



# Şekil 6.9 C dilinde yazılmış ProjectPopulation fonksiyonu

Başlığı "void" terimi ile başlatmak C programcısının, program biriminin geri değer döndürmeyeceğini belirtme metodudur. Geri dönen değerleri kısa zamanda öğreneceğiz

Resmi parametre listesi. Birçok programlama dilinde olduğu gibi C dili de her bir parametrenin veri türünün belirtilmesini gerektirir.

```
void ProjectPopulation (float GrowthRate)
```

```
{ int Year;
```

Year adında  
lokal değişken tanımlar.

```
Population[0] = 100.0;  
for (Year = 0; Year <= 10; Year++)  
Population[Year+1] = Population[Year] + (Population[Year] * GrowthRate);  
}
```

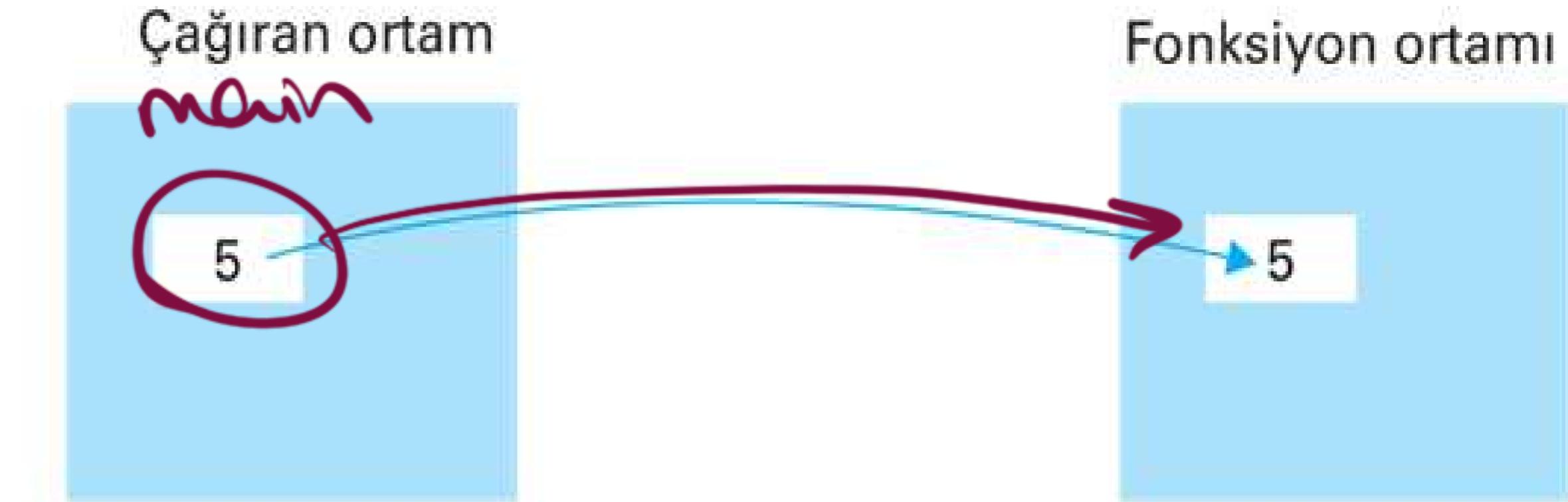
Bu deyimler popülasyonun nasıl hesaplanacağını ve Population adındaki global bir dizide saklanacağını belirtir.

# Şekil 6.10 Demo fonksiyonunun çalıştırılması ve parametreleri değer ile aktarma



Call by Value

- a. Fonksiyon çağrııldığı zaman verinin bir kopyası fonksiyona verilir



- b. ve fonksiyon kopyayı işler.



- c. Böylece fonksiyon sonlandığında çağrıran ortam değişmemiş olur.



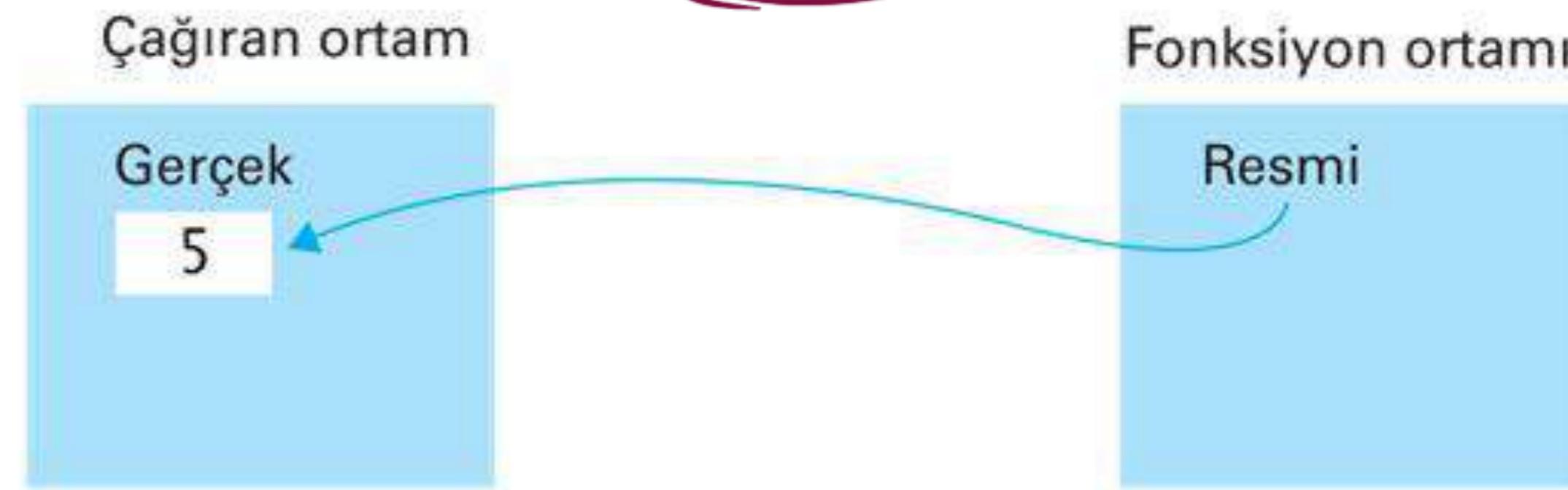
# Şekil 6.11

## Demo

### fonksiyonunun çalıştırılması ve referans ile parametre aktarma

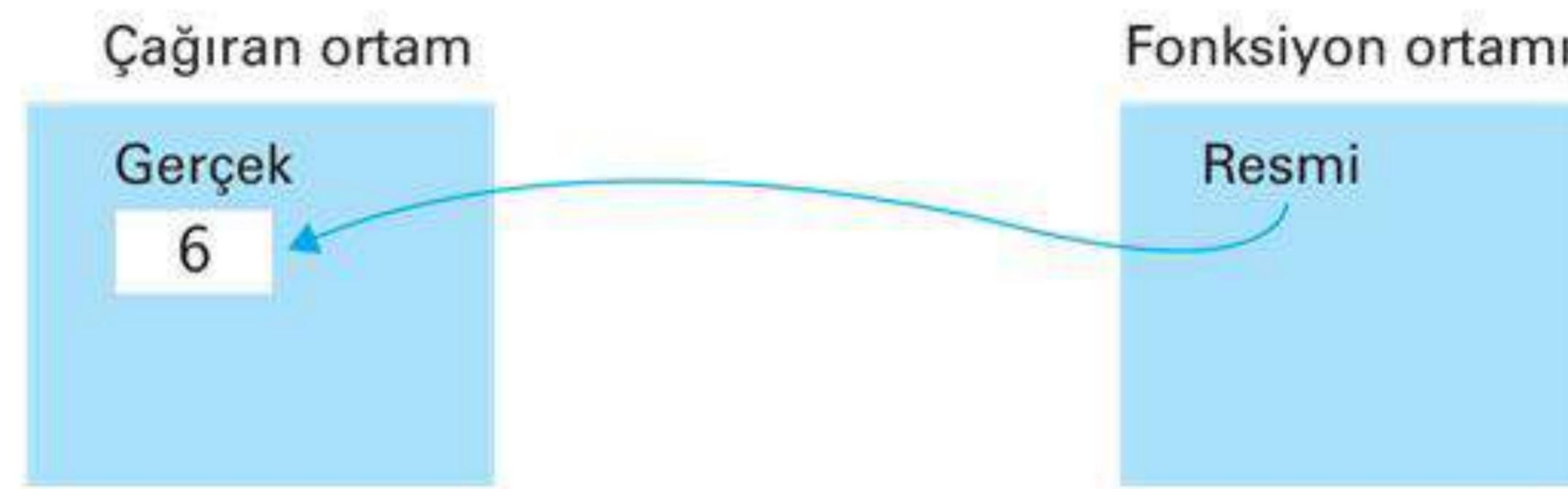
Call by  
Reference

- a. Fonksiyon çağırıldığında, resmi parametre gerçek parametreye yapılan bir referans haline gelir.



C++  
C

- b. Böylece, fonksiyon tarafından yönetilen değişiklikler gerçek parametreye yapılır.



- c. ve bu nedenle fonksiyon tamamlandığında korunmuş olur.



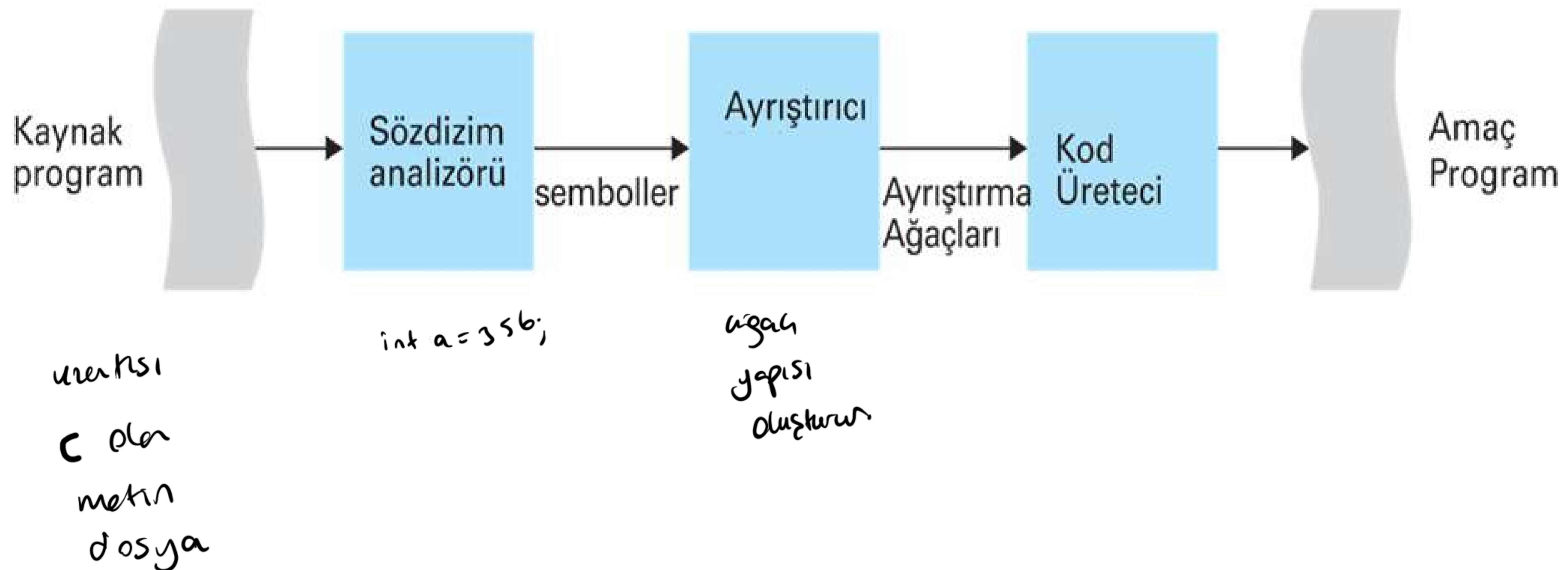
[tutorialspoint.com/compile-C-online.php](http://tutorialspoint.com/compile-C-online.php)

adresini deðilde kendisini gonderen ana programa yasimasi icin  
ana programda return

## 6.4 Programlama Dili Tasarlanması

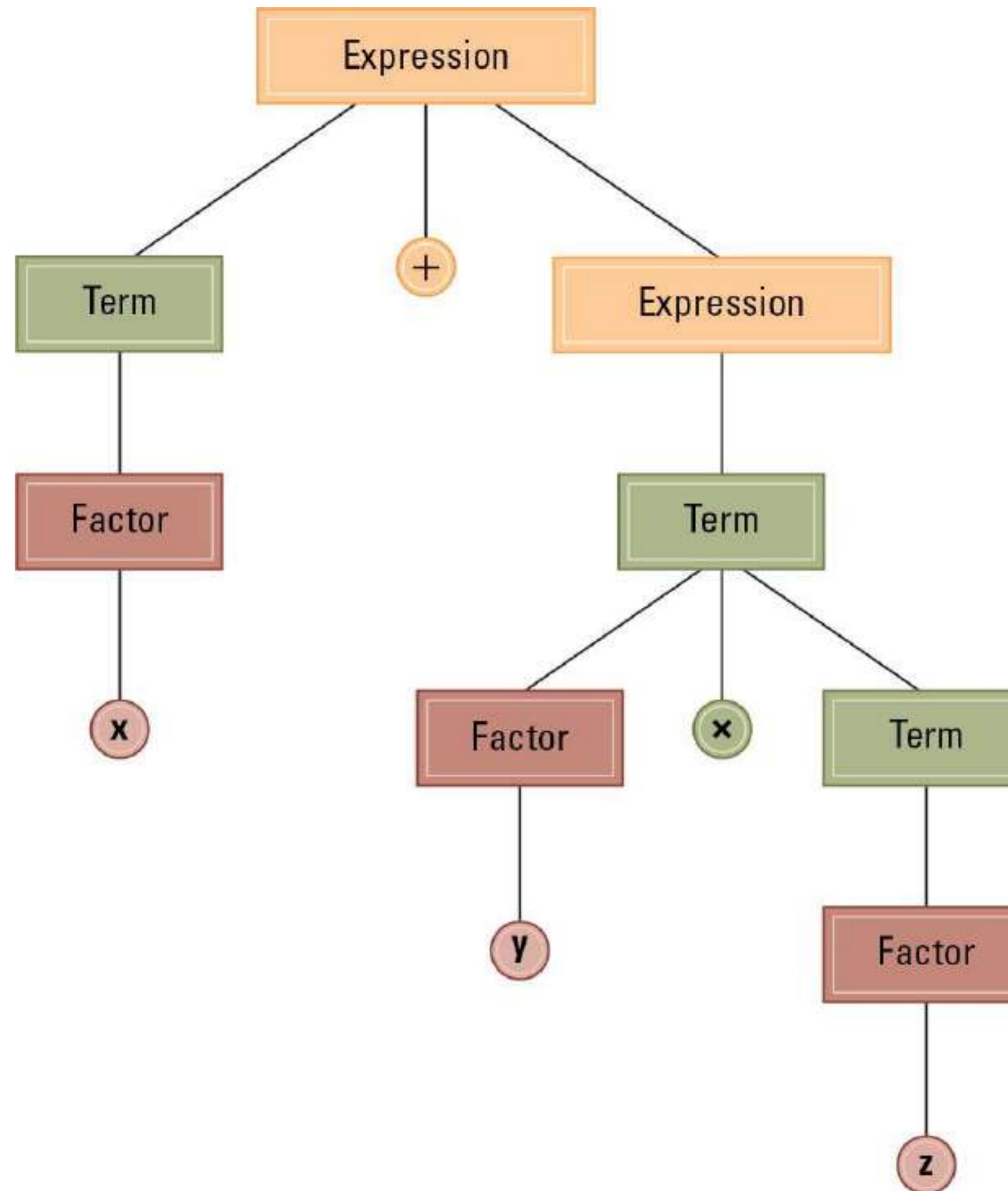
- Yüksek seviye bir dille yazılmış bir programın makine tarafından çalıştırılabilir forma çevrilmesi işidir.
  - **Sözdizim** analizörü, tek bir işaret ya da varlıkla gösterilen sembol dizisini anımsar.
  - **Ayrıştırıcı**, işaretleri bir araya getirip durumlara dönüştürür. Bunu ayrıştırma ağaçları üretmek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

## Şekil 6.13 Çevrim süreci

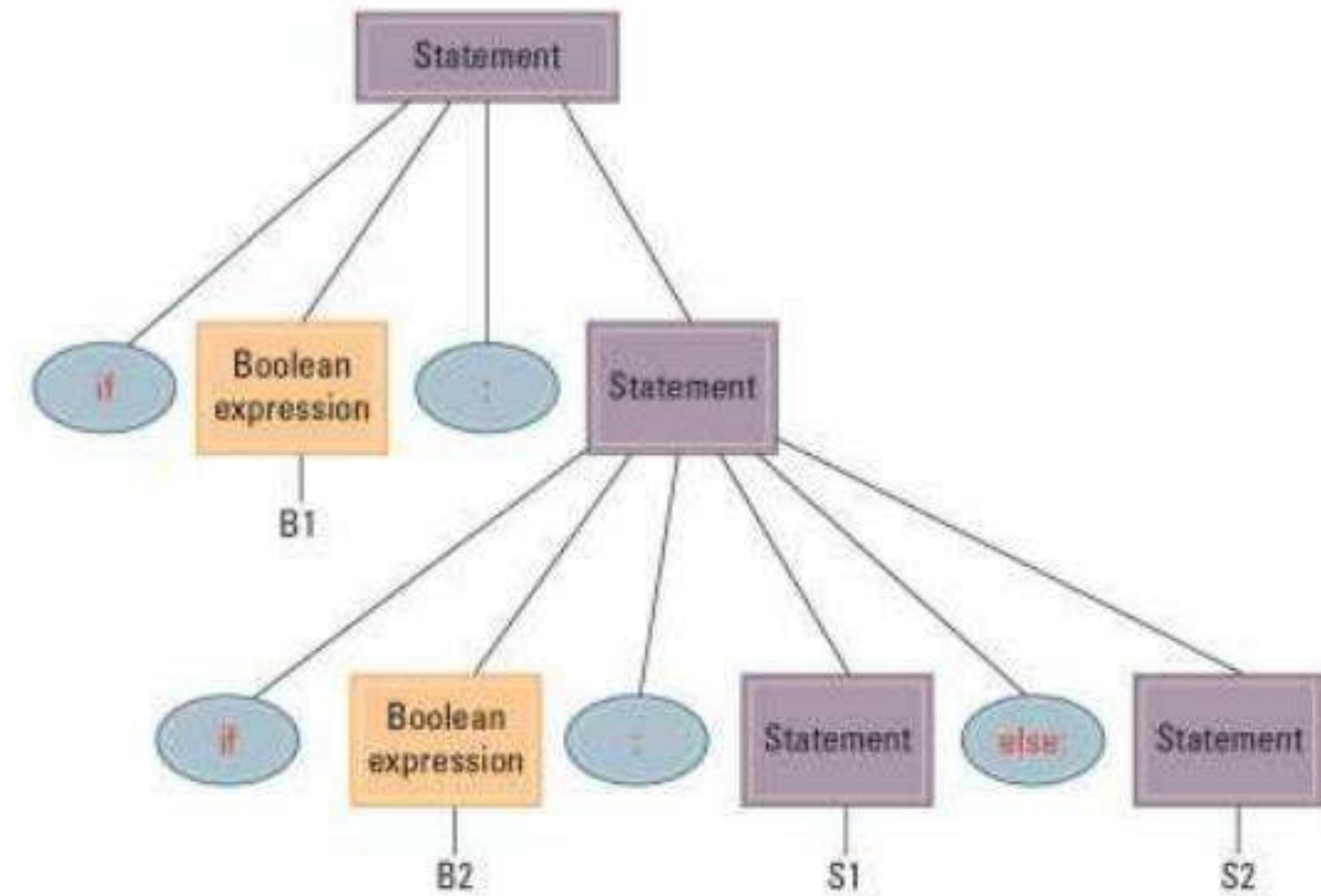
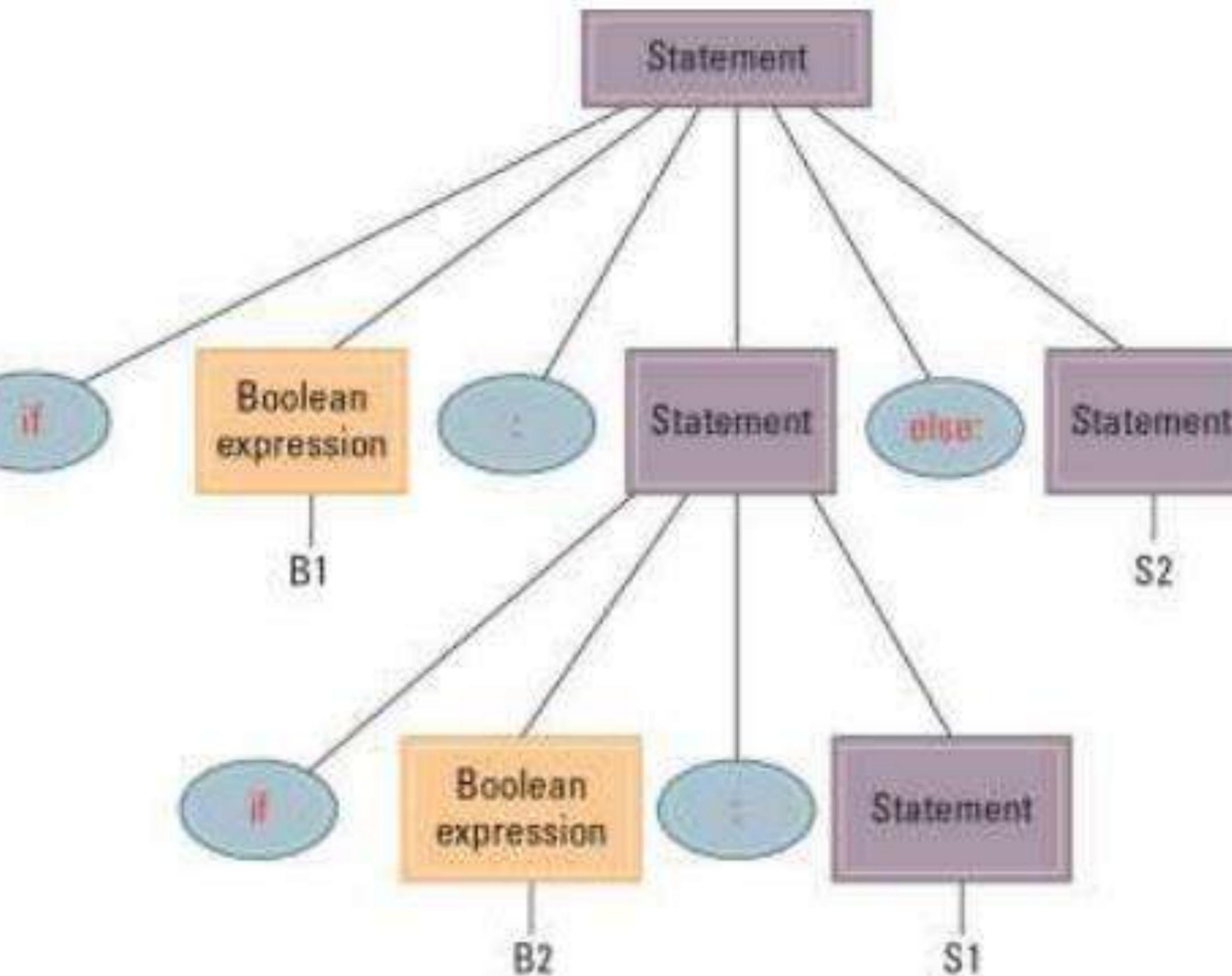
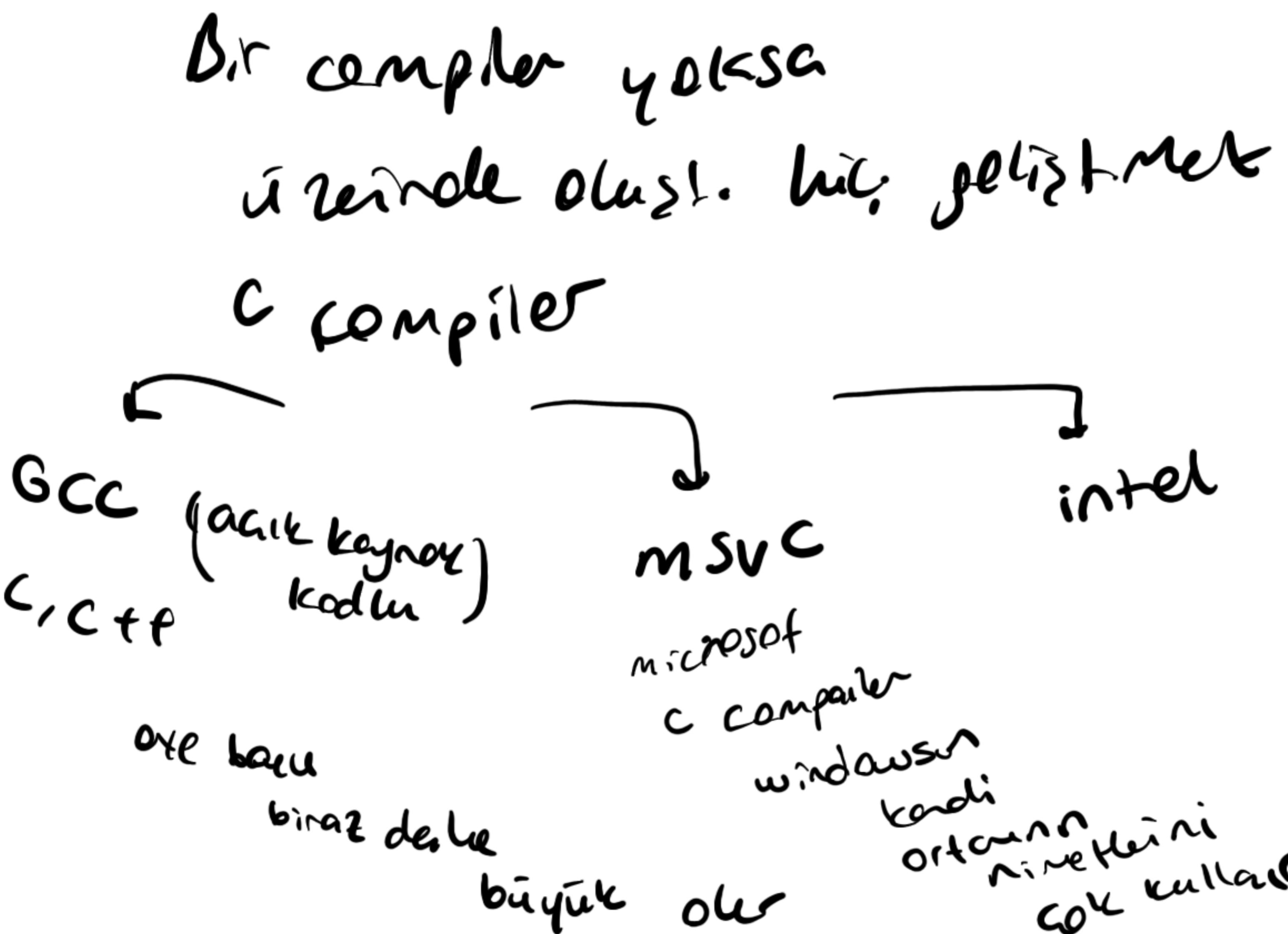


# Şekil 6.16 Şekil 6.17'teki sözdizimi şemalarını temel alan $x+y*z$ ifadesi için bir ayrıştırma ağacı

AST ( Abstract Syntax Tree )



# Şekil 6.17 Basit cebirsel bir ifadeyi tanımlayan sözdizimi şeması



## 6.5 Nesneye yönelik programlama

- **Nesne:** Hem veri hem prosedürleri içeren aktif program birimidir  
O sınıfın özelliklerini gösteren bir bilgesidir 1981
- **Sınıf:** Nesne topluluğundan oluşan bir şablondur C ile  
class ↓

Nesneye bir sınıfın **örneği** denir.

Sırmgarlı  
↑  
Sınıf

Maynay  
↑  
Nesne

C with  
classes

C++

1995 ↓

(Nesneye Yönelik Prg. 2.0'ncı) Java

Nesneler ve bu nesnelerin birbiryle ola ilişkileri

## Şekil 6.19 Bir bilgisayar oyunundaki lazer silahını tanımlayan bir sınıfın yapısı

Bir çok  
eğer  
bi...  
Proje  
fü...  
n

Java C++

```
class LaserClass
```

```
{ int RemainingPower = 100;
```

```
    void turnRight ( )
```

```
    { ... }
```

```
    void turnLeft ( )
```

```
    { ... }
```

```
    void fire ( )
```

```
    { ... }
```

```
}
```

Bu türden olan her bir nesnenin içinde bulunan verinin tanımı

Bu türden bir nesnenin çeşitli mesajlara nasıl cevap vereceğini tanımlayan metodlar

# Nesnenin Bileşenleri

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

*constructor* → class 'ın adıyla aynı adı sahip olur  
farklıdır. *Otomatik* olabilir.

## Şekil 6.21 Yapıçı fonksiyonlu bir sınıf

Java Derleyicisi

JDK

Apache Netbeans

Eclipse

Oracle Java  
Günlük mesajları  
ticari uyg.

OpenJDK  
ticari sunucular  
Java

```
class LaserClass
{ int RemainingPower;
    LaserClass (InitialPower)
    { RemainingPower = InitialPower;
    }

    void turnRight ( )
    { ... }

    void turnLeft ( )
    { ... }

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

Nesne oluşturulduğunda  
yapıcı RemaininPower'a  
bir değer atar

otomatik  
giriş

↳ Yapı

ilk başta main'i barındırır  
class çalışmaya başlar.

Kaynak  
geliştirmesi  
ayrı  
bağımsız  
alt  
bileşenler  
ne  
gelistişir  
olarak

# Nesne Bütünlüğü

- **Kapsülleme:** Nesnenin iç bileşenlerine erişimi kontrol etmenin bir yolu
  - Public
  - Private *Bir nesneyi veya nesnenin metodlarını private ettiğimizde dışarıya erişilemeye olur.*  
*fire sadece o classın kendi içinde kullanılabiliyor.*

## 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.

syntax hatalarında real time'da vir

calıstırınca çok belirli türkestr.

```
class LaserClass
{
    private int RemainingPower;

    public LaserClass (InitialPower)
    {
        RemainingPower = InitialPower;
    }

    public void turnRight ( )
    {
        ...
    }

    public void turnLeft ( )
    {
        ...
    }

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

BİL !

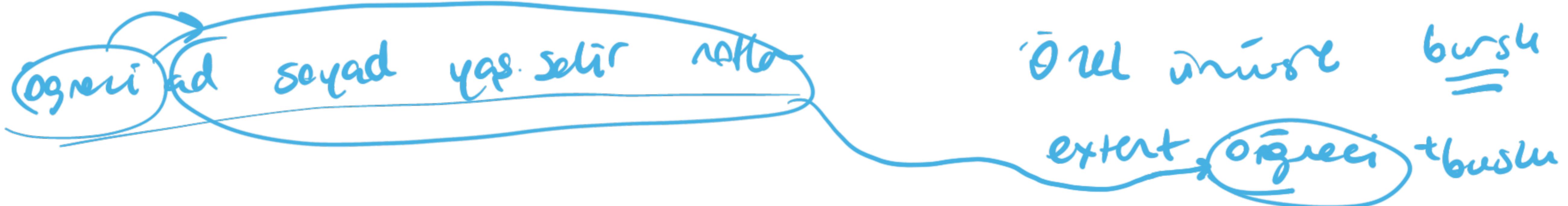
kapıca  
fonksiyon  
nedir ?

class  
nedir ?

# Diğer Nesneye Yönelik Programlama Kavramları

Public  
Private  
Protected

- **Miras Alma (Inheritance):** Önceden tanımlanmış sınıfların terimlerini yeni sınıflarda kullanmaya olanak sağlar
- **Çok Biçimlilik (Polymorphism):** *Fark yada metod ayrı ama çağırıldığı degisir.*
  - Bir türün bir başka tür gibi davranışabilme 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 gereklidir
  - Zaman paylaşımı CPU'lar yardımıyla simül edilebilir

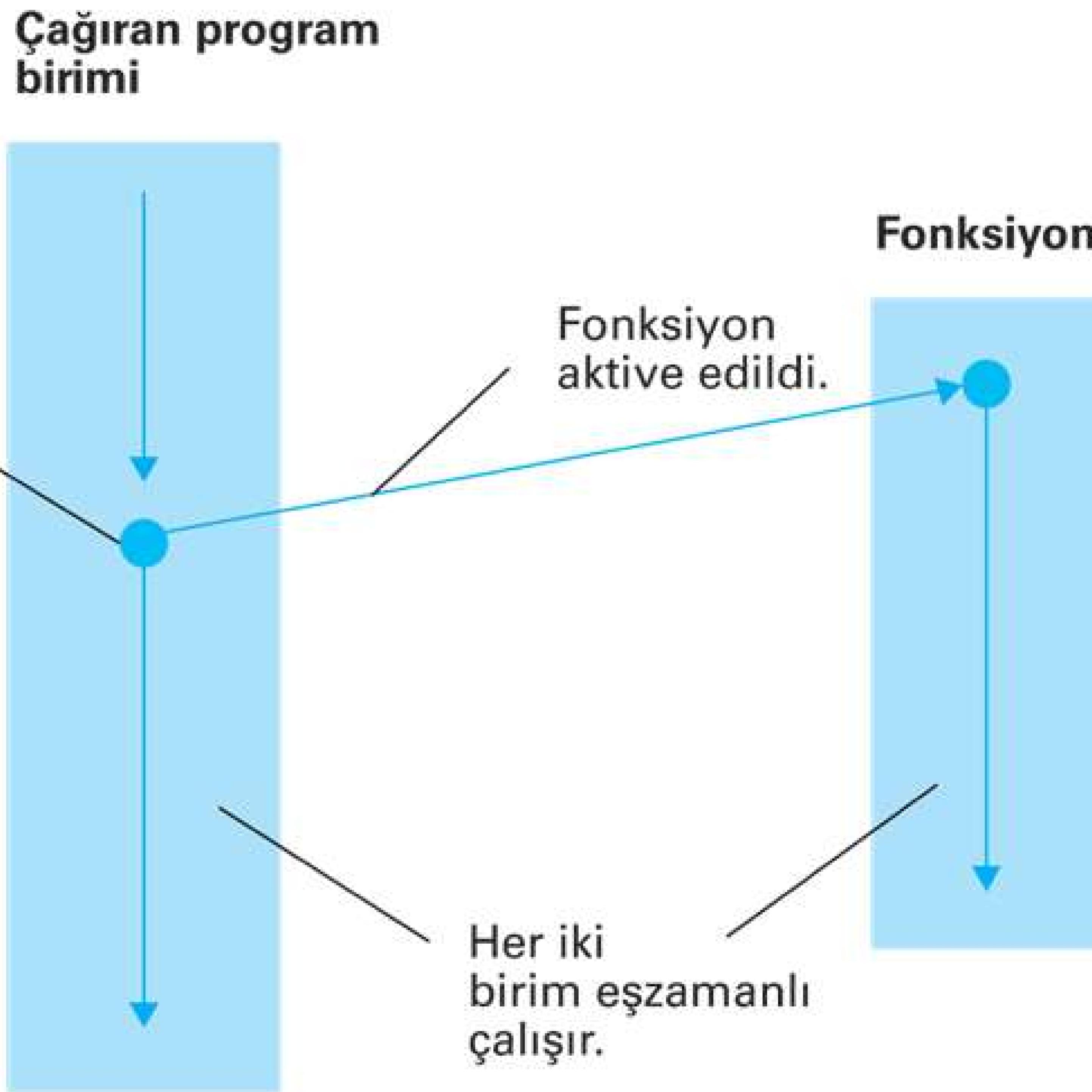
örnek  
paralel işlemi göster → 8 adımda 1: qada 7 s:  
çözüme

## Şekil 6.23 İş parçacığı (thread) oluşturulması

thread

Fonk. başka bir  
nedenle çalışmıyor.

Çağırılan  
program birimi  
fonksiyonu ister.

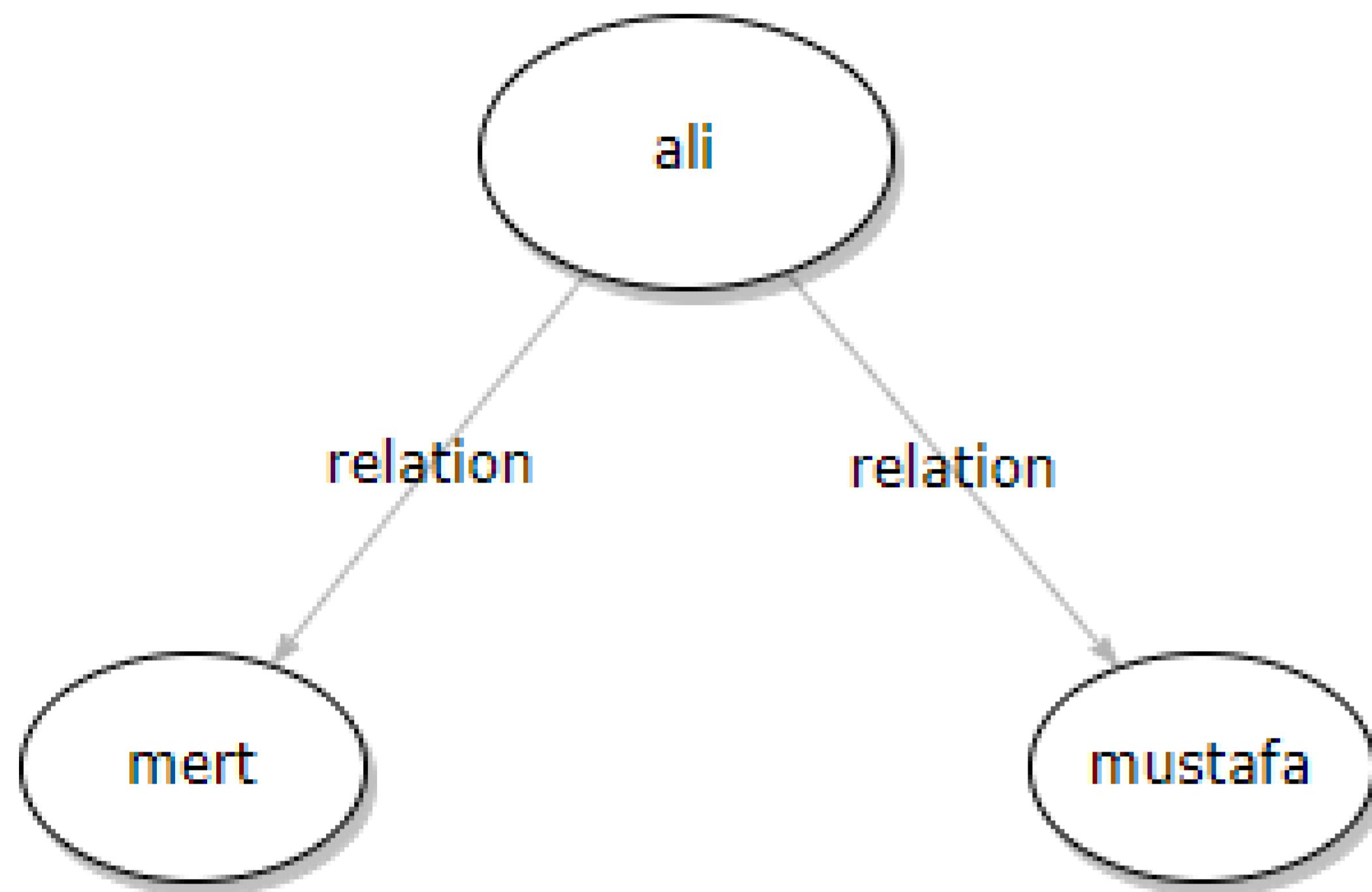


## 6.7 Bildirimsel Programlama

- **Çözüm:** İki veya daha fazla durumu bir araya getirerek tek bir durum ortaya çıkarmak (bu orjinalinin mantıksal bir sonucudur).
  - Örnek:  $(P \text{ OR } Q) \text{ AND } (R \text{ OR } \neg Q)$   
 $(P \text{ OR } R)$  sonucuna tekabül eder
- Bildirimsel Programlama örneği: PROLOG

Mantıksal işlevler. (Döngü, dizi, değişken sileri  
yapıları clası'nu aralarda kullanın) *(Handwritten note)*

# Prolog Örneği



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

↓  
operator  
=

↓  
!=

Standart  
Java  
esnek