

Bölüm 10
Yapılar ve Birleşimler

- Yapı Tanımı
- Yapılara İlk Değer Atama
- Yapı Elemanlarına Erişim
- İçiçe Yapılar
- Yapılar ve Fonksiyonlar
- Yapı Dizileri
- Veri Tipi İsmi Tanımlama
- Birleşimler

Yapı Tanımı

Yapılar, birden fazla, aynı ya da farklı tipte eleman tutabilen bileşik veri tipleridir.

```
struct yapı_tipi {
    veri_tipi   yapı_elemanı1;
    veri_tipi   yapı_elemanı2;
                    .
                    .
                    .
};
```

Örnek:

```
struct ogrenci {
    char isim[15];
    int yas;
    float ortalama;
    char cinsiyet;
};
```

Yapı Tanımı

Bir yapı değişkeni aşağıdaki şekilde tanımlanır.

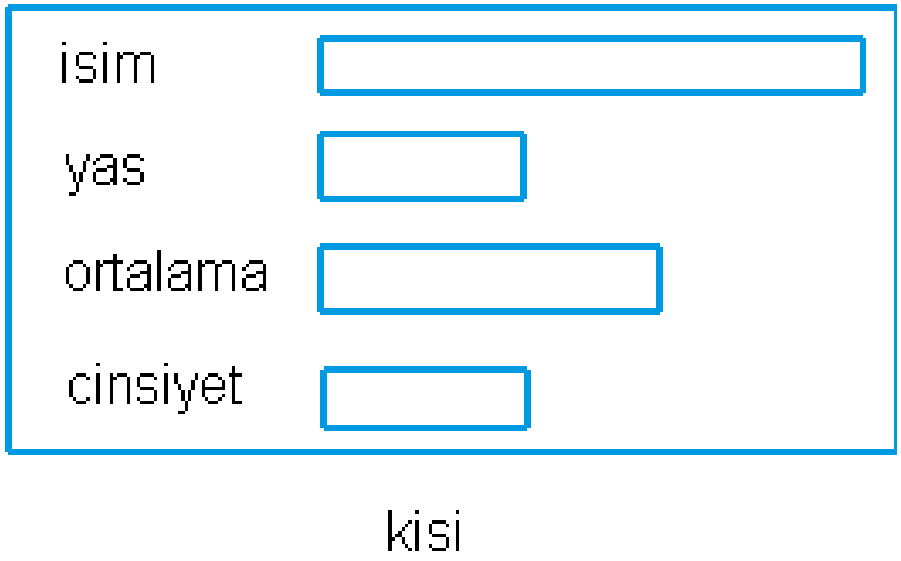
struct yapı_tipi yapı_değişkeni;

Örnek:

```
struct ogrenci {
    char isim[15];
    int yas;
    float ortalama;
    char cinsiyet;
};

struct ogrenci kisi;
```

Bellek Görüntüsü



Yapı Tanımı

```
struct yapı_tipi {
   veri_tipi   yapı_elemanı1;
   veri_tipi   yapı_elemanı2;
                .
                .
                .
}yapı_değişkeni;
```

Örnek:

```
struct ogrenci{
   char isim[15];
   int yas;
   float ortalama;
   char cinsiyet;
}kisi;
```

Yapı Tanımı

```
struct {
   veri_tipi  yapı_elemanı1;
   veri_tipi  yapı_elemanı2;
                .
                .
                .
}yapı_değişkeni;
```

Örnek:

```
struct{
   char isim[15];
   int yas;
   float ortalama;
   char cinsiyet;
}kisi;
```

Yapı Tanımı

```
struct {
   veri_tipi  yapı_elemanı1;
   veri_tipi  yapı_elemanı2;
                .
                .
                .
}yapı_değişkeni;
```

Örnek:

```
struct{
   char isim[15];
   int yas;
   float ortalama;
   char cinsiyet;
}kisi;
```

Örnek:

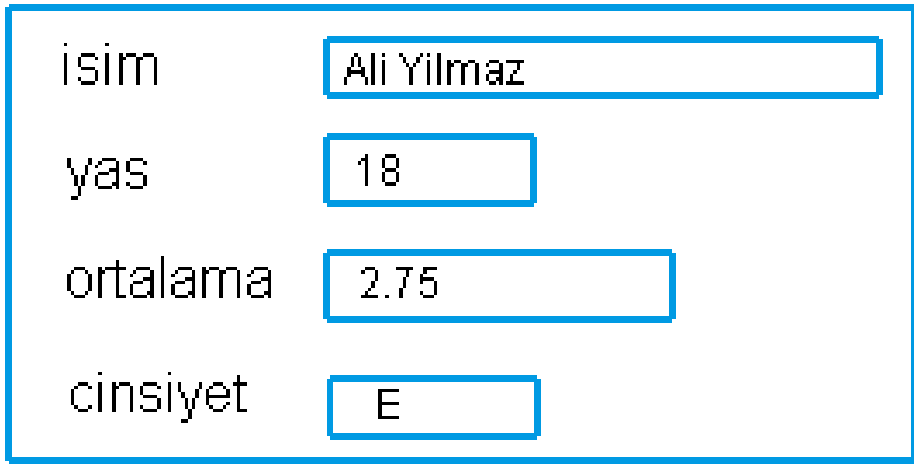
```
struct ogrenci{
        .
        .
        .
}bir,iki,uc;
```

Yapılara ilk Değer Atama

Örnek:

```
struct ogrenci {
   char isim[15];
   int yas;
   float ortalama;
   char cinsiyet;
   };
```

Bellek Görüntüsü




```
struct ogrenci kisi = {"Ali Yilmaz",18,2.75,'E'};
```

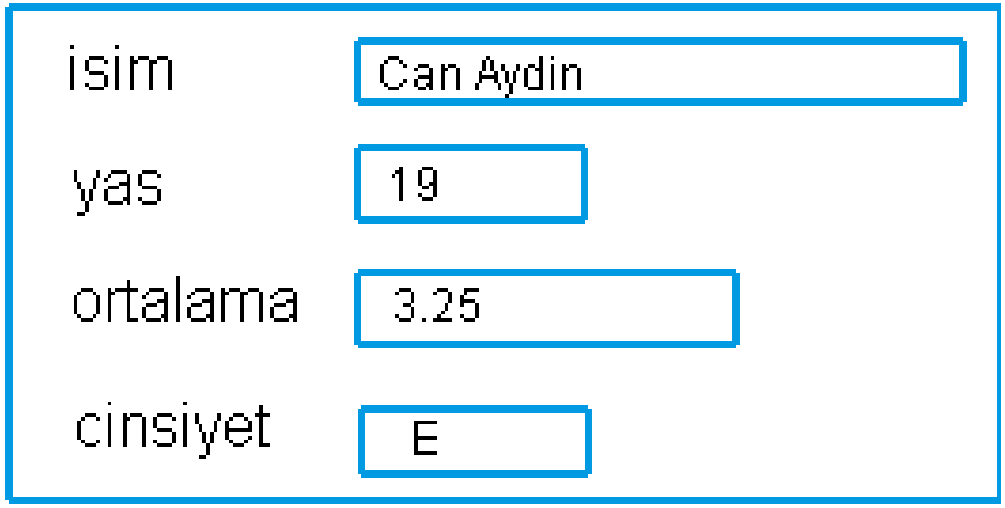
Yapı Elemanlarına Erişim

Yapı elemanlarına program içinde nokta '.' operatörüyle erişmemiz gerekir.

yapı_adı.eleman_adı

Örnek:

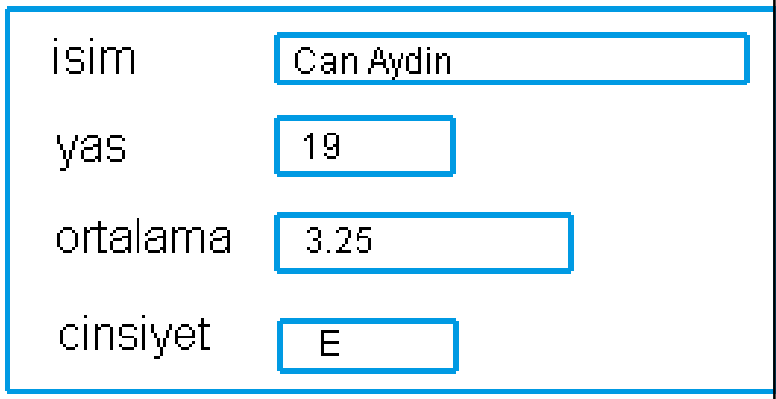
```
struct ogrenci {
   char isim[15];
   int yas;
   float ortalama;
   char cinsiyet;
};
```



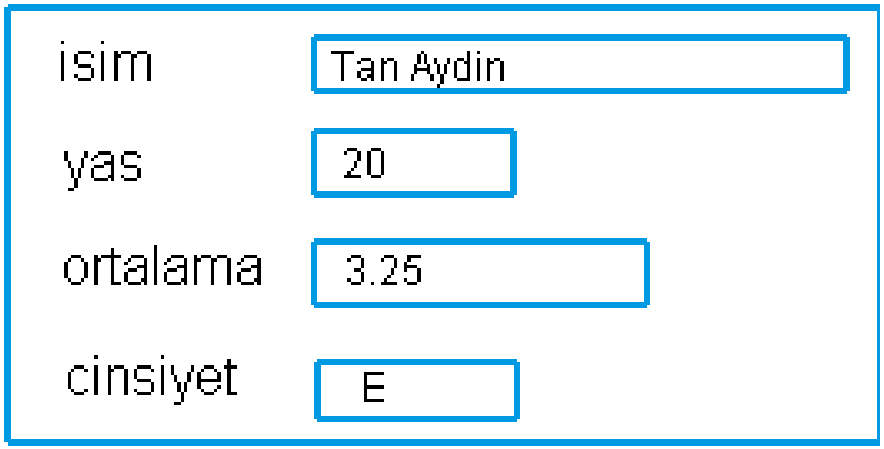

```
struct ogrenci kisi = {"Can Aydin",19,3.25,'E'};
```

Yapı Elemanlarına Erişim

```
kisi.isim[0] = 'T';
kisi.yas = 20;
scanf ("%c", &kisi.cinsiyet);
printf("%s\n", kisi.isim);
if (kisi.ortalama > 3.0)
    printf ("Burs kazandiniz.\n");
```



kisi



kisi

Çıktı:

Tan Aydin

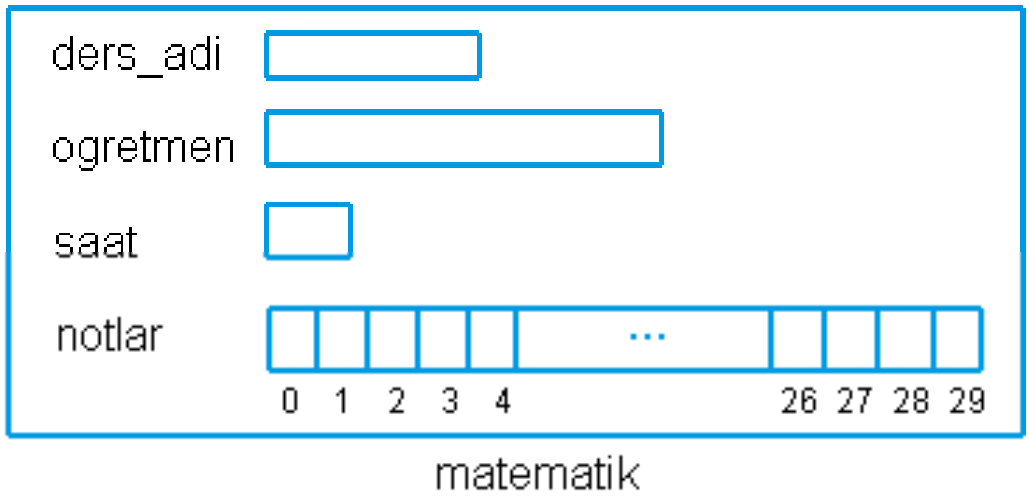
Burs kazandiniz.

Yapı Elemanlarına Erişim

Yapı elemanları, tamsayı, karakter gibi temel tiplerden olabildiği gibi dizi gibi bileşik değişkenlerden de oluşabilir.

Örnek:

```
struct ders {
        char ders_adi[10];
        char ogretmen[20];
        int saat;
        int notlar[30];
        };
struct ders matematik;
.
.
.
for (i=0; i<30; i++)
scanf("%d",&matematik.notlar[i]);
```



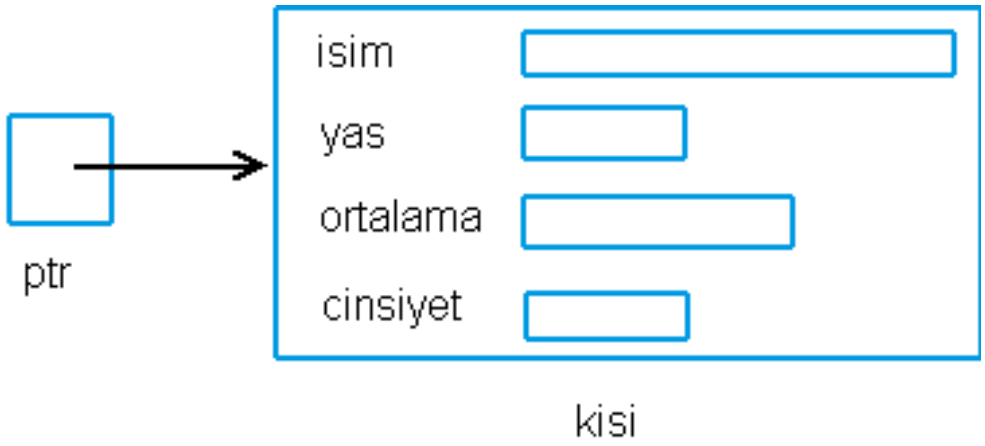
Yapı Elemanlarına Erişim

Eğer yapımıza bir gösterge ile erişmek istersek '->' ***dolaylı alan seçme*** operatörünü aşağıda gösterildiği gibi kullanmamız gerekir.

yapı_göstergesi -> eleman_adı

Örnek:

```
struct ogrenci {
      char isim[15];
      int yas;
      float ortalama;
      char cinsiyet;
   };
struct ogrenci kisi;
struct ogrenci *ptr =&kisi;
...
(*ptr).yas = 17;   veya    ptr->yas = 17;
```



İçiçe Yapılar

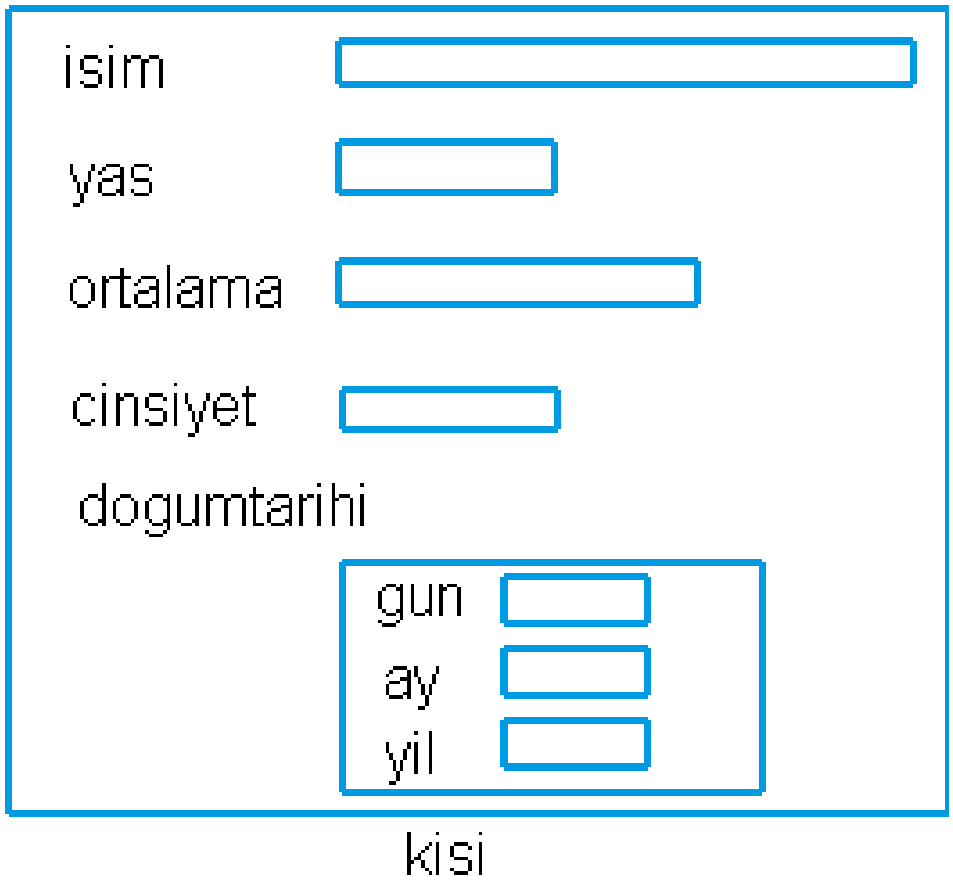
Bir yapının başka bir yapı tipinde elemanı olabilir.

Örnek:

```
struct tarih
{   int gun;
    char ay[10];
    int yil;
};


struct ogrenci
{   char isim[15];
    int yas;
    float ortalama;
    char cinsiyet;
    struct tarih dogumtarihi;
 }kisi;
```

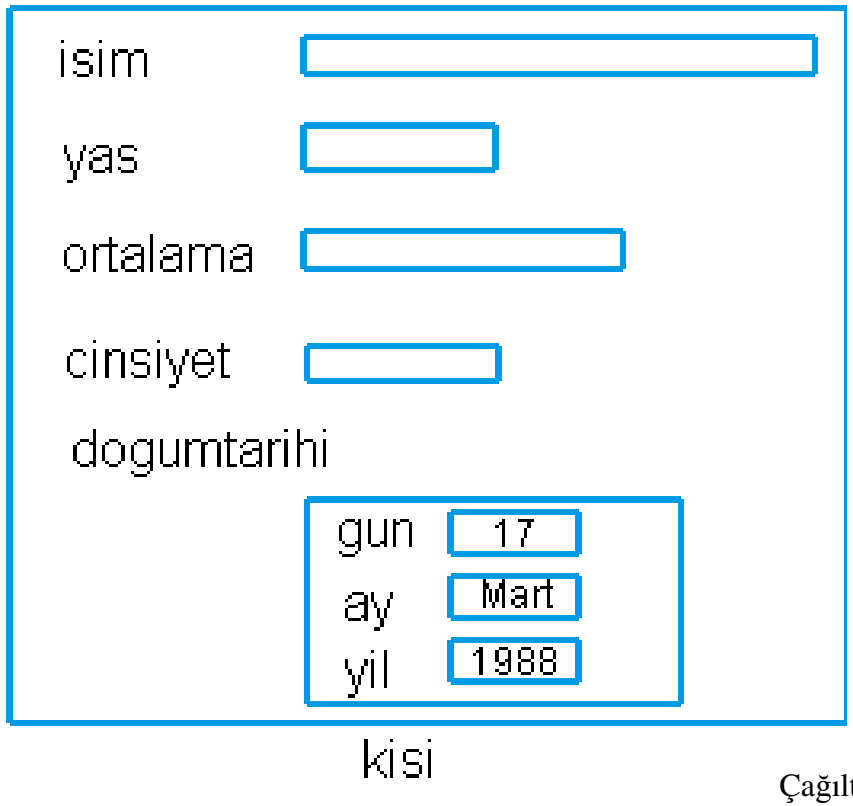
Bellek Görüntüsü



Yapı Elemanlarına Erişim:

```
scanf("%d",&kisi.dogumtarihi.gun);
gets(kisi.dogumtarihi.ay);
kisi.dogumtarihi.yil = 1988;
```

Bellek Görüntüsü



Yapı Değerinin Döndürülmesi

```
struct ogrenci {
  char isim[15];
  int yas;
  float ortalama;
  char cinsiyet;
};
struct ogrenci fon(void);
int main(void){
  struct ogrenci kisi;
  kisi=fon();
       . . .
}
struct ogrenci fon(void){
  struct ogrenci a;
       . . .
  return (a);
}
```

Yapı Elemanının Parametre Olarak Fonksiyona Geçirimi

```
struct ogrenci {
   char isim[15];
   int yas;
   float ortalama;
   char cinsiyet;
};
void fon(int x) {
        . . .
}
int main(void){
    struct ogrenci kisi;
    fon(kisi.yas);
}
```

Yapının Değer Parametresi Olarak Fonksiyona Geçirimi

```
struct ogrenci {
   char isim[15];
   int yas;
   float ortalama;
   char cinsiyet;
};
void fon(struct ogrenci yapı_parametresi){
    yapı_parametresi.yas=20;
         . . .
}
int main(void){
    struct ogrenci kisi;
    fon(kisi);
}
```

Yapının Kaynak Parametresi Olarak Fonksiyona Geçirimi

```
struct ogrenci {
   char isim[15];
   int yas;
   float ortalama;
   char cinsiyet;
};
void fon(struct ogrenci *yapi_gostergesi){
    yapi_gostergesi->yas=20;
         . . .
}
int main(void){
    struct ogrenci kisi;
    fon(&kisi);
        . . .
}
```

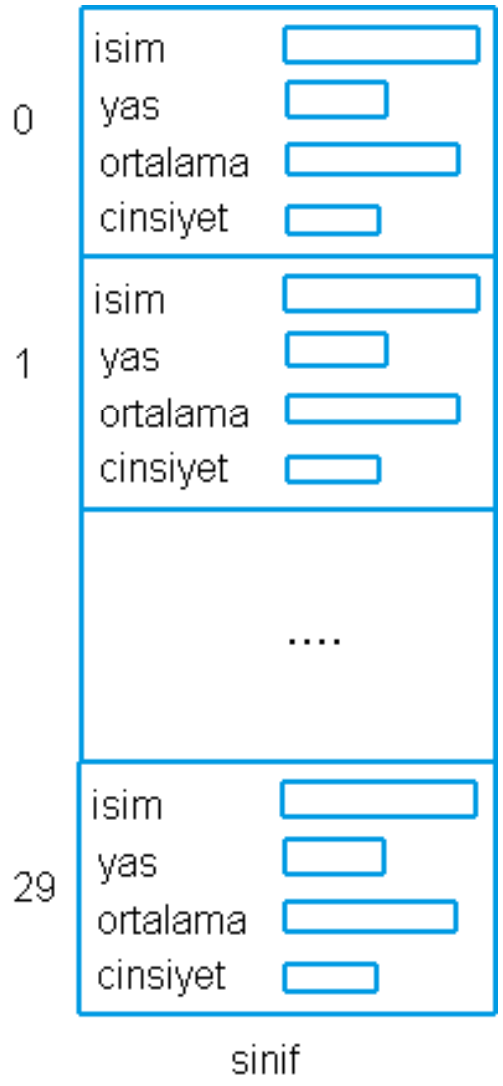
Yapı Dizileri

Aynı tipte birden fazla yapı değişkenine ihtiyacımız varsa bir yapı dizisi oluşturabiliriz.

Örnek:

```
struct ogrenci {
   char isim[15];
   int yas;
   float ortalama;
   char cinsiyet;
};

struct ogrenci sinif[30];
```

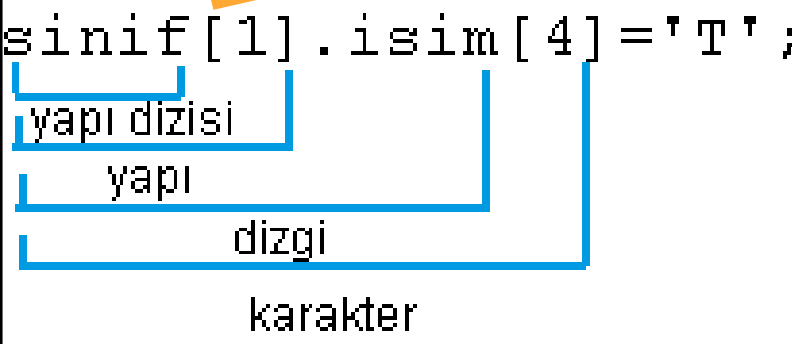


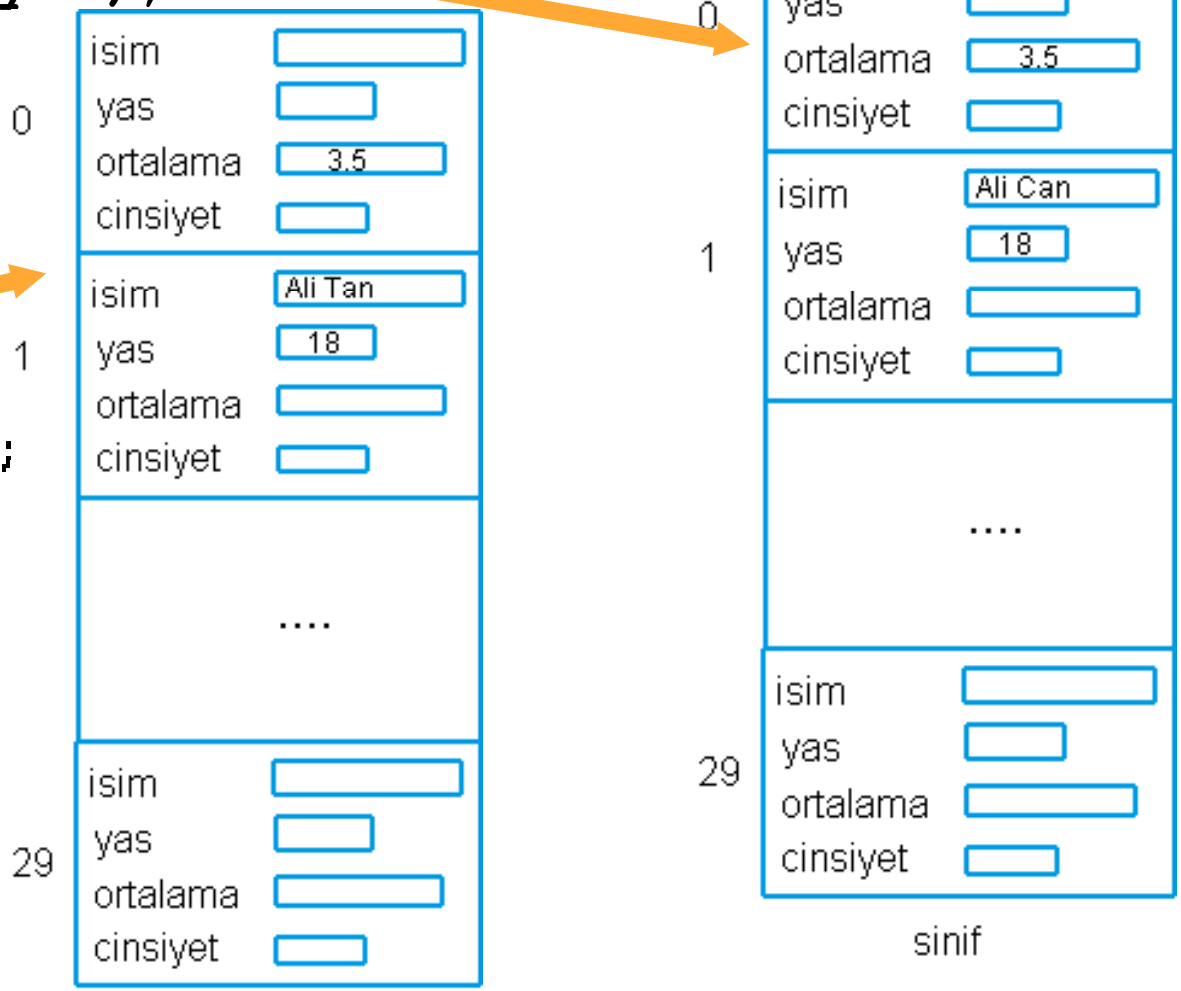
Yapı Dizileri

```
sinif[0].ortalama = 3.5;
scanf("%d",&sinif[1].yas);
gets(sinif[1].isim);
```

Girdi:

```
18
Ali Can
```





Yapı Dizileri

Örnek Bu programda 30 kişilik sınıf bilgilerini bir fonksiyon kullanarak okuyalım ve en yüksek ortalamalı öğrenciyi bulup ismini ve not ortalamasını yazdıralım.

```
#include <stdio.h>
 #include <string.h>
 struct ogrenci
 {
    char isim[15];
    int yas;
    float ortalama;
  };
  void oku(struct ogrenci s[]);
  int main(void)
  {
     struct ogrenci sinif[30];
     float enbuyuk = 0.0;
     char birinci[15];
     int i;
```

Yapı Dizileri

```
/* sinif dizisini okuyalim   */
    oku(sinif);
    /* 30 ogrenciden en yüksek ortalamayi tutturan
     ogrencinin ortalamasini ve adini saklayalim */
    for (i=0;i<30;i++)
    {
       if ((sinif[i].ortalama)>enbuyuk)
          {  enbuyuk = sinif[i].ortalama;
             strcpy(birinci,sinif[i].isim);
          }
    }
    printf("Sinif birincisi: %s",birinci);
    printf("Ortalama: %5.2f",enbuyuk);
 }
 /* yapi dizisini okuyan fonksiyon */
 void oku (struct ogrenci s[])
 {
    int i;
    for (i=0;i<30;i++)
    { scanf("%s,",s[i].isim);
      scanf("%d %f",&s[i].yas,&s[i].ortalama);
    }
 }
```

Veri Tipi İsmi Tanımlama

Tanımlanmış bir veri tipine yeni bir isim vermek amacıyla ***typedef*** bildirimi kullanılır.

Örnek:

```
typedef int Tam;
Tam x;
```

Örnek:

```
struct ogrenci {
   char isim[15];
   int yas;
   float ortalama;
   char cinsiyet;
};
typedef struct ogrenci Ogr;
```

veya

```
typedef struct {
   char isim[15];
   int yas;
   float ortalama;
   char cinsiyet;
}Ogr;
```

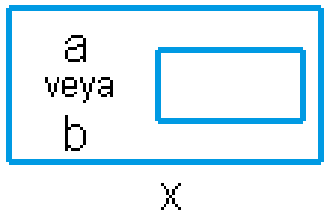
```
Ogr kisi;
Ogr sinif[30];
```

Birleşimler

Birleşimler, yapılara benzer şekilde aynı veya farklı tipteki elemanlardan oluşur; ancak tüm elemanlar bellekte aynı yeri kullanırlar.

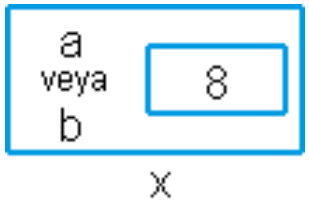
Birleşim_adı.eleman_adı
Birleşim_göstergesi->eleman_adı

```
union numara
{
   int a;
   double b;
};
```

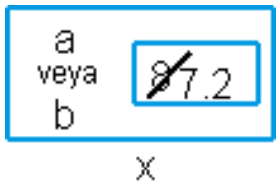



```
union numara x;
union numara x = {5};
```

```
x.a = 8;
```




```
x.b = 7.2;
```



Birleşimler

Örnek:

```
struct {
    char isim[30];
    int no;
    char calisma_tipi[4];
    union {
        double aylik;
        int saat;
    }bilgi;
}calisanlar[100];
...
strcpy(calisanlar[0].calisma_tipi,"yari");
calisanlar[0].bilgi.saat = 10;
```

