

ALGORİTMA VE PROGRAMLAMA II

HAFTA#2

Genel Bakış...

2

- **Yapılar ve Birlikler**
 - **enum** Deyimi / Sayma Sabitleri (Enumeration Constants)
 - **struct** Deyimi / Yapı (Structures)
 - **typedef** Deyimi
 - **union** Deyimi / Birlik

2. BÖLÜM

3

Enumeration

enum Deyimi

4

- **Sayma tipi/sabiti** olarak adlandırılırlar.
- Bir sayma tipi, **enum** anahtar kelimesiyle tanıtılır ve tanıtıcılar ile temsil edilen **tamsayı sabitlerinin kümesidir**.
- Bu sayma sabitleri, değerleri otomatik olarak belirlenen sembolik sabitlerdir.
- enum içindeki **değerler** aksi belirtilmedikçe 0 ile başlar ve **1** arttırılır.
- Bu tip, **değişkenin** alabileceği **değerlerin sabit** olduğu durumlarda programı daha **okunabilir** hale getirmek için kullanılır.

enum Deyimi (devam...)

5

- Mantıksal örnekler:
 - **Cinsiyet:** Kadın, Erkek
 - **Calisma_Durumu:** İssiz, Calisan, Öğrenci, Emekli
 - **Evlilik_Durumu:** Evli, Bekar
 - **Egitim_Durumu:** İlkokul, Lise, OnLisans, Lisans, LisansUstu
 - **Menu_Turu:** Topla, Cikart, Bol, Carp

enum Deyimi (devam...)

6

- Hangisi daha **anlaşılır**?

```
if (kisi_cinsiyet == 1)
{
}
```

?

```
if (kisi_cinsiyet == Erkek)
{
}
```

enum Deyimi (devam...)

7

- Genel yazım biçimi:

```
enum tip_adı
{
    değer_1,
    değer_2,
    ...,
    değer_n
};
```

enum Deyimi (devam...)

8

- Örnek:

```
enum eBolumler
```

```
{  
    programcilik,  
    donanim,  
    muhasebe,  
    motor  
};
```



Tanımı ile alınan değerler:

programcilik için **0**,
donanim için **1**,
muhasebe için **2** ve
motor için **3** değerini

Değerler, aksi belirtilmedikçe 0 ile başlar ve 1 arttırılır.

enum Deyimi (devam...)

9

- Örnek:

```
enum eBolumler
{
    programcilik,
    donanim,
    muhasebe,
    motor
};
```

```
void main ()
{
    enum eBolumler bolum_degiskeni;
    int bolum_degiskeni2;

}
```

Proje oluşturalım ve örneği uygulayalım...

Örnek: Basit enum Kullanımı

10

```
#include <stdio.h>
#include <stdlib.h>

enum eBolumler
{
    programcilik = 1, /* 1 */
    donanim,          /* 2 */
    muhasebe,          /* 3 */
    motor,              /* 4 */
    buro               /* 5 */
};

int main()
{
    enum eBolumler bolum_degiskeni;

    bolum_degiskeni = donanim;
    printf("Bolum : %d\n", bolum_degiskeni);

    bolum_degiskeni += 2;
    printf("Yeni bolum : %d\n", bolum_degiskeni);

    return 0;
}
```

```
Bolum : 2
Yeni bolum : 4
```

Alıştırma: **enum** Oluşturma

11

- Aşağıdaki bilgilerin **enum** tiplerini oluşturunuz:
 - **Cinsiyet:** Kadın, Erkek
 - **Calisma_Durumu:** İssiz, Calisan, Öğrenci, Emekli
 - **Evlilik_Durumu:** Evli, Bekar
 - **Egitim_Durumu:** İlkokul, Lise, OnLisans, Lisans, LisansUstu

2. BÖLÜM

12

Yapılar (Struct) ve Birlikler

struct (Yapı) Deyimi

13

- Yapılar, **farklı veri türlerine** sahip değişkenlerin bir **grup** olarak değerlendirilmesi ve bu grubun **bir isimle** kullanılması amacıyla tercih edilen **kullanıcı tanımlı bir veri türüdür**.
- Yapılar diziler gibi bellekte sürekli kalır.
- Bir yapı içerisindeki elemanlara **üye** (member) denir.
- **Üyelerin** her biri **farklı veri tipine** sahip olabilir.
- Yapılar sayesinde kendi veri tipinizi üretmeniz mümkündür.
- Yapılar farklı programlama dillerinde **Record (Kayıt)** olarak da geçerler.

struct Deyimi (devam...)

14

- Yapılar int, float, char gibi farklı tipteki değişkenleri veya aynı türdeki **birden çok** değişkeni bir araya getirmeyi sağlar.
- **Nesne tabanlı programlamanın** da **temelini** oluşturan bir yaklaşımdır.
- İlerleyen zamanlarda **C#** ve **Java** gibi tamamen nesneye dayalı programlamayı benimsemiş gelişmiş **dilleri öğrenmeye de yardımcı olacaktır.**

struct Deyimi (devam...)

15

- Yapı gösterimi aşağıdaki gibidir:

```
struct yapı_adı
```

```
{
```

```
    veri_tipi değişken_ismi1;
```

```
    veri_tipi değişken_ismi2;
```

```
    ...
```

```
};
```

Mantıksal Gruplama

struct Deyimi (devam...)

16

- **Örnek:** Bir üniversitedeki öğrencilerin bilgilerini programda kullanalım. Kullanılacak bilgiler aşağıdaki gibi olsun, nasıl tanımlarız?
 - Öğrenci Numarası
 - Öğrenci Adı
 - Öğrenci Soyadı
 - Öğrenci Cinsiyeti
 - Öğrenci Fakültesi/Bölümü
 - Öğrenci Genel Not Ortalaması
- `int` No;
- `char` Ad[50];
- `char` Soyad[50];
- `int` Cinsiyet;
- `int` FakulteBolum;
- `float` GenelOrtalama;

struct Deyimi (devam...)

17

Değişkenleri **tek tek tanımlamak** zorundayız.

Değişkenler arasında bir **bütünlük olmaz**.

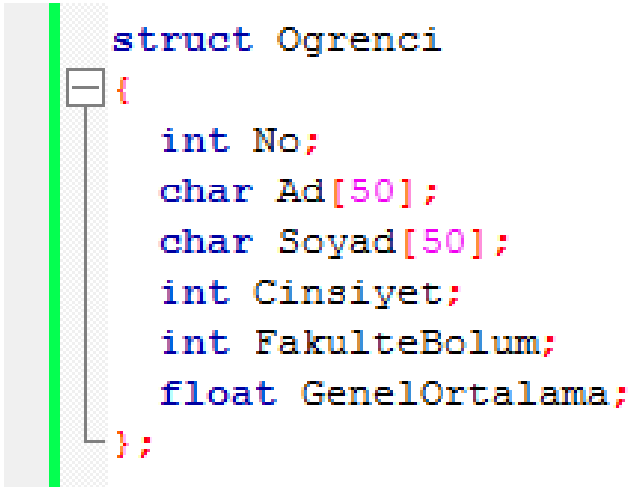
Bu bilgiler aynı öğrenciye mi ait yoksa başka bir öğrenciye mi **ayırt edemeyiz**.

Birden fazla öğrenci için nasıl bir tanım yapacağız?

struct Deyimi (devam...)

18

- **Örnek:** Bu öğrencilerin bilgilerini **struct** kullanarak ifade edelim.
- Kullanılacak bilgiler aşağıdaki gibi olsun:
 - Öğrenci Numarası
 - Öğrenci Adı
 - Öğrenci Soyadı
 - Öğrenci Cinsiyeti
 - Öğrenci Fakültesi/Bölümü
 - Öğrenci Genel Not Ortalaması



```
struct Ogrenci
{
    int No;
    char Ad[50];
    char Soyad[50];
    int Cinsiyet;
    int FakulteBolum;
    float GenelOrtalama;
};
```

struct Deyimi (devam...)

19

Oğrenci varlığı (**entity**) içerisinde
öğrenci ile ilgili bilgileri
kapsüllemiş (**encapsulation**) olduk.

struct Deyimi (devam...)

20

```
struct Ogrenci  
{  
    int No;  
    char Ad[50];  
    char Soyad[50];  
    int Cinsiyet;  
    int FakulteBolum;  
    float GenelOrtalama;  
};
```

```
struct Ogrenci ogrenci_bilgisi;  
  
ogrenci_bilgisi.No = 1122344;
```



Öğrenci tanımlama ve
değer atama

Örnek: struct Okuma ve Yazma

```
struct Ogrenci
{
    int No;
    char Ad[50];
    char Soyad[50];
    int Cinsiyet;
    int FakulteBolum;
    float GenelOrtalama;
};
```

```
int main()
```

```
{
    struct Ogrenci ogrenci_bilgisi1;
    printf("*****\n      VERI GIRISI\n*****\n");
    printf("Ogrenci No:");
    scanf("%d",&ogrenci_bilgisi1.No);
    printf("Ogrenci Ad:");
    scanf("%s",ogrenci_bilgisi1.Ad);
    printf("Ogrenci Soyad:");
    scanf("%s",ogrenci_bilgisi1.Soyad);
    printf("Ogrenci Cinsiyet:");
    scanf("%d",&ogrenci_bilgisi1.Cinsiyet);
    printf("Fakulte/Bolum:");
    scanf("%d",&ogrenci_bilgisi1.FakulteBolum);
    printf("Genel Ortalama:");
    scanf("%f",&ogrenci_bilgisi1.GenelOrtalama);
```

Okuma

```
printf("*****\n      VERI OKUMA\n*****\n");
printf(" No: %d \n Ad: %s \n Soyad: %s \n Cinsiyet: %d \n Fakulte-Bolum: %d \n Genel Ortalama: %0.2f",
    ogrenci_bilgisi1.No,
    ogrenci_bilgisi1.Ad,
    ogrenci_bilgisi1.Soyad,
    ogrenci_bilgisi1.Cinsiyet,
    ogrenci_bilgisi1.FakulteBolum,
    ogrenci_bilgisi1.GenelOrtalama
);
return 0;
```

Yazma

İYİ ÇALIŞMALAR...