BLM 112- Programlama Dilleri II

Hafta 8
Enum, Typedef ve Struct (Yapı)

Dr. Öğr. Üyesi Caner Özcan

Bilim doğanın dilini anlama çabasıdır. O dili anlayan doğayı dost, anlamayan düşman bilir.

Enum

- Değişkenin alabileceği değerlerin belli (sabit) olduğu durumlarda programı daha okunabilir hale getirmek için kullanılır.
- Enum'lar sabitler gibidir.
  - Enum değerleri otomatik olarak ayarlanır
  - Değerler 0'dan başlar ve 1'er artırılır.
  - Benzersiz sabit isimlerine ihtiyaç vardır.
- Biz enum ile kendi veri tipimizi oluşturabiliriz.
- Örneğin yeni bir boolean tipi oluşturabiliriz.
  - Bu boolean tipinde 0 false 1 ise true olabilir.

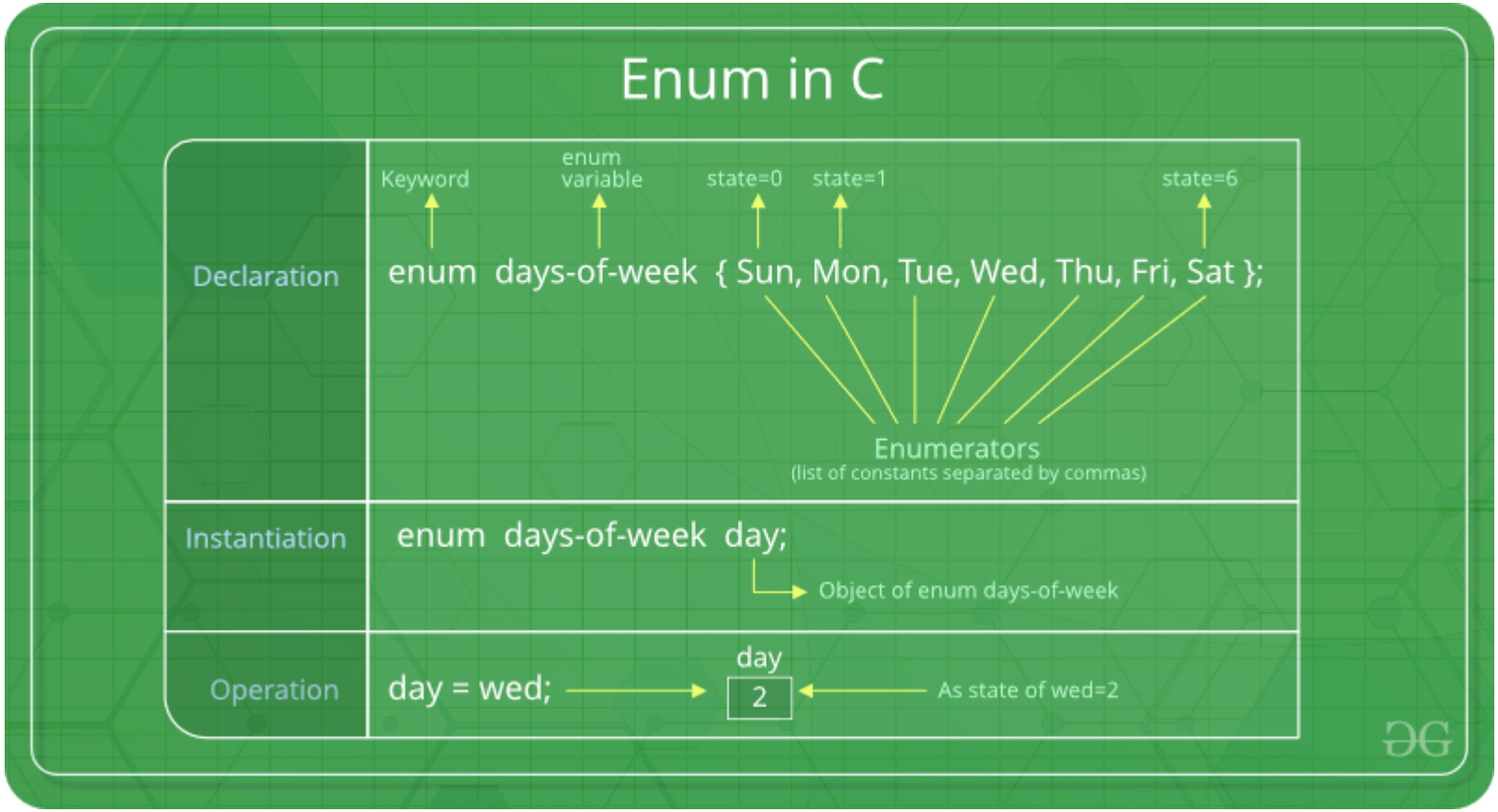
Enum

```
#include<stdio.h>
int main( void )
{
    // Define new data type boolean
    enum boolean {
            false = 0,
            true = 1
    };
    // Now define a variable with new data type boolen

    enum boolean isTrue;
    isTrue = true;
    if( isTrue == true )
            printf( "True\n" );

    return 0;
}
```

Enum


Enum in C
Declaration
Keyword
enum variable
state=0
state=1
state=6
enum days-of-week { Sun, Mon, Tue, Wed, Thu, Fri, Sat };
Enumerators
(list of constants separated by commas)
Instantiation
enum days-of-week day;
Object of enum days-of-week
Operation
day = wed;
day
2
As state of wed=2

Enum

```c
#include<stdio.h>
int main( void )
{
    // Define new data type mainColors
    enum mainColors {
            Red,
            Blue,
            Yellow
    };

    // Define variable
    enum mainColor pixel;

    // Set value of pixel to blue
    // You can set Yellow or Red also.
     pixel = Blue;

    // Compare variable's value.
    if( pixel == Red )
            printf( "Red pixel \n" );
    else if( pixel == Blue )
            printf( "Blue pixel \n" );
    else
            printf( "Yellow pixel\n" );

    return 0;
}
```

Enum

```
// An example program to demonstrate working of enum in C
#include<stdio.h>

enum week{Mon, Tue, Wed, Thur, Fri, Sat, Sun};

int main()
{
        enum week day;
        day = Wed;
        printf("%d",day);
        return 0;
}
```

Enum

Two enum names can have same value. For example, in the following C program both ‘Failed’ and ‘Freezed’ have same value 0.

```
#include <stdio.h>
enum State {Working = 1, Failed = 0, Freezed = 0};

int main()
{
printf("%d, %d, %d", Working, Failed, Freezed);
return 0;
}
```

Enum

```
#include <stdio.h>
enum day {sunday = 1, monday, tuesday = 5,
        wednesday, thursday = 10, friday, saturday};

int main()
{
   printf("%d %d %d %d %d %d %d", sunday, monday,
tuesday,
          wednesday, thursday, friday, saturday);
   return 0;
}
```

Output: 1 2 5 6 10 11 12

Enum

- Gruplanması gereken bir veri kümeniz var ise Enum kullanışlı olabilir. Örneğin:
  - enum egitim { ilkokul, ortaokul, lise, lisans };
  - enum egitim ogrenci;
  - enum cinsiyet { kadin, erkek };
  - enum cinsiyet kisi;
- Her bir değişken tanımlaması için enum deyimi kullanılmaktadır.
- Her değişken tanımlamasında enum yazmamanın iki yolu var.
  - Değişkeni enum tanımlaması ile yazmak
  - Typedef kullanmak

Enum

```
#include<stdio.h>
int main( void )
{
    // Define new data type
    // Also define a new variable with the new data type,

    enum boolean {
            false = 0,
            true = 1
    } isTrue;

    isTrue = true;
    if( isTrue == true )
            printf( "True \n" );
    return 0;
}
```

Typedef

- Veri tiplerini kullanıcı tanımlı adlar ile isimlendirmek için kullanılır.
- Typedef kullanım formatı:
  - typedef *eski_veri tipi yeni_veri_tipi*
  - typedef int tamsayi
    - int tipini tamsayi ismi ile tanımlar

Typedef

```
#include<stdio.h>
int main( void )
{
    // Define new data type
    // Also define a new variable with the new data type,
    enum boolean {
            false = 0,
            true = 1
    };
    // With this definition we can create boolean type variables with
    //one step
    typedef enum boolean bool;

    bool isTrue;

    isTrue = true;
    if( isTrue == true )
            printf( "True \n" );
    return 0;
}
```

Enum

- Eğer enum global olarak tanımlanırsa bir fonksiyona parametre olarak kullanılabilir.

```
#include<stdio.h>
// We create month list. Starting from 1 for January, months take
//numerical values.
enum month_list {
    january = 1, february, march, april,
    may, june, july, august,
    september, october, november, december
};
// Using typedef to make variable definitions easy. We will just
// type month to create variable
typedef enum month_list months;

void writeMonthName ( months );
int main( void )
{
    // Create a variable with months data type and assign value as
   //  november.
    Months thisMonth = november;
    // november is actually 11 in numerical representation.
    printf( "Month- %d is: ", thisMonth);
    // call function.
    writeMonthName( thisMonth );
    return 0;
}
```

Enum

```
void writeMonthName( months nameOfMonth )
{
    switch( nameOfMonth ) {
            case january: printf( "January\n" );break;
            case february: printf( "February\n" );break;
            case march: printf( "March\n" );break;
            case april: printf( "April\n" );break;
            case may: printf( "May\n" );break;
            case june: printf( "June\n" );break;
            case july: printf( "July\n" );break;
            case august: printf( "August\n" );break;
            case september: printf( "September\n" );break;
            case october: printf( "October\n" );break;
            case november: printf( "November\n" );break;
            case december: printf( "December\n" );break;
    }
}
```

Yapılar (Struct)

- Farklı tipte değişkenleri tek bir yapı altında gruplamak için kullanılır.
- Yapılar, nesne yönelimli programlama için önemlidir.

```
#include<stdio.h>
int main( void )
{
    struct {
            int year;
            int month;
            int day;
    } birth_day;
    printf( "Enter your birth day " );
    printf( " in MM-DD-YYYY format> ");
    scanf( "%d-%d-%d",    &birth_day .month,
                          &birth_day .day,
                          &birth_day .year );
    printf( "Your birth day: " );
    printf( "%d/%d/%d\n", birth_day.month,
                          birth_day.day,
                          birth_day.year );
    return 0;
}
```

Yapılar (Struct)

- Eğer bu örnekte yapı kullanılmasaydı 9 farklı değişken tanımlamak zorunda kalacaktık.
- Yapı kullanarak 3 değişken yeterli.
- Bir kişinin 20 farklı bilgisi üzerinde işlem yapan bir program düşünün.
- 3 kişi için 60 farklı değişken tanımlamalısınız.
- Yapıların bir başka avantajı kolaylıkla kveriyi bir değişkenden opyalayabilmek.
- you = yourSister ataması yourSister'ın verisini you üzerine kopyalar.

```
#include<stdio.h>
int main( void )
{
    struct {
            int year;
            int month;
            int day;
    } you, yourSister, yourBrother;
    printf( "Enter your birth day " );
    printf( " in MM-DD-YYYY format> ");
    scanf( "%d-%d-%d",    &you.month,
                          &you.day,
                          &you.year );
    printf( "Enter your sisters birthday> " );
    scanf( "%d-%d-%d",    &yourSister.month,
                          &yourSister.day,
                          &yourSister.year );
    printf( "Enter your brothers birthday> " );
    scanf( "%d-%d-%d",    &yourBrother.month,
                          &yourBrother.day,
                          &yourBrother.year );
    return 0;
}
```

İç İçe Yapılar

```
#include<stdio.h>
int main( void )
{
    struct {
            char name[40];
            int lenght;
            struct {
                    int year;
                    int month;
                    int day;
            } bornInformation;
    } person;
    printf( "Your name: " );
    scanf( "%s", person.name );
    printf( "Your lenght: " );
    scanf( "%d", &person.lenght );
    printf( "Your birth day: ");
    scanf( "%d-%d-%d",    &person.bornInformation.month,
                          &person.bornInformation.day,
                          &person.bornInformation.year );
    printf( "Entered information:\n" );
    printf( "Name: %s\n", person.name );
    printf( "Lenght: %d\n", person.lenght );
    printf( "Birth day: %d/%d/%d\n",     person.bornInformation.month,
                                         person.bornInformation.day,
                                         person.bornInformation.year );
    return 0;
}
```

Yapıları Etiketleme

- Yapıları etiketlemenin birçok avantajı vardır.

- Eğer etiketleme yapılmaz ise değişkenleri de yapıyı tanımlarken tanımlamak zorunda kalırsınız.

- Eğer etiket kullanırsanız programın her hangi bir yerinde yapıdan dilediğiniz kadar değişken tanımlayabilirsiniz.

- Yapıyı kullanabilmemiz için tanımladığımız etiket ile bir değişken oluşturmalıyız.

```
#include<stdio.h>
#include<string.h>
int main( void )
{
    // personData is the label of our struct
    struct person_Data {
            char name[40];
            int length;
    };

    // We create two variables using struct.
    struct person_Data person_1;
    struct person_Data person_2;

    // We store the first person's data.
    strcpy( person_1.name, "AHMET" );
    person_1.length = 170;

    // We store the second person's data.
    strcpy( person_2.name, "MEHMET" );
    person_2.lenght = 176;

    return 0;
}
```

Yapılar İçin Başlangıç Değeri

- Yapılar değişkenlerinin başlangıç değerleri ile tanımlanabilir.
- Değerlerin sırası yapıdaki sırayla olmalıdır.
- Etiket ile tanımlanmış veya tanımlanmamış yapılar için başlangıç değeri verebilirsiniz.

```
#include<stdio.h>
int main( void )
{
    struct {
            char name[40];
            int lenght;
    } person = { "Ali", 167 };

    return 0;
}
```

```
#include<stdio.h>
int main( void )
{
    struct person_Data {
            char name[40];
            int lenght;
    };

    struct person_Data person = { "Ali", 167 };

    return 0;
}
```

Yapı Dizileri

```c
#include<stdio.h>
int main( void )
{
    int i;

    struct birthDate {
            int day;
            int month;
            int year;
    };

    struct person_data {
            char name[40];
            int lenght;
            //Define a variable of an other structure type inside
            //struct
            struct birthDate date;
    };

    struct person_data person[3] = { "Ali", 170, { 17, 2, 1976 },
                                     "Veli", 178, { 14, 4, 1980 },
                                     "Cenk", 176, { 4, 11, 1983 } };

    // Print all values of people defined in array
    for( i = 0; i < 3; i++ ) {
            printf( "Record No.: %d\n", ( i + 1 ) );
            printf( "Name: %s\n", person[i].name );
            printf( "Length: %d\n", person[i].lenght );
            printf( "Birth Date: %d/%d/%d\n\n", person[i].date.day,
                                                person[i].date.month,
                                                person[i].date.year );
    }

    return 0;
}
```

Yapılara Pointer ile Erişmek

```
#include<stdio.h>
int main( void )
{
    int i;

    struct birthDate {
            int day;
            int month;
            int year;
    };

    struct person_data {
            char name[40];
            int lenght;
            //Define a variable of an other structure type inside
            //struct
            struct birthDate date;
    };

   struct person_data *ptr;

   struct person_data person[3] = { "Ali", 170, { 17, 2, 1976 },
                                    "Veli", 178, { 14, 4, 1980 },
                                    "Cenk", 176, { 4, 11, 1983 } };

    //Print all values of people defined in array
    for( i = 0, ptr = &person[0]; ptr <= &person[2]; ptr++, i++ ) {
            printf( "Record No.: %d\n", ( i + 1 ) );
            printf( "Name: %s\n", ptr->name );
            printf( "Length: %d\n", ptr->lenght );
            printf( "Birth day: %d/%d/%d\n\n", ptr->date.day,
                                               ptr->date.month,
                                               ptr->date.year );
    }

    return 0;
}
```

Yapıları Fonksiyonlara Parametre Olarak Gönderme

- Yapıyı global olarak tanımla ve fonksiyona gönderme.

```
#include<stdio.h>
#include<string.h>
struct person_data {
    char name[40];
    int length;
};

struct person_data getPersonData( void );
void showPersonData( struct person_data );

int main( void )
{
    struct person_data person;
    person = getPersonData( );
    showPersonData( person );

    return 0;
}
struct person_data getPersonData( void )
{
    struct person_data person;
    printf( "Name> " );
    gets( person.name );
    printf( "Length> " );
    scanf( "%d", &person.length );
    return person;
}
void showPersonData( struct person_data person )
{
    printf( "Name: %s\n", person.name );
    printf( "Length: %d\n", person.length );
}
```

Gelecek Hafta

- Tek Bağlı Doğrusal Listeler

Kaynaklar

- Doç. Dr. Fahri Vatansever, “Algoritma Geliştirme ve Programlamaya Giriş”, Seçkin Yayıncılık, 12. Baskı, 2015.
- Kaan Aslan, “A’dan Z’ye C Klavuzu 8. Basım”, Pusula Yayıncılık, 2002.
- Paul J. Deitel, “C How to Program”, Harvey Deitel.
- “A book on C”, All Kelley, İra Pohl

S o r u l a r ?

Dinlediğiniz için teşekkürler

CANER ÖZCAN
canerozcan@karabuk.edu.tr

Dip not..

Türkiye Cumhuriyetinin, özellikle bugünkü gençliğine ve yetişmekte olan çocuklarına hitap ediyorum: Batı senden, Türk'ten çok geriydi. Manada, fikirde, tarihte bu böyleydi. Eğer bugün batı teknikte bir üstünlük gösteriyorsa, ey Türk Çocuğu, o kabahat da senin değil, senden öncekilerin affedilmez ihmalinin bir sonucudur. **Şunu da söyleyeyim ki, çok zekisin!.. Bu belli. Fakat zekânı unut!.. Daima çalışkan ol…**

Gazi Mustafa Kemâl ATATÜRK