Veri Yapıları ve Algoritmalar

Diziler, Fonksiyonlar,
Struct, Union ve Bit Maskeleme

Dr. Öğr. Üyesi M. Ozan AKI

30.10.2020

C Dili Anahtar Sözcükleri

Anahtar sözcükler, değişken, fonksiyon adı olarak kullanılamazlar.

auto, break, case, char, const, continue, default, do, double, else, enum, extern, float, for, goto, if, int, long, register, return, short, signed, sizeof, static, struct, switch, typedef, union, unsigned, void, volatile, while

Fonsiyonlar

Fonksiyonlara gönderiler sabit değer ya da değişkenlere **parametre** adı verilir.

Fonksiyonlardan elde edilen sonuç ise **dönüş değeri** olarak adlandırılır.

Bir fonksiyon,

- Hiçbir parametre almayıp, Hiçbir değer döndürmüyor olabilir.
- Sadece parametre alıp, Değer döndürmüyor olabilir
- Hiçbir parametre almayıp, Değer döndürüyor olabilir.

Fonksiyonlar, birden fazla parametre alabilir ancak, sadece bir değer döndürebilirler. (Birden fazla değer gerekirse ne olacak?)

Fonsiyonlar

Fonksiyonlar, dönüş adreslerini **Yığın (Stack)** Bellekte saklar.
Yığın bellek, en son girenin ilk çıktığı (LIFO) bir bellek türüdür.
PUSH, POP

Ancak Fonksiyonlar Parametre alıyor ya da Değer Döndürüyor ise, Bu değerlerde yine Yığın Bellek üzerine kaydedilerek fonksiyon çağırılır.

İç içe çağırılan fonksiyonlarda Yığın bellek kümülatif olarak dolar.

Bu nedenle, özellikle Recursive (Yinelemeli – kendi kendini çağıran) fonksiyonlar Yığın bellek taşmalarına (Stack Overflow) neden olabilir.

Karakter Dizileri (String)

Karakter dizileri, bellekte ardaşık olarak dizilmiş anlamlı karakterler bütünüdür.

Programlamada anlamlı olan karakter dizisi uzunlu açıkça belirtilmiş olmalıdır. Bunun için, farklı diller farklı yaklaşımlar getirmişlerdir;

Pascal / Delphi dillerinde karakter dizilerinin uzunluğu ilk karakerde saklanır.
isim[] = [19, T, r, a, k, y, a, ,U, n, i, v, e, r, s, i, t, e, s, i]

C / C++ dillerinde ise karakter dizisi NULL, \0 ile bitmek zorundadır.
isim[] = [T, r, a, k, y, a, ,U, n, i, v, e, r, s, i, t, e, s, i, \0]

Diziler

Belleğin ardaşık adreslerinde aynı veri tipindeki değişkenlerin saklanması yoluyla diziler elde edilir.

Diziler, birçok veri modeli için uygun bir veri yapısı sunar. Böylelikle liste, yığın, graf, ağaç gibi yapılar diziler üzerinde kolayca uygulanabilir.

C dilinde diziler, doğrudan değişken bildirimi sırasında tanımlanır;

```
int boy[20];
float kilo[25];
char *ad[40];
char harf[29];
float yillik_oran[12][31];
int xyz[10][20][30];
```

Diziler

Öntanımlı veri tipleri dışında, struct yapıları kullanılarak özel dizilerde kolayca oluşturulabilir;

```
struct tarih {
        unsigned char gun;
        unsigned char ay;
        short int yil;
}

struct tarih dogum_gunleri[2000];
```

Structure & Union

Structure yapısı içerisinde C dilinin ön tanımlı veri tipleri kullanıldığı gibi, bit sayıları özelleştirilerek özel yapılar oluşturulabilir. Bu, özellikle çeşitli dosya formatlarının okunup yazılmasında oldukça kullanışlıdır.
Bunun yanında Structure ve Union birlikte kullanılarak değişkenlere bitsel düzeyde erişim de sağlanabilir;

Örn; IEEE 754 Sayı Formatının Bitlerini struct ve union ile ayırabiliriz

```
union {
        float f;
        struct {
                unsigned carpan:23;
                unsigned us:8;
                unsigned isaret:1;
        } bit;
} ortak;
```

Structure & Union

Örneğin 8 bitlik bir sayıyı bitlerine ayırabilir, isteğimiz bir biti okuyabilir ya da değiştirebiliriz.

```
union {
        unsigned char butun;
        struct {
                unsigned b0:1;
                unsigned b1:1;
                unsigned b2:1;
                unsigned b3:1;
                unsigned b4:1;
                unsigned b5:1;
                unsigned b6:1;
                unsigned b7:1;
        } bit;
} sayi;
```

Structure & Union

Daha güncel uygulamalar için, örneğin personel kaydı tutan bir programda her bir personel kaydı struct olarak tanımlanabilir.

```
struct tarih {
        unsigned char gun;
        unsigned char ay;
        unsigned short yil;
};
struct  personel {
        char tc_no[11];
        char ad[20];
        char soyad[20];
        tarih dogum_tar;
        tarih ise_baslama;
        int maas_carpan;
} ;
```

Sizde okul numaranızı struct ile çözümleyerek birinci öğretim, ikinci öğretim, kayıt yılı, okul kodu, bölüm ve sıra numarası bilgilerini çıkarınız.

Operatörler ile Bit Maskeleme

Bir abc integer değişkenin 3. biti 1 ise;
if(abc & 0x0008) { ... }

Bir abc integer değişkenin 6. biti 0 ise;
if((abc & 0x0040)==0) { ... }

Bir abc integer sayının 2. bitini 1 yapalım;

abc |= 0x0004; *// (abc = abc | 0x0004)*

Bir abc integer değişkenin 4. bitini 0 yapalım;

abc &= ~0x0010; *// abc = abc & (~0x0010)*

Operatörler ile Bit Maskeleme

Bir abc integer değişkenin 3,4,5. bitlerini başka bir değişkene alın;
baska = (abc & 0x0038) >> 3;

Bir abc integer değişkenin 6,7,8. bitlerin yerine 5 sayısını yazalım;
abc = (abc & ~(0x01C0)) | (5<<5);

16 bit integer bir değişkenin üst 8 biti ile alt 8 bitini yer değiştirin;

yeni = (abc & 0x00ff)<<8 | (abc & 0xff00)>>8;

integer bir sayının 1., 4., 7. bitleri aynı anda 1 olduğunda doğru (true) döndüren ifadeyi yazınız

if(abc&0x0092) { ... }

Operatörler ile Bit Maskeleme

Bir sayının bitlerini en sağdaki bit en solda, en soldaki bit en sağda olacak şekilde tersine çeviren programı yazınız

```
unsigned char i, sayi, yeni=0;

sayi=??;
int bitsay = 8*sizeof(sayi);

for(i=0; i<bitsay; i++)
{
        if(sayi&(1<<i)) yeni|=1<<(bitsay-i-1);
}

printf("yeni= %u\n", yeni);
```