Diziler

Ders notları
(1) C How to Program 8th Edition, by Deitel & Deitel
(2) Bilgisayar Programlama III notları – (Dr. Süha Tuna)

İÇİNDEKİLER

- Diziler
- Dizi tanım
- Dizi elemanlarına referans verme
- Dizilerin fonksiyonlara geçişi

Diziler (arrays)

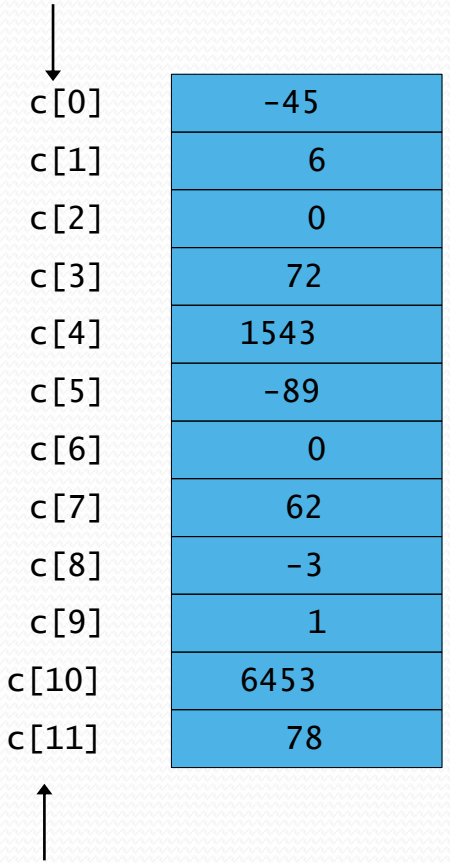
- Aynı tipte tekrar eden değişkenlerin bir grup ismi altında toplanması
- Tek boyutlu diziler
  - Dizi tip tanımı
    - type array_name[3];
    - int ders_not[3];
  - Diziye değer atama
    - dersNot[1]=90;
- Dizi indisleri o'dan başlar.

Dizinin ismi. Tüm elemanlara dizinin ismi ile erişilir.

Diziler

- Array
  - Ardışıl bellek adreslerinden oluşur.
  - Aynı isimli ve aynı türden elemanlar içerir
- Bir elemana referans etmek için
  - Array ismi
  - Pozisyon numarası
- Format:

  arrayname`[` *position number* `]`
  - İlk eleman pozisyon `0'dadır.`
  - `C isimli en elemanlı bir dizi tanımı:`
    - `c[ 0 ], c[ 1 ]...c[ n - 1 ]`



İlk değer atamaları

- İlk değer atama (initialize)

```
int n[ 5 ] = { 1, 2, 3, 4, 5 };
```

  - Eğer ilk değer sayısı eleman sayısından azsa kalanlar 0 ile doldurulur.

```
int n[ 5 ] = { 0 }
```

    - Tüm elemanlara 0 atanır.
  - Eğer eleman fazlaysa yazım hatası oluşur
  - C dizilerde sınır değer kontrolü yoktur.
  - Boyut boş bırakılırsa atanan elemanlar boyutu tanımlar.

```
int n[ ] = { 1, 2, 3, 4, 5 };
```

  - 5 ilk değer, 5 elemanlı dizi

Karakter Dizileri

- Metin "`first`" gerçekte static karakter dizisidir.
  - Karakter dizileri string sabitler kullanılarak ilk değerlenebilir.

```
char string1[] = "first";
```

    - Null karakter `'\0'` stringleri sonlandırır.
    - `string1` gerçekte 6 elemana sahiptir
      - Aşağıdaki yazım şekline eşittir.

```
char string1[] = { 'f', 'i', 'r', 's', 't', '\0' };
```

  - Özel bir karakterine erişilebilir.

```
string1[ 3 ] is character 's'
```

  - Array ismi dizi adresini verir, bu durumda scanf fonksiyonunda & kullanmak gerekmez.

```
scanf( "%s", string2 );
```

    - Boşluk gelene kadar gelen karakterler okunur.
    - Dizi sınırlarından taşma olabilir ! Dikkatli kullanınız.

İlk değer atamaları

- char s[] = "hello";
- Yukarıdaki atama çalışır ancak aşağıdaki atama çalışmaz çünkü 1. satırda programcı tarafından yapılmış bir pointer ataması yoktur. Derleyici dahili olarak bir pointer atamıştır. İkinci satırdaki atama iki diziyi birbirine atama anlamındadır ve gerçekleştirilemez.
- char s[100];
- s = "hello";

Metin dizilere ilk değer atamada ayrıntılar

- String (metin) son karakteri ‘\0’ olan bir char dizi ile ifade edilir.
- char message[] = "Hello"; // ‘\0’ (null) karakteri otomatik eklenir.
- Yukarıdaki atama aşağıdaki atama ile aynı anlamdadır.
- char message[] = {'H', 'e', 'l', 'l', 'o', '\0'};
- Array uzunluğu metin uzunluğu +1’dir. +1 değeri.

Dizileri Fonksiyonlara geçmek

- Fonksiyon çağrılırken dizi ismi [] kullanmadan eklenir.

```
int myArray[24];
myFunction(myArray, 24);
```

  - Genellikle array size' da fonksiyona gönderilir.
  - Dizi geçişi call-by-reference şeklindedir.
  - array ismi ilk elemanın adresidir.
  - Functionun dizinin saklandığı hafıza alanını bilmesini sağlar.
  - Böylece fonksiyonlar diziyi orijinal yerinde değiştirir.
- Array eleman geçişi
  - Eleman geçişi call-by-value şeklindedir.
  - Örnek(i.e., `myArray[3]`)

Çok boyutlu diziler

- int dersNot[][];
- Örnek ilk sene aldığınız ders notlarını toplatan bir proje yazınız.

Dizi Eleman Sayısını Bulmak

- Dizi büyüklüğünü bir elemanın büyüklüğüne bölerek gerçekleştirilir.
- Büyüklük sizeof fonksiyonu ile bulunur.

```
int str_numbers[5] = {1,4,8,2,9};
int total = sizeof(str_numbers)/sizeof(str_numbers[0]);
printf ("%d\n", total);
return 0;
```

Eleman sayısı tanımı için bir makro

- Eleman sayısı makro ile de tanımlanabilir.

```
#define ARRAY_LENGTH     (25)
for (size_t i=0; i<ARRAY_LENGTH; i++) {
   // Do something with array[i]
}
```

Dizi ve sınırlar

- C dizi sınırlarını doğrudan tanımlamaz. Bellekteki adreslere erişmek için dizi değişkenini kullanır.

```
int a[10];
a[3]=4;
a[11]=3;//does not give segmentation fault
a[25]=4;//does not give segmentation fault
a[20000]=3; //gives segmentation fault
return 0;
```

- C ve C++ atama ve değer alma işlemlerinde bir kontrol yapmaz.
- Bellek aşılırsa dizi İşletim Sistemi özelliklerine göre hata verir veya vermez.
- Kontrol için özel sonlanma karakterleri haricen eklenebilir. Örneğin karakterler için '\0'

Örnek: Karakterleri enter girilene kadar diziye atan bir uygulama yazınız. (Farklı bir if kullanımı)

```
main( ) {
        int n, c;
        char line[100];
        n = 0;
        while( (c=getchar( )) != '\n' ) {
                if( n < 100 )
                        line[n] = c;
                n++;
        }
        printf("length = %d\n", n);
}
```

Metin dizilerde atama şekli ve sınırlar

- char message[] = "Merhaba"; // şeklindeki bir atamada sona bir ‘\0’ karakteri eklenir.
- Dizi uzunluğu **sizeof(message)/sizeof(message[0])** şeklinde hesaplanırsa metin +1 (örnekte 8) çıkar.
- Eğer atama
- char message[] = {‘M’,’e’,’r’,’h’,’a’,’b’,’a’}; // şeklindeki yapılırsa sona bir ‘\o’ karakteri eklenmez. Bu durumda dizi uzunluğu 7 çıkar

Metin dizilerde atama şekli ve sınırlar

Metin atama ile oluşturulan diziler while(message[i]) ile son elemana kadar dönen döngüler oluşturulabilir. Döngü '\0' ile karşılaşınca sonlanır.

Örneğin aşağıdaki atamada while 3 kere döner. Çünkü sona otomatik '\0' atanmıştır.

```
char message[] = "abc";      //
int i=0;
while(message[i]){
        printf("İndis%d", i);
        i++;
}
```

Metin dizilerde atama şekli ve sınırlar

Aşağıdaki atamada while belirsiz sayıda döner. Çünkü '\0' atanmamıştır. Bellekte '\0' karakterine karşılaşıncaya kadar döner.

```
char message[] = {'a','b','c'};      //
int i=0;
while(message[i]){
        printf("indis:  %d", i);
        i++;
}
```

Kaynaklar

- *McConnell Steve, Code Complete, Second Edition,2002 MS press*
- **C ve C++ (Deitel & Deitel)**