

İskenderun Teknik Üniversitesi



Fakülte: Mühendislik ve Doğa Bilimleri *Fakültesi*

Bölüm: *Bilgisayar Mühendisliği Bölümü*

Ders: Algoritmalar ve Programlama

Dönem: 2021 – 2022 (Güz)

Öğretim üyesi: *Dr. Öğr. Üyesi Mehmet SARIGÜL*

12.

Hafta

- Program süresince statik bir yerel değişken mevcuttur, ancak yalnızca işlev gövdesinde görülebilir.
- Statik kelimesini bir yerel dizi tanımına uygulayabiliriz, böylece işlev her çağrıldığında dizi oluşturulmaz ve başlatılmaz ve işlev programdan her çıkıldığında dizi yok edilmez.
- Bu, özellikle büyük diziler içeren ve sık adı verilen işlevlere sahip programlar için program yürütme süresini azaltır.

Statik Yerel Diziler ve Otomatik Yerel Diziler

- StaticArrayInit işlevi iki kez çağrılır.
- Fonksiyondaki yerel statik dizi, program başlatılmadan önce sıfır olarak başlatılır.
- İşlev diziyi yazdırır, her öğeye 5 ekler ve diziyi yeniden yazdırır.
- İşlev ikinci kez çağrıldığında, statik dizi ilk işlev çağrısı sırasında depolanan değerleri içerir.

Statik Yerel Diziler ve Otomatik Yerel Diziler




```
#include<stdio.h>

void staticArrayInit(void);
void automaticArrayInit(void);

int main(void)
{
    printf("First call to each function:\n");
    staticArrayInit();
    automaticArrayInit();

    printf("Second call to each function:\n");
    staticArrayInit();
    automaticArrayInit();


    return 0;
}
```



```
void staticArrayInit(void)
{
    static int array1[3];

    printf("\nValues on entering staticArrayInit:\n");

    for (int i = 0; i < 3; i++)
    {
        printf("array1[%d]=%d ", i, array1[i]);
    }
    printf("\nValues on exiting staticArrayInit:\n");
    for (int i = 0; i < 3; i++)
    {
        printf("array1[%d]=%d ", i, array1[i]+=5);
    }
}
```



```
void automaticArrayInit(void)
{
    int array2[3] = {1,2,3};

    printf("\nValues on entering automaticArrayInit:\n");

    for (int i = 0; i < 3; i++)
    {
        printf("array2[%d]=%d ", i, array2[i]);
    }
    printf("\nValues on exiting automaticArrayInit:\n");
    for (int i = 0; i < 3; i++)
    {
        printf("array2[%d]=%d ", i, array2[i] += 5);
    }
}
```


First call to each function:

Values on entering staticArrayInit:

array1[0]=0 array1[1]=0 array1[2]=0

Values on exiting staticArrayInit:

array1[0]=5 array1[1]=5 array1[2]=5

Values on entering
automaticArrayInit:

array2[0]=1 array2[1]=2 array2[2]=3

Values on exiting
automaticArrayInit:

array2[0]=6 array2[1]=7 array2[2]=8

Second call to each function:

Values on entering staticArrayInit:

array1[0]=5 array1[1]=5 array1[2]=5

Values on exiting staticArrayInit:

array1[0]=10 array1[1]=10

array1[2]=10

Values on entering automaticArrayInit:

array2[0]=1 array2[1]=2 array2[2]=3

Values on exiting automaticArrayInit:

array2[0]=6 array2[1]=7 array2[2]=8

- Statik olan diziler program başlangıcında bir kez başlatılır.
- Statik bir diziyi açık bir şekilde başlatmazsanız, bu dizinin öğeleri varsayılan olarak sıfıra başlatılır.
- Daha sonra, yerel bir statik dizi ile `staticArrayInit` işlevini ve yerel bir otomatik dizi ile `automaticArrayInit` işlevini gösterir.

Statik Yerel Diziler ve Otomatik Yerel Diziler

- `automaticArrayInit` işlevi de iki kez çağrılır.
- Fonksiyondaki otomatik yerel dizinin elemanları 1, 2 ve 3 değerleriyle başlatılır.
- İşlev diziyi yazdırır, her öğeye 5 ekler ve diziyi yeniden yazdırır.
- İşlev ikinci kez çağrıldığında, dizi öğeleri tekrar 1, 2 ve 3 olarak başlatılır çünkü dizi otomatik depolamaya sahiptir.

Statik Yerel Diziler ve Otomatik Yerel Diziler

- Bir dizi bağımsız değişkenini bir işleve geçirmek için, dizinin adını köşeli parantez olmadan belirtin.
- Örneğin, hourlyTemperatures dizisi şu şekilde tanımlandıysa
- `int hourlyTemperatures [24];`
- işlem çağırısı
- `modifyArray (hourlyTemperatures, 24);`
- diziyi hourlyTemperatures ve boyutunu modifyeArray işlevine geçirir.

Dizileri İşlevlere Aktarma

- C, dizileri referans ile işlevlere otomatik olarak iletir - çağrılan işlevler, arayanların orijinal dizilerindeki öge değerlerini değiştirebilir.
- Dizinin adı, dizinin ilk elemanının adresi olarak değerlendirilir.
- Dizinin başlangıç adresi aktarıldığı için, çağrılan işlev dizinin tam olarak nerede saklandığını bilir.
- Bu nedenle, çağrılan işlev, işlev gövdesindeki dizi öğelerini değiştirdiğinde, dizinin asıl öğelerini orijinal bellek konumlarında değiştirir.

Dizileri İşlevlere Aktarma

- Dizilerin tamamı referans ile aktarılsa da, tek tek dizi öğeleri tam olarak basit değişkenler gibi değere göre aktarılır.
- Bir dizinin bir öğesini bir işleve geçirmek için, işlev çağrısında bir argüman olarak dizi öğesinin dizinlenmiş adını kullanın.

Dizileri İşlevlere Aktarma

- Bir işlevin işlev çağrısı aracılığıyla bir dizi alması için, işlevin parametre listesinin bir dizinin alınacağını belirtmesi gerekir.
- Örneğin, modifyArray işlevinin işlev başlığı şu şekilde yazılabilir:
 - `void modifyArray (int b[], int boyut)`
- modifyArray parametresinin b parametresinde bir tamsayı dizisi ve parametre boyutunda dizi öğelerinin sayısını almayı beklediğini belirtir.
- Dizi parantezleri arasında dizinin boyutu gerekli değildir.


```
#include <stdio.h>
void modifyArray(int b[], int size)
{
    for (int j = 0; j < size; ++j)
        b[j] *= 2;
}
void modifyElement(int e)
{
    printf("Value in modifyElement is %d\n", e *= 2);
}
int main(void)
{
    int a[5] = { 0, 1, 2, 3, 4 };
    printf("The values of the original array are:");
    for (int i = 0; i < 5; i++)
        printf("%3d", a[i]);
    modifyArray(a, 5);
    printf("\nThe values of the modified array are:");
    for (int i = 0; i < 5; i++)
        printf("%3d", a[i]);
    printf("\nThe value of a[3] is %d\n", a[3]);
    modifyElement(a[3]);
    printf("The value of a[3] is %d\n", a[3]);
}
```



```
1 2 3 4 5
2 4 6 8 10
Modified element : 16
8
```

- C'deki dizilerin birden çok indisi olabilir.
- Çok boyutlu dizilerin yaygın bir kullanımı, satırlar ve sütunlar halinde düzenlenmiş bilgilerden oluşan değer tablolarını temsil etmektir.
- Belirli bir tablo ögesini tanımlamak için iki dizin belirtmeliyiz: Birincisi (geleneksel olarak) ögenin satırını ve ikincisi (geleneksel olarak) ögenin sütununu tanımlar.
- Çok boyutlu dizilerin ikiden fazla indisi olabilir.
- Belirli bir ögeyi tanımlamak için iki indis gerektiren tablolar veya diziler, iki boyutlu diziler olarak adlandırılır.

Çok boyutlu diziler

	Column 0	Column 1	Column 2	Column 3
Row 0	a[0][0]	a[0][1]	a[0][2]	a[0][3]
Row 1	a[1][0]	a[1][1]	a[1][2]	a[1][3]
Row 2	a[2][0]	a[2][1]	a[2][2]	a[2][3]

Column index
 Row index
 Array name

Çok boyutlu diziler

- Şekil iki boyutlu bir diziyi göstermektedir, a.
- Dizi üç satır ve dört sütun içerir, bu nedenle 3'e 4'lük bir dizi olduğu söylenir.
- Genel olarak, m satırlı ve n sütunlu bir dizi m'e n dizi olarak adlandırılır

Çok boyutlu diziler

- A dizisindeki her eleman, Figure'de $a[i][j]$ biçimindeki bir eleman adıyla tanımlanır; a , dizinin adıdır ve i ve j , a 'daki her bir öğeyi benzersiz şekilde tanımlayan indislerdir.
- 0 satırındaki öğelerin adlarının hepsinin ilk indeksi 0'dır; 3. sütundaki öğelerin adlarının hepsinin ikinci dizini 3'tür.

Çok boyutlu diziler

- Çok boyutlu bir dizi, tek boyutlu bir dizi gibi, tanımlandığında başlatılabilir.
- Örneğin, iki boyutlu bir `int b [2][2]` dizisi tanımlanabilir ve şu şekilde başlatılabilir:
- `int b[2][2] = { {1, 2}, {3, 4} };`
- Değerler, kaşlı ayraçlar içinde satırlara göre gruplandırılmıştır.
- İlk küme ayracı kümesindeki değerler 0 satırını ve ikinci küme ayracı kümesindeki değerler 1. sırayı başlatır.
- Dolayısıyla, 1 ve 2 değerleri sırasıyla `b[0][0]` ve `b[0][1]` öğelerini başlatır ve 3 ve 4 değerleri `b [1][0]` ve `b[1][1]` öğelerini başlatır.

Çok boyutlu diziler

- Belirli bir satır için yeterli sayıda başlatıcı yoksa, o satırın kalan öğeleri 0 olarak başlatılır.
- Böylece,
- $\text{int } b[2][2] = \{\{1\}, \{3, 4\}\};$
- $b[0][0] \rightarrow 1$, $b[0][1] \rightarrow 0$, $b[1][0] \rightarrow 3$ ve $b[1][1] \rightarrow 4$ olarak başlatılır.

Çok boyutlu diziler



```
#include <stdio.h>
int main(void)
{
    int array1[2][3] = { { 1, 2, 3 }, { 4, 5, 6 } };
    printf ("Values in array1 by row are:");
    printArray(array1);

    int array2[2][3] = { 1, 2, 3, 4, 5 };
    printf ("Values in array2 by row are:");
    printArray(array2);

    int array3[2][3] = { { 1, 2 }, { 4 } };
    printf ("Values in array3 by row are:");
    printArray(array3);
}
```


1	2	3
4	5	6
1	2	3
4	5	0
1	2	0
4	0	0

- 10 elemanlı bir dizinin elemanlarını kullanıcıdan alıp, sonrasında küçükten büyüğe sıralayarak ekrana yazdırınız.
- Bir iskambil destesinin tüm kartlarını rastgele olarak `deste[4][13]` dizisinin içersine yerleştiriniz.

Egzersiz



Ders Sonu

