Dizilerin Tanımlanması

C d­ili ile programlamada b­ir d­izi­ i­le ­islem yapab­ilmek i­çi­n, d­iger veri­ ti­pleri­ndeki­ degi­skenlerde

oldugu gi­b­i, öncel­ikle d­izi­ni­n tanımlanması gerekmektedi­r.

Di­zi­leri­n tanımlanmasındak­i genel ­ifade; <dizi tip>i <dizi adı> [dizi uzunluğu];

D­iz­ini­n hang­i t­ipteki­ ver­ilerden olusacagını göster­ir (­int, char, double, float vb. ver­i t­ipler­ olabi­l­ir).

Tanımlanan di­z­ini­n adını ifade eder.

Kösel­i parantez ­içeri­s­inde beli­rt­ilen bu deger, d­iz­in­in uzunlugunu bel­irti­r.

int tamsayiDizisi[10];

char karakterDizisi[20];

**Dizilere Deger Atama**

Bir diziyi veri tipi, isim ve kapasite belirterek tanımladıgımızda bilgisayar hafızasında dizi için bir yer ayrılır; fakat dizi elemanlarına bir deger ataması yapılmaz.

Dizilere deger atamak için dizi tanımlaması ile birlikte veya tanımlamadan sonra kodlanan çesitli komutlar vardır. Dizinin elemanları üzerinde gezinecek döngüler aracılıgıyla dizi elemanları sıfırlanabilir, bir veri kaynagından kullanıcı, metin dosyası, veri tabanı vb.) alınan degerler dizinin elemanlarına atanabilir, dizi elemanlarının degerleri programcı tarafından kod içerisinde belirtilebilir. Bu noktada önemli olan husus, programcının ihtiyacı dogrultusunda en dogru yöntemin uygulanmasıdır.

#include <stdio.h>

int main(void) {

int dizi[6],

int i;

for(i=0; i<6; i++) {

dizi[i] = 0;

printf(“dizi[%d] = %d\n”, i, dizi[i]);

}

getch();

return 0;

}

dizi[0] = 0

dizi[1] = 0

dizi[2] = 0

dizi[3] = 0

dizi[4] = 0

dizi[5] = 0

Ana fonksiyon içerisinde int veri tipinde, dizi1 ve dizi2 adlarıyla, 5 eleman kapasiteli iki dizi tanımlanmıstır. Birinci dizinin tanımlanmasında dizinin tüm elemanlarına deger atanmıstır. Ikinci dizinin tanımlanmasında ise dizinin ilk iki elemanına deger atanmıstır. Bu dizinin diger elemanları otomatik olarak 0 degerini almıstır. Daha sonra kurulan döngüde dizilerin elemanları sırayla ekrana yazdırılmıstır.

#include <stdio.h>

#define N 5

int main(void) {

int dizi1[N] = {1,2,3,4,5};

int dizi2[N] = {1,2};

int i;

for(i=0; i<N; i++) {

printf(“dizi1[%d] = %d\t”, i, dizi1[i]);

printf(“dizi2[%d] = %d\n”, i, dizi2[i]);

}

getch();

return 0;

}

dizi1[0] = 1 dizi2[0] = 1

dizi1[1] = 2 dizi2[1] = 2

dizi1[2] = 3 dizi2[2] = 0

dizi1[3] = 4 dizi2[3] = 0

dizi1[4] = 5 dizi2[4] = 0

C d­ili­nde tanımlı scanf fonks­yonunu kullanarak, 5 elemanlı b­r tamsayı d­iz­isi­ne ver­ gi­ri­s­ yapılan bi­r program yazınız.

**Iki Boyutlu Diziler**

Satır ve sütunlardan olusan tablolar seklinde tanımlanabilen iki boyutlu diziler, çok boyutlu dizilerin en yalın halidir. m adet satır, n adet sütundan olusan iki boyutlu bir dizi, toplam *(m x n)* elemana sahip olabilir.

#include <stdio.h>

#define M 4

#define N 3

int main(void) {

int dizi[M][N] = {{1,2,3}, {4,5,6}, {7}, 8,9,10};

int i, j;

for(i=0; i<M; i++) {

for(j=0; j<N; j++) {

printf(“dizi[%d][%d] = %d\n”, i, j, dizi[i][j]);

}

}

getch();

return 0;

}

dizi[0][0] = 1

dizi[0][1] = 2

dizi[0][2] = 3

dizi[1][0] = 4

dizi[1][1] = 5

dizi[1][2] = 6

dizi[2][0] = 7

dizi[2][1] = 0

dizi[2][2] = 0

dizi[3][0] = 8

dizi[3][1] = 9

dizi[3][2] = 10

**Üç Boyutlu Diziler**

Üç boyutlu diziler, iki boyutlu dizilerin katmanlar halinde bir araya gelmesiyle olusur. Üç boyutlu bir diziyi, *“iki boyutlu dizilerin dizisi”* olarak da tanımlamak mümkündür. Boyut uzunlukları sırasıyla a, b, c olan üç boyutlu bir dizinin sahip olacagı toplam eleman sayısı a\*b\*c kadar olur.

#include <stdio.h>

int main(void) {

int d[3][2][2] = {12,11,10,9,8,7,6,5,4,3,2,1};

int i, j, k;

for(i=0; i<3; i++) {

for(j=0; j<2; j++) {

for(k=0; k<2; k++) {

printf(“%d\n”, d[i][j][k]);

}

}

}

getch();

return 0;

}

#include <stdio.h>

int main(void)

{

int idizi[10];

int id;

// Dizinin elemanlarına 1-10 arasındaki değerleri sıra ile atama

for (id=0; id<10; id++) {

idizi[id] = id+1;

}

// Dizi değerlerini ekrana yazma

for (id=0; id<10; id++) {

printf("%d ", idizi[id]);

}

return 0;

}

Yukarıdaki programı derleyip çalıştırdığımızda, aşağıdaki ifadeleri ekrana yazar:

1 2 3 4 5 6 7 8 9 10

Program önce 10 elemanlık int bir dizi bildirimi yapar. Bir for döngüsü ile dizi elemanlarına 1-10 arasındaki değerleri atar. İkinci for döngüsü ile dizi değerlerini ekrana yazar. İlk for döngüsü sonunda dizinin aldığı değerler aşağıda gösterildiği şekildedir:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **dizi[0]** | **dizi[1]** | **dizi[2]** | **dizi[3]** | **dizi[4]** | **dizi[5]** | **dizi[6]** | **dizi[7]** | **dizi[8]** | **dizi[9]** |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |

**Tek boyutlu dizilere ilk değer atama**

Tıpkı diğer değişkenler de olduğu gibi, dizi elemanlarına da dizinin tanımlandığı işlem satırında bir ilk değer verebiliriz. Bu işlemi yerine getirmek için aşağıdaki genel yapı kullanılır:

**veri-türü dizi-adı [boyut] = { sabit-değerler };**

Yukarıdaki satırda yer alan sabit değerlerin veri türü dizi veri türü ile aynı olmalıdır.

**Tek boyutlu int dizilere ilk değer atama**

Sabit değerler listesinde yer alan sabitler sırası ile dizi elemanlarına atanır. Örneğin, aşağıdaki işlem satırı tek boyutlu ve 5 elemanlı idizi dizisine sırası ile 1, 2, 3, 4 ve 5 sayılarını atar:

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

Yukarıdaki satırda yapılan diziye ilk değer atama işlemi, aşağıdaki işlem satırlarının yaptığı işin aynısını yapar:

int idizi[5], id;

for (id=0; id<5; id++) idizi[id] = id+1;

Şimdi, dizilere ilk değer atama yöntemini bir örnek üzerinde incelemeye çalışalım:

Örnek

#include <stdio.h>

int main(void)

{

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

int id;

for (id=0; id<5; id++) printf("%d ", idizi[id]);

return 0;

}

Yukarıdaki programı derleyip çalıştırdığımızda, aşağıdaki ifadeleri ekrana yazar:

1 2 3 4 5

Program önce oluşturduğu tek boyutlu beş elemanlı ve idizi adlı int bir diziye, 1'den 5'e kadar olan sayıları ilk değer atama metodu ile atar ve bir for döngüsü ile dizi elemanlarını ekrana yazar.

**Tek boyutlu char dizilere ilk değer atama**

Karakter dizilerine ilk değer atama işlemi ise iki farklı şekilde yapılabilir.

Karakter dizisinin her elemanına ayrı ayrı değer atanacaksa, diziye atanacak karakter sayısının bir fazlası kadar dizi boyutu tanımlamak ve en son elemana '\0' karakteri atamak gerekir.

Eğer bir karakter dizisine toplu değer ataması yapılacaksa, karakter dizisi (" ") işaretleri arasında tanımlanmalıdır. Bu durumda dizi boyutu karakter dizisinde yer alan karakter sayısından bir fazla tanımlanır ve karakter dizisinin sonuna derleyici tarafından otomatik olarak '\0' karakteri eklenir.

char cdizi[8] = { 'D', 'i', 'z', 'i', 'l', 'e', 'r', '\0' }; // Her bir dizi elemanına ayrı ayrı değer atama

char cdizi[8] = "Diziler"; // Diziye toplu değer atama

Yukarıdaki her iki işlem satırı da aynı sayıda karakteri dizilere atar. Ancak, ikinci işlem satırındaki dizinin son elemanı derleyici tarafından eklenecek olan 0 değeri için ayrılmıştır. İlk değer atama işlemi için bir karakter dizisi kullanılırsa, { } işaretleri işlem satırında yer almaz.

Karakter dizilerine ilk değer atama yöntemini kullanırken, atanan değerler ile tanımlanan dizi boyutu birbirine uygun olmalıdır. Aksi takdirde, derleyici programını derlerken hata verir.

Şimdi, karakter dizilerine ilk değer atama işlemini örnekler üzerinde incelemeye çalışalım:

Örnek

#include <stdio.h>

int main(void)

{

char cdizi1[9] = { 'K', 'a', 'r', 'a', 'k', 't', 'e', 'r', '\0' };

char cdizi2[9] = "dizileri" ;

printf("%s %s", cdizi1, cdizi2);

return 0;

}

Yukarıdaki programı derleyip çalıştırdığımızda, aşağıdaki ifadeleri ekrana yazar:

Karakter dizileri

Program önce cdizi1 ve cdizi2 adlı tek boyutlu ve 9 elemanlı 2 adet char dizi oluşturur. cdizi1 dizisine "Karakter" kelimesini, cdizi2 dizisine "dizileri" kelimesini ilk değer atama metodu ile atar. Sonra her iki dizinin içeriğini ekrana yazar.

**Karakter dizileri**

\* C'de **karakter dizisi** olarak tanımlanan bir **veri türü yoktur**.

\* Karakter dizisi veri desteğini sağlamak için tek boyutlu char diziler kullanılır.

\* Karakter dizisi **NULL(0)** değer (**'\0'**) ile sona eren bir char dizidir.

Bir karakter dizisi tanımlandığında, derleyici otomatik olarak dizi sonuna 0 değeri ekleyerek dizi sonunu belirler.

char dizilere değer atarken ve ekrana yazdırırken karakterlere tek tek işlem yapılacağı zaman dizinin en sonunda dizi sonu tanımlayan '\0' karakteri için yer ayırmaya gerek yoktur. Ancak, diziye tek seferde işlem yapılacağı zaman dizi sonunda '\0' karakteri mutlaka tanımlanmalıdır.

Bu nedenle, bir karakter dizisi içerecek bir karakter dizisi tanımlarken, dizi boyutunu karakter dizisinden **bir byte yani bir karakter fazla olacak şekilde** daha uzun tanımlamak gerekir. Derleyici 0 değerini diziye eklemek için ayrılan bir byte'lık boş alanı kullanacaktır.

Klavyeden veri okuma işleminde, girilebilecek maksimum karakter sayısından (count-1) daha az karakter girilirse, giriş işlemini sona erdiren yeni satır karakteri (10), '\0' karakterinden önce diziye eklenir.

\* fgets() fonksiyonu ile karakter dizisi okuturken okunacak karakter sayısı tanımlanır.

Şimdi, klavyeden girilen karakter dizilerini okumak için, fgets() fonksiyonunun kullanılmasını bir örnek üzerinde incelemeye çalışalım:

Örnek

#include <stdio.h>

int main(void)

{

char cdizi[40];

printf("Bir karakter dizisi giriniz:\n");

fgets(cdizi, 40, stdin); // En fazla 39 adet karakter okur.

printf(cdizi);

return 0;

}

Yukarıdaki örnekte, program 40 elemanlı bir karakter dizi bildirimi yaptıktan sonra bir karakter dizisi girilmesini ister. Girilen karakter dizisini fgets() fonksiyonu ile okuyarak cdizi adlı diziye atar ve dizi içeriğini ekrana yazar.

Bunlar giriş, çıkış ve hata çıktısı için standart akışlardır.

Standart giriş, standart çıktı ve standart hata ekranda yazdırıldığında varsayılan olarak klavyeden okunurdur.

Aşağıdaki akış işaretçileri standart akışlara erişmek için kullanılabilir:

| **AÇIKLAMALAR** | |
| --- | --- |
| **İşaretçi** | **Akış** |
| stdin | Standart giriş |
| stdout | Standart çıktı |
| stderr | Standart hata |

https://www.bilgigunlugum.net/