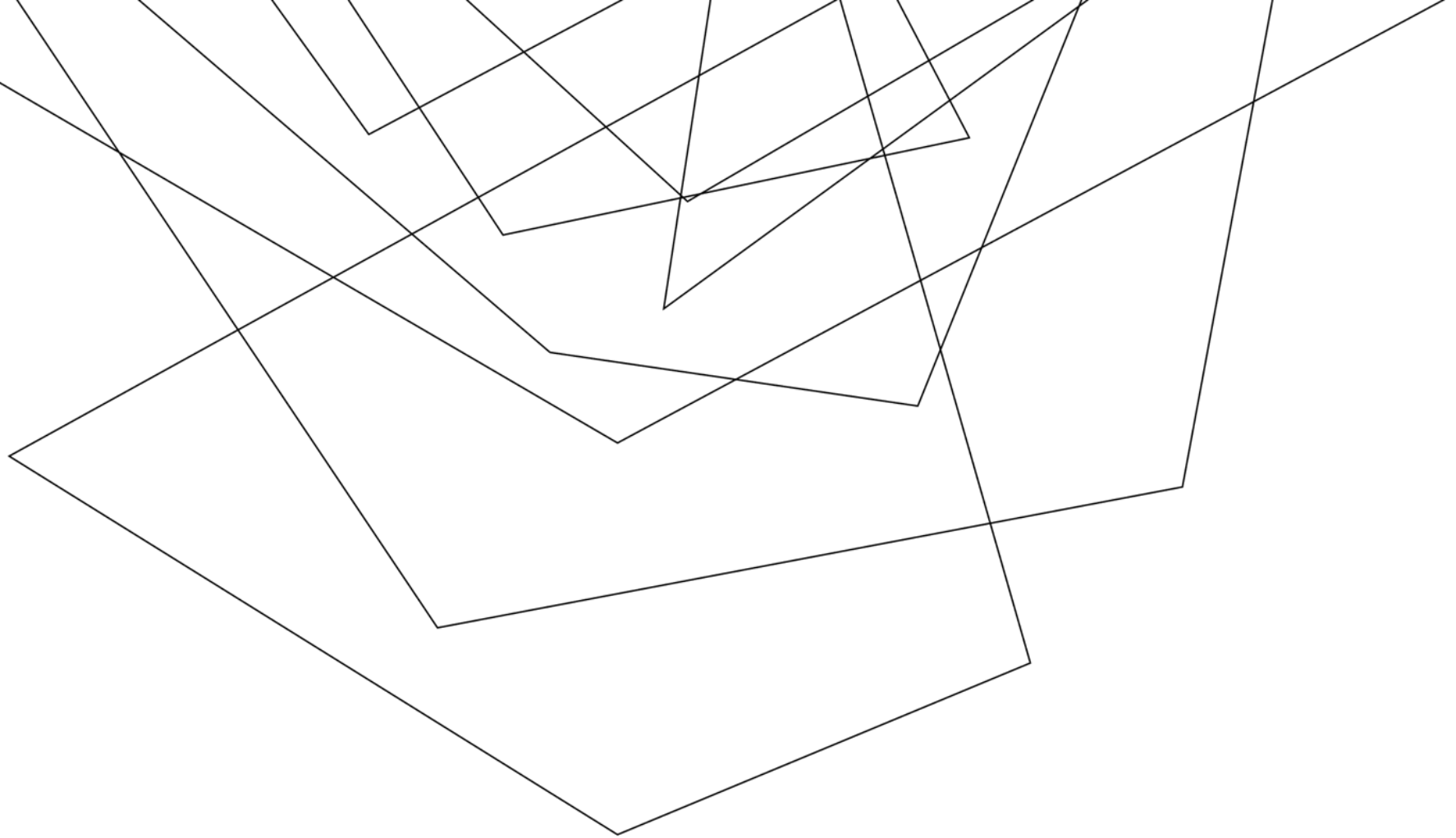

6- DİZİLER

DİZİ (ARRAY)

- C'de, aynı veri tipindeki değişkenleri bir araya getirerek dizi (array) adlı bir veri yapısı oluşturabiliriz.
- Aynı veri türünden ve farklı isimlere sahip çok fazla sayıda değişken tanımlamak yerine, dizileri kullanmak daha pratiktir.
- Dizi tanımlandığında, dizinin her bir elemanına dizi adı ile birlikte 0'dan başlayarak kullanılan indeks değeri ile erişim sağlanır.

TEK BOYUTLU DİZİ

- Tek boyutlu dizilerin bildirimi için kullanılan genel yapı:
  - veri-tipi dizi-adı[boyut];
- Örnek:
  - int dizi[10];
  - char strings[50];
  - float numArray[3];

TEK BOYUTLU DİZİ

- Tek boyutlu dizinin elemanlarına erişme ve değer atama:

```
int dizi[10];   // 10 elemanlı dizi adlı bir dizi oluşturur.
dizi[0] = 21;  // Dizinin ilk elemanına 21 değerini atar.
dizi[3] = 36;  // Dizinin dördüncü elemanına 36 değerini atar.
dizi[7] = 174; // Dizinin sekizinci elemanına 174 değerini atar.
```

- Birçok programlama dilinde diziler 0. elemanla başlar.

21			**36**				**174**		
0	1	2	3	4	5	6	7	8	9

TEK BOYUTLU DİZİ

21	?	?	36	?	?	?	174	?	?
0	1	2	3	4	5	6	7	8	9

- Atama yapılmayan elemanlarda rastgele değerler yer alabilir.
- C dili oluşturulan bir dizi için ilgili veri tipine uygun büyüklüklerde bellekte ardışık yerler tahsis eder ancak bu adreslerde daha önceki işlemlerden kalan bitleri silmez.
- Bu nedenle eğer dizi elemanları üzerinde matematiksel işlem yapılacaksa diziyi oluşturduktan sonra her eleman tek tek sıfırlanmalıdır.

TEK BOYUTLU DİZİ

- Dizi değerlerine erişim ve atama için döngü kullanılabilir.

```
#include <stdio.h>
int main() {
    int array[5];
    for (int i = 0; i < 5; i++) {
        array[i] = 0;
    }
    return 0;
}
```

TEK BOYUTLU DİZİ

- Dizi değerlerine ilk atama (initialization) dizi tanımının yapıldığı satırda yapılabilir.

```
#include <stdio.h>
int main() {
    int array[5] = { 1, 2, 3, 4, 5 };
    return 0;
}
```

TEK BOYUTLU DİZİ

- Char tipindeki dizilere iki türlü ilk atama yapılabilir:

```
#include <stdio.h>
int main() {
    char string[8] = { ‘d', 'i', 'z', 'i', 'l', 'e', 'r', '\0’ };
    char string[8] = "Diziler";
    return 0;
}
```

TEK BOYUTLU DİZİ

- 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.

İKİ BOYUTLU DİZİ

- İki boyutlu dizilerin bildirimi için kullanılan genel yapı:
  - veri-tipi dizi-adı[boyut1][boyut2];
- Örnek:
  - int matris[3][5];
  - char table[10][10];
  - float 2d_array[2][2];

İKİ BOYUTLU DİZİ

- İki boyutlu diziler için iç içe iki döngü kullanılabilir:

```
#include <stdio.h>
int main() {
    int matrix[3][5];
    for (int i = 0; i < 3; i++) {
        for (int j = 0; j < 5; j++) {
            matrix[i][j] = 0;
        }
    }
    return 0;
}
```

ÇOK BOYUTLU DİZİ

- veri-tipi dizi-adı[boyut1][boyut2]..[boyut_n];
- char calendar[31][12][5000];