



Arrays

Multi-Dimensional Arrays

Lesson:
JAVA Chapter 11

Arrays

Arrays aynı data type sahip birden fazla değer depolayan object'tır.

Arrays'de sadece primitive datalar veya non-primitive datalara ait referans değerleri depolanır.

* TRICK :) Arrays'de farklı data type depolanmaz.



Arrays



Ördek obj.
Referans değeri

Arrays



Arrays

Avantajları

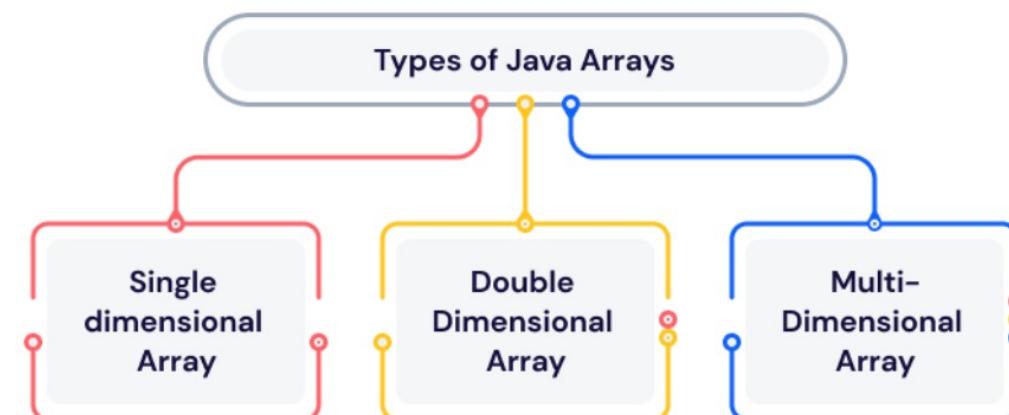
Kod Optimizasyonu: Kodu optimize eder, verileri verimli bir şekilde alabilir veya sıralayabiliriz.

Rastgele erişim: Bir dizin konumunda bulunan herhangi bir veriyi alabiliriz.

Dezavantajları

Array'in uzunluğu değiştirilemez. Bunun için ana bellekte yeni bir obj. tanımlanmalıdır.

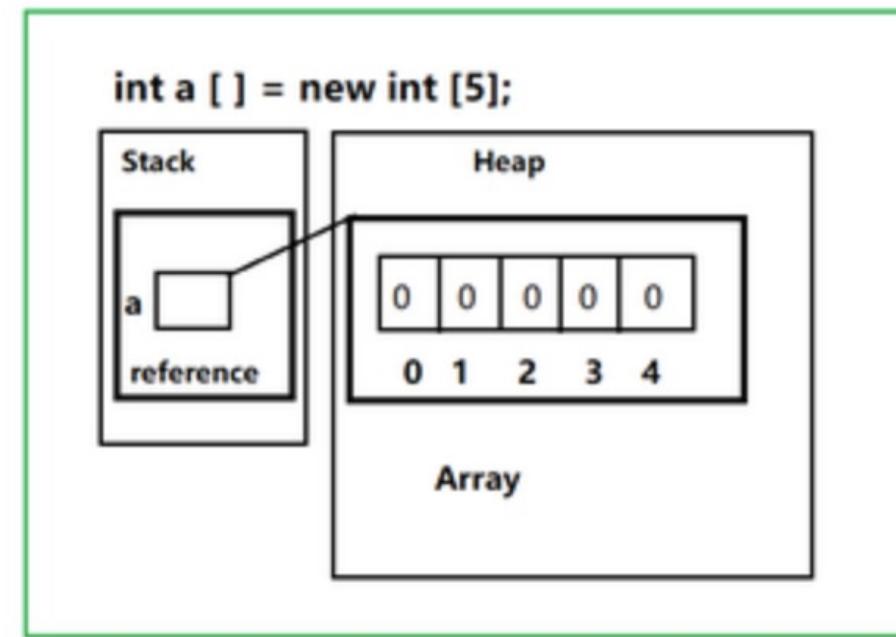
Array'in data type'ı tektir. Birbirinden farklı data type atanamaz.



Arrays

Array data type olarak **object (Non-Primitive)** olduğu için;

- * Heap Memory'de depolanır.
- * Value ile birlikte method'lara da sahiptirler.
- * Runtime'da oluşturulurlar.



Array Declaration (Dizi Tanımlama)

Array Declaration (Tanımlama):

- **int ageArray[];** -> Best Practice.
- **int [] ageArray;**

```
int [ ] array1 = { 5, 17, 350 }  
array data type      the square brackets indicate that it is an array  
array name          in braces specified numbers that we assigned in array1
```

Array Create:

```
veriTipi [] diziAdı = new veriTipi [elemanSayısı];
```

int ageArray[] = new int[7]; → length'i 7 olan ageArray create edildi.

* TRICK :) Array create edilirken length'i tanımlanmazsa CTE alınır.
Array'e eleman atanmazsa data type'a göre default değerler atanır.



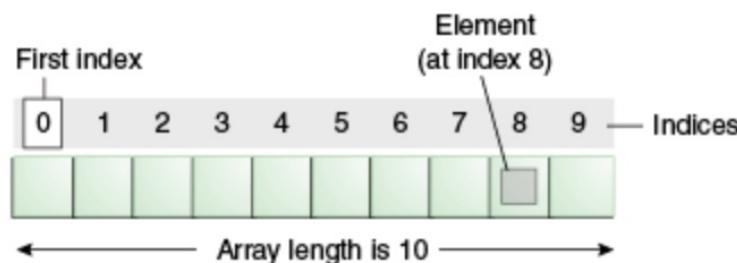
Arrays Assignment

Array'e Değer Atama:

`int ageArray[] = {20, 17, 34};` → Array declaration ve value assaignment tek satırda yapılabilir.

`int ageArray[] = new int[3];` → Array declaration yapılp index no ile value assaignment teker teker ayrı satırlarda yapılabilir

```
ageArray[0] = 20;  
ageArray[1] = 17;  
ageArray[2] = 34;
```



Arrays

Array'in elemanlarına ulaşma ve update etme:

```
int ageArray[ ] = {20, 17, 34};
```

Array index tabanlıdır, Array'in ilk elemanı 0. Index'te depolanır, son eleman (`length-1`). index'te saklanır.

Array elemanlarına index'i girilerek ulaşılır.

`ageArray [0]` → 20,

`ageArray [1]` → 17,

`ageArray [2]` → 34,

* TRICK :) Array'de olmayan index kullanılırsa “`ArraysIndexOutOfBoundsException`” hatası alınır.



Array Length (Uzunluk)

Array Uzunluğu (length):

```
int ageArray[ ] = {20, 17, 34};  
int size= ageArray.length;
```

* TRICK :) String ve Array length method'larında syntax farkı vardır.

String → **length()**

Arrays → **length**



Arrays

Array Elemanları Print Etme:

```
int ageArray[ ] = {20, 17, 34};
```

```
* for ( int i= 0 ;i<ageArray.length;i++){
```

```
    System.out.println(ageArray[i]);
```

```
}
```

```
* System.out.println(Arrays.toString(ageArray)); → Best Practice
```



Arrays

Array Elemanları Sıralama:

```
int ageArray[ ] = {20, 17, 34};
```

Arrays. **sort** (ageArray); → naturel (doğal k->b) sıralama

* TRICK :) Ters Sıralama (b->k) için...

-**sort()** method ile naturel sıralama sonrası loop ile index no değiştirilir



Arrays Binary Search

Array'de Arama Yapma:

Bir array'de herhangi bir elemanın varlığını kontrol etmek için öncelikle **Arrays.sort()** methodu ile sıralama yapılmalı sonra **Arrays.binarySearch()** method'u ile arama yapılır ve kontrolü yapılan elemanın index değeri return edilir.

```
String[] programlamaDilleri = {"C++", "Java", "C#", "C", "Php", "Python"};
Arrays.sort(programlamaDilleri); // C,C#,C++,Java,Php,Python
System.out.println(Arrays.binarySearch(programlamaDilleri, key: "Java")); // java elemanı 3. index
System.out.println(Arrays.binarySearch(programlamaDilleri, key: "C#")); // C# elemanı 1. index
System.out.println(Arrays.binarySearch(programlamaDilleri, key: "COBOL")); // COBOL elemanı arrayda mevcut değil (-insertion point) - 1=-4. index
System.out.println(Arrays.binarySearch(programlamaDilleri, key: "Assembly")); // Assembly elemanı arrayda mevcut değil (-insertion point) - 1=-1. index
```

* TRICK :) Varlığı kontrol edilen eleman array'de yoksa output $(-(\text{insertion point}) - 1)$.index olur.
 $(-\text{insertion point}) - 1 \rightarrow$ elemanın array' ekleneceği nokta index'i



Arrays Equals

Array Eşitliği:

`Arrays.equals()` methodu parametre olarak aldığı iki array'i eleman değeri ve index'lerine göre karşılaştırarak eşit olup olmadığını kontrol eder. İki array arasında eşitlik varsa **true**, yoksa **false** return eder.

```
int age[] = {11, 20, 17, 48, 34};  
int year[] = {34, 20, 48, 17, 11};  
System.out.println(Arrays.equals(age, year)); // false
```

```
Arrays.sort(age);  
Arrays.sort(year);  
System.out.println(Arrays.equals(age, year)); // true
```



String Convert to Arrays

String'i Array Elemanlarına Çevirme:

`obj.split()` method'u String'i girilen parametreye göre parçalayıp yeni String elemanlar create eder.

```
String str = "Java bilenin sırtı yere gelmez :) java bilmeyenin yüzü gülmez :( ";
String arr[] = str.split( regex: " " );
System.out.println(Arrays.toString(arr)); // [Java, bilenin, sırtı, yere, gelmez, :), java, bilmeyenin, yüzü, gülmez, :()]
String arr1[] = str.split( regex: ":" );
System.out.println(Arrays.toString(arr1)); // [Java bilenin sırtı yere gelmez , ) java bilmeyenin yüzü gülmez , ( ]
String arr2[] = str.split( regex: "" );
System.out.println(Arrays.toString(arr2)); // [J, a, v, a, , b, i, l, e, n, i, n, , s, ı, r, t, ı, , y, e, r, e,
                                         // , g, e, l, m, e, z, , :, ), , j, a, v, a, , b, i, l, m, e, y, e, n, i, n,
                                         // , y, ü, z, ü, , g, ü, l, m, e, z, , :, (, ]
```



Arrays Copy

Array Kopyalama:

Bir array'den belli bir uzunlukta yeni bir array oluşturmak için **Arrays.copyOf()** metodu, belli bir aralıkta yeni bir array oluşturmak için ise **Array.copyOfRange()** metodu kullanılır.

```
int num[] = {6, 1, 55, 21, 33, -321, -21, 2, -11, 27};
```

```
int[] copyArray = Arrays.copyOf(num, newLength: 3); // yeni array boyutu 3 olacak şekilde create edilir.  
System.out.println(Arrays.toString(copyArray)); // [6, 1, 55]
```

```
int[] copyOfRangeArray = Arrays.copyOfRange(num, from: 3, to: 5); // yeni array için 3. ve 4. index dahil 5. index hariç  
System.out.println(Arrays.toString(copyOfRangeArray)); // [21, 33]
```



Arrays Fill

Array'i Belirli Bir Eleman İle Update Etme:

`Arrays.fill()` metodu ile array'in belirli bir bölgelere yeni değer atanabilir. `Arrays.copyOf()` metodu, belli bir aralıkta yeni bir array oluşturmak için ise `Array.copyOfRange()` methodu ile değerler atanabilir.

```
int num[] = {15, 1, 99, 7, 7, -22, 11, 2, -49, 52};
```

```
Arrays.fill(num, val: 2);
```

```
System.out.println(Arrays.toString(num)); // [2, 2, 2, 2, 2, 2, 2, 2, 2]
```

```
Arrays.fill(num, fromIndex: 3, toIndex: 5, val: 7);
```

```
System.out.println(Arrays.toString(num)); // [2, 2, 2, 7, 7, 2, 2, 2, 2]
```



Arrays



OCA Question

What is the result of the following?

```
int[] random = { 6, -4, 12, 0, -10 };  
int x = 12;  
int y = Arrays.binarySearch(random, x);  
System.out.println(y);
```

- A. 2
- B. 4
- C. 6
- D. The result is undefined.
- E. An exception is thrown.
- F. The code does not compile.



Multi-Dimensional Arrays

Array iç içe birden çok Array'lerden meydana gelmişse array'e Multi Dimensional Array denir.

`int[][] intArray = {{10,20,30},{100,200,300}};`

The diagram illustrates a 2D array structure. It features a red border and rounded corners. Inside, there are two rows labeled 'Rows' (0 and 1) and three columns labeled 'Columns' (0, 1, and 2). The data elements are represented by colored boxes: the first row contains blue boxes with values 10, 20, and 30; the second row contains pink boxes with values 100, 200, and 300. The entire diagram is enclosed in a purple rounded rectangle.

		Columns		
		0	1	2
Rows	0	10	20	30
	1	100	200	300



Multi-Dimensional Arrays

Çok Boyutlu Diziler

Java'da Çok Boyutlu Diziler varsayılan bir veri tipi olarak bulunmazlar ve matris olarak adlandırılırlar. Dizilerin 2 boyutlu halleri şeklinde tanımlanırlar. Matrisler satır ve sütun şeklinde tablo verisi formatındaki verileri tutmak için kullanılır. Diziler liste halinde veriler için uygunken, matrisler tablo şeklindeki veriler için uygundur. Oluşturulan tabloda bir değere ulaşmak istersek satır ve sütun sayısını girmemiz yeterli olacaktır.

* **TRICK :)** İki boyutlu dizilerde tek boyutlu diziler gibi indis değeri 0'dan başlar. Dizide tutulacak veri tipleri aynı olmak zorundadır farklı veri tiplerini aynı matriste tutamayız.

3x3 boyutunda bir matris örneği :

	Column 0	Column 1	Column 2
Row 0	x[0][0]	x[0][1]	x[0][2]
Row 1	x[1][0]	x[1][1]	x[1][2]
Row 2	x[2][0]	x[2][1]	x[2][2]



Multi-Dimensional Arrays

```
int[][][] a = {  
    {1, 2, 3},  
    {4, 5, 6, 9},  
    {7},  
};
```

Gördüğümüz gibi, çok boyutlu dizinin her elemanı bir dizinin kendisidir. Ayrıca, C/C++'dan farklı olarak, Java'daki çok boyutlu dizinin her satırı farklı uzunluklarda olabilir.

	Column 1	Column 2	Column 3	Column 4
Row 1	1 a[0][0]	2 a[0][1]	3 a[0][2]	
Row 2	4 a[1][0]	5 a[1][1]	6 a[1][2]	9 a[1][3]
Row 3	7 a[2][0]			



Multi-Dimensional Arrays

Mesafe Tablosu(KM)

	İstanbul	Ankara	Adana	Bursa	Muğla	Sivas
İstanbul	0	453	939	243	783	891
Ankara	453	0	490	384	620	439
Adana	939	490	0	839	863	423
Bursa	243	384	839	0	544	823
Muğla	783	620	863	544	0	1045
Sivas	891	439	423	823	1045	0

```
int[][] uzaklik ={
    {0, 453, 939, 243, 783, 891},
    {453, 0, 490, 384, 620, 439},
    {939, 490, 0, 839, 863, 423},
    {243, 384, 839, 0, 544, 823},
    {783, 620, 863, 544, 0, 1045},
    {891, 439, 423, 823, 1045, 0}
};
```



Multi-Dimensional Arrays

**Array'i tanımlarken (declaration),
her bir boyut için [] kullanılmalıdır.**

degisenTipi[][] arraysmi;
int matrix[][];

İki boyutlu dizimizi tanımlarken değerlerini atamak istersek aşağıdaki yöntemi kullanabiliriz.

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

İlk köşeli parantez birinci boyutu (satırları), diğer ise ikinci boyutu (sütunları) belirtir. Aşağıdaki kodu çalıştırırsak, 3 satırlı ve 4 sütunlu bir matris oluşturur:

```
int matrix[][] = new int[3][4];
```

yada

```
matrix = new int[3][4];
```

oluşturulan matrix değişkeninin default değeri (a) ile gösterilen tablodaki gibidir.

```
[0] [1] [2] [3]  
[0] 0 0 0  
[1] 0 0 0  
[2] 0 0 0
```

(a)



Multi-Dimensional Arrays

Bu matrisin bütün elemanlarına ulaşmak için kullanmamız gereken indeks numaralarını aşağıdaki tabloda görebilirsiniz:

[0] [0]	[0] [1]	[0] [2]	[0] [3]
[1] [0]	[1] [1]	[1] [2]	[1] [3]
[2] [0]	[2] [1]	[2] [2]	[2] [3]

```
matrix[1][2]; // Matrisin 2. satır ve 3. sütunundaki elemana erişiliyor  
matrix[0][3]; // Matrisin 1. satır ve 4. sütunundaki elemana erişiliyor  
matrix[2][0]; // Matrisin 3. satır ve 1. sütunundaki elemana erişiliyor
```

Matrisin 2. satırı ve 3. sütununda yer alan değerini 7 yapalım.

```
matrix[2][1] = 7;
```



Multi-Dimensional Arrays

Multi Dimensional Array'in Tüm Elemanları Print Etme:

```
int num[][] = {  
    {1, 2, 3},  
    {4, 5, 6},  
    {7, 8, 9},  
    {10, 11, 12}  
};  
  
for (int i = 0; i < num.length; i++) {  
    for (int j = 0; j < num[i].length; j++) {  
        System.out.println(num[i][j] + " ");  
    }  
}
```

```
System.out.println(Arrays.deepToString(num)); // [[1, 2, 3], [4, 5, 6], [7, 8, 9], [10, 11, 12]] -> Best Practice  
System.out.println(Arrays.toString(num)); // [[I@22f71333, [I@13969fbe, [I@6aaa5eb0, [I@3498ed]] -> referans değerleri
```



THANKS!

Any questions?

HalUk Bilgin | JAVA Backend Developer

BAŞARI GAYRETE AŞIKTIR ☺

