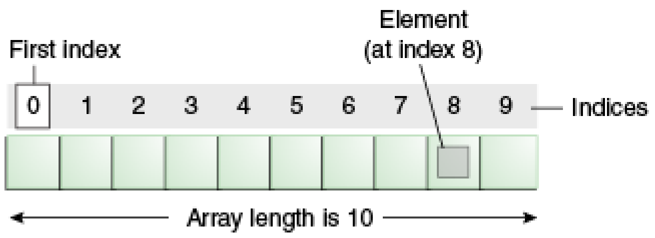
Arrays and String

An array is a collection of similar type of elements which is stored in a contiguous memory location.

Java array is an object which contains elements of a similar data type. Additionally, The elements of an array are stored in a contiguous memory location. It is a data structure where we store similar elements. We can store only a fixed set of elements in a Java array.

Array in Java is index-based, the first element of the array is stored at the 0th index, 2nd element is stored on 1st index and so on.



One Dimensional Array :

A list of items can be given one variable name using only one subscript such a variable is called a single-scripted or one –dimensional array.
Example
int a[]=new int[5];
for this array computer reserve five memory locations as

a[0]	a[1]	a[2]	a[3]	a[4]

The values assigned to the array elements, as follows:
a[0]=20
a[1]=40
a[2]=45
a[3]=100
a[4]=55

Creating Array:

Array must be declared and created in computer memory before they are used. Creation of array involves three steps:

1. Declare the array

2. Create memory locations
3. Put values into the locations

<u>Declaration of Arrays</u>

Array in java may be declared in two forms.

<u>Form1</u>

```
Datatype Arrayname[ ];
```

<u>Form2</u>

```
Datatype[ ] Arrayname;
```

Example:

```
int  number[ ];
floar percentage []
int[ ] count;
float[ ] average;
```

<u>Creation of Arrays</u>

After declaring the array, we need to create it in the memory.

Java allows us to create arrays using <u>new</u> operator only, as below :

```
Arrayname=new datatype[size];
```

Example:

```
number=new int[5];
average=new float[10];
```

these lines creates sufficient memory space for array number and average.

It is also possible to combine two steps – declaration and creation into one ,shown below :

```
int number [ ] = new int[5];
```

<u>Initialization of Array: [put values into array]</u>

Putting the values into the array is known as initialization . This is done using the array subscript as shown below.

```
Arrayname [Subscript] = value;
```

Ex.

```
Number[0]=5;
Number[1]=40;
..
Number[4]=60;
```

We can also initialize array automatically in the same wayas the ordinary variable when they are declared, as shown below:

```
Datatype  arrayname[ ] = { list of values };
```

Values separated by commas and surrounded by curly braces.

Ex.

```
Int number[ ] = {10,20,50,100,55};
```

Program to demonstrate one dimensional array

```
class Testarray
{
public static void main(String args[])
{
int a[]={33,3,4,5}; //declaration, instantiation and initialization
//printing array
for(int i=0;i<a.length;i++)  //length is the property of array
System.out.println(a[i]);
}
}
```

Output:

```
33
3
4
5
```

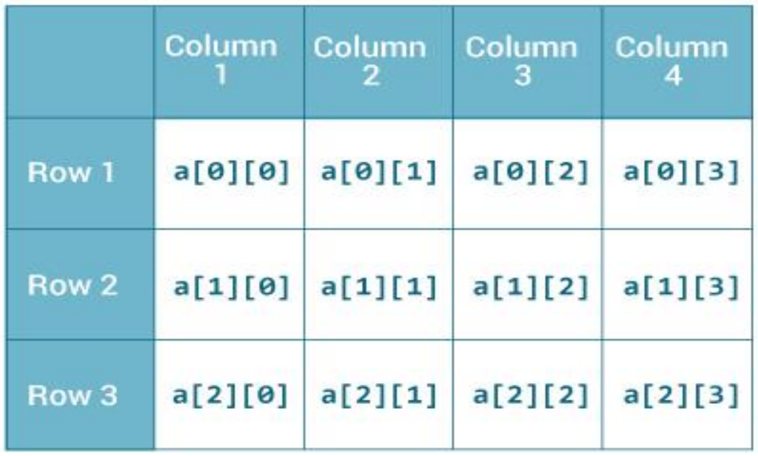
Two Dimensional array

Two dimensional array requires two subscript one for Row and another for Column .Data in two dimensional arrays are stored in tabular form (in row major order).

Declaration of 2D array:

```
int a[ ][ ];
a = new int [3][4];
        or
int[][] a = new int[3][4];
```

Here, a is a two-dimensional (2d) array. The array can hold maximum of 12 elements of type int.



	Column 1	Column 2	Column 3	Column 4
Row 1	a[0][0]	a[0][1]	a[0][2]	a[0][3]
Row 2	a[1][0]	a[1][1]	a[1][2]	a[1][3]
Row 3	a[2][0]	a[2][1]	a[2][2]	a[2][3]

Initialization of 2D array :

Ex.

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

//Program to demonstrate two dimensional array

```
class 2darray
{
public static void main(String args[])
{
//declaring and initializing 2D array
int arr[][]={{1,2,3},{2,4,5},{4,4,5}};
//printing 2D array
for(int i=0;i<3;i++)
{
 for(int j=0;j<3;j++)
{
   System.out.print(arr[i][j]+" ");
 }
 System.out.println();
}
}
}
```

Output:

```
1 2 3
2 4 5
4 4 5
```

String Array

Strings represent a sequence of character.
Java String array is used to hold fixed number of Strings
Java String array is basically an array of objects.
There are two ways to declare string array – declaration without size and declare with size.
There are two ways to initialize string array – at the time of declaration, populating values after declaration.
We can do different kind of processing on string array such as iteration, sorting, searching etc.

Java String Array Declaration

Below code shows different ways for string array declaration in java.

```
    String[ ] strArray; //declare without size

    String[ ] strArray1 = new String[3]; //declare with size
```

Note that we can also write string array as String strArray[] but above shows way is the standard and recommended way. Also in the above   code, strArray is null whereas strArray1 value is [null, null, null].
Java String Array Initialization
Let’s look at different ways to initialize string array in java.

```
//inline initialization
String[ ] strArray1 = new String[] {"BMW","Ford","AUDI"};
String[] strArray2 = {"BMW","Ford","AUDI"};

//initialization after declaration
String[] strArray3 = new String[3];
strArray3[0] = "BMW";
strArray3[1] = "FORD";
strArray3[2] = "AUDI";

//Program to demonstrate string array
public class StringExample
{
public static void main(String args[])
{
String s1="java";  //creating string by java string literal
char ch[]={'s','t','r','i','n','g','s'};
String s2=new String(ch);  //converting char array to string
String s3=new String("example");  //creating java string by new keyword
System.out.println(s1);
System.out.println(s2);
System.out.println(s3);
}
}
```

Output:

```
java
strings
example
```

String Methods:

The String class has a set of built-in methods that you can use on strings.

Method	**Description**	**Return Type**
charAt()	Returns the character at the specified index (position)	char
compareTo()	Compares two strings lexicographically	int
concat()	Appends a string to the end of another string	String
contains()	Checks whether a string contains a sequence of characters	boolean
equals()	Compares two strings. Returns true if the strings are equal, and false if not	boolean
equalsIgnoreCase()	Compares two strings, ignoring case considerations	boolean
format()	Returns a formatted string using the specified locale, format string, and arguments	String
getChars()	Copies characters from a string to an array of chars	void
indexOf()	Returns the position of the first found occurrence of specified characters in a string	int
isEmpty()	Checks whether a string is empty or not	boolean
lastIndexOf()	Returns the position of the last found occurrence of specified characters in a string	int

length()	Returns the length of a specified string	int
replace()	Searches a string for a specified value, and returns a new string where the specified values are replaced	String
replaceFirst()	Replaces the first occurrence of a substring that matches the given regular expression with the given replacement	String
replaceAll()	Replaces each substring of this string that matches the given regular expression with the given replacement	String
split()	Splits a string into an array of substrings	String[]
startsWith()	Checks whether a string starts with specified characters	boolean
subSequence()	Returns a new character sequence that is a subsequence of this sequence	CharSequence
substring()	Extracts the characters from a string, beginning at a specified start position, and through the specified number of character	String
toCharArray()	Converts this string to a new character array	char[]
toLowerCase()	Converts a string to lower case letters	String
toString()	Returns the value of a String object	String
toUpperCase()	Converts a string to upper case letters	String
trim()	Removes whitespace from both ends of a string	String

Ex.

1. String s1=”sachin”;
String s2 = s1.concat(“Tendulkar”);
System.out.println(s2);
Output:
Sachin Tendulkar
2. String s1=”HELLO”;
String s2 = s1.toLowerCase;
String s3 = s1.ChartAt(4);
System.out.println(s2);
System.out.println(s3);
Output:
hello
O

By

Prof. Suchitra K. Kasbe
Department of Computer Science

****