UNIT I – BASIC CONCEPTS **9**

Object Oriented Programming - Benefits of OOP – Applications of OOP. Java fundamentals: Features of java – Java development environment – Bytecode - Data types- Variables -Operators – Expressions – Functions – Static Members – Arrays – Strings – Classes and objects – Constructing objects using constructors.

STRINGS

String manipulation is the most common part of many Java programs. Strings represent a sequence of characters. While using char data type we can hood a single character. In Java Strings are class objects and implemented using two classes. They are

- String
- String Buffer

STRING CLASS

Strings may be declared and created as follows

Syntax:

String stringname;

Stringname = new String (“string”);

Example:

String firstname;

firstname=new

String(“Anil”);

The above two statements can be combined together as follows

String firstname=new String(“AAAA”);

Like arrays it is possible to get the length of string using length method of String class

Syntax:

datatype variablename =string name.length();

Example:

int m=firstname.length( );

Note the Java strings can be concatenated using + operator.

String fullname = name1 + name2;

Where name1 and name2 are Java Strings containing string constants.

STRING ARRAYS:

We can also create and use arrays that contain string.

Syntax:

String stringarrayname[ ] =new String[size];

Example:

String strarray [ ] = new String[3];
The above line will create an strarray of size 3 to hold three string constants.
Commonly used Inbuilt String Methods:

Functions	Task performed
S2=S1.toLowerCase;	Converts the string s1 to all lowercase
S2=S1.toUpperCase;	Converts the string s1 to all uppercase
S2=S1.replace(‘x’, ’y’);	Replace all appearances of x with y
S2=S1.trim();	Remove white spaces at the beginning and end of the string s1
S1.equals(s2)	Returns ‘true’ if s1 is equals to s2
S1.equalsIgnoreCase(s2)	Same as equals but cannot consider upper or lower case
S1.length()	Gives the length of s1
S1.charAt(n)	Gives nth character of s1
S1.compareTo(s2)	Returns negative if s1<s2,positive if s1>s2,and zero if s1=s2
S1.concat(s2)	Concatenates s1 and s2
S1.subString(n)	Gives substring starting from nth character
S1.subString(n,m)	Gives substring starting from nth character up to mth
S1.indexOf(‘x’)	Gives the position of the first occurrence of ‘x’ in the string s1
S1.indexOf(‘x’,n)	Gives the position of ‘x’ that occurs after nth position in the string s1

Example Program to implement Strings and their methods:

```
class StringExample
{
	public static void main(String s[])
	{
		int i1,i2,i3;
		String a="Good";
		String b="Morning";
		String c;
		c=a+b;
```

```
            System.out.println("First String:"+a);
            System.out.println("Second String:"+b);
            i1=a.length();
            i2=b.length();
            i3=c.length();
            System.out.println("Length of First String:"+i1);
            System.out.println("Length of Second String:"+i2);
            System.out.println("Concatenated String :"+c);
            System.out.println("Length of Concatenated
            String:"+i3;
            if(c.equals(a))
            {
                  System.out.println("The first and concatenated strings are equal!");
            }
            else
            {
                  System.out.println("The first and concatenated strings are not
            equal!");
            }
      }
}
```

Execution & Output:

```
Z:\>javac string.java
Z:\>java string
First String:Good
First String:Morning
Length of First String:4
Length of Second String:7
Concatenated String :GoodMorning
Length of Concatenated String:11
The first and concatenated strings are not equal!
```

Explanation:

Program execution starts from the Main function. Next line to be executed is given below

```
String a="Good";
String b="Morning";
```

Here we are assigning the text Good to String a and Morning to String b. Next line to be executed is given as follows.

String c; c=a+b;
c is declared as String data type, we are concatenating the content in String a and content in String b and storing the resultant in String c as output.Out put stored in c is Good morning
l1=a.length();
l2=b.length();
l3=c.length();
The above lines calculates the length of String and stores in respective variables l1,l2 and l3.
Next line to be executed is if(c.equals(a))
Here we are going to compare content of string c with content of string a.If both strings are equal.We are going to print a statement that strings are equal otherwise we are going to print a statement that strings are not equal.

Program 2:

```
class Test
{
	public static void main(String ar[])
	{
		String s1=new String(“XYZ”);
		String s2=new String(“college”);
		System.out.println(s1.toLowerCase());
		System.out.println(s2.toUpperCase());
		System.out.println(s1.replace(‘X’,’Y’);
		System.out.println(s1.trim());
		System.out.println(s1.equals(s2));
		System.out.println(s1.equalsIgnoreCase(s2);
		System.out.println(s1.length());
		System.out.println(s1.CharAt(3));
		System.out.println(s1.compareTo(s2));
		System.out.println(s1.concat(s2));
		System.out.println(s1.substring(1));
		System.out.println(s2.substring(0,3);
		System.out.println(s1.indexOf(‘Y’);
		System.out.println(s2.indexOf(‘e’,4));
	}
}
```

Output

xyz
COLLEGE
YYZ
XYZ
False
False
3
Y
-1
XYZcollege
YZ
coll
2
6

Other Examples:

1.charAt()

```
public class CharAtExample
{
	public static void main(String args[])
	{
		String name="javatpoint";
		char ch=name.charAt(4);//returns the char value at the 4th index
		System.out.println(ch);
	}
}
```

Output: t

2.compareTo()

The java string compareTo() method compares the given string with current string lexicographically. It returns a positive number, negative number or 0.

```
public class LastIndexOfExample
{
	public static void main(String args[])
	{
		String s1="hello";
		String s2="hello";
```

```
                String s3="meklo";
                String s4="hemlo";
                System.out.println(s1.compareTo(s2));
                System.out.println(s1.compareTo(s3));
                System.out.println(s1.compareTo(s4));
        }
}
```

o/p:

```
 0
-5
-1
```

3. String concat()

```
public class ConcatExample
{
        public static void main(String args[])
        {
                String s1="java string";
                s1.concat("is immutable");
                System.out.println(s1);
                s1=s1.concat(" is immutable so assign it explicitly");
                System.out.println(s1);
        }
}
```

o/p: java string
java string is immutable so assign it explicitly

4. String contains()

```
class ContainsExample
{
        public static void main(String args[])
        {
                String name="what do you know about me";
                System.out.println(name.contains("do you know"));
                System.out.println(name.contains("about"));
                System.out.println(name.contains("hello"));
        }
```

```
}
```
o/p: true
true
false

5. String endsWith

```
public class EndsWithExample
{
	public static void main(String args[])
	{
		String s1="java by javatpoint";
		System.out.println(s1.endsWith("t"));
		System.out.println(s1.startsWith("point"));
	}
}
```
o/p: true
true

6. String equals

```
public class EqualsExample
{
	public static void main(String args[])
	{
		String s1="javatpoint";
		String s2="javatpoint";
		String s3="JAVATPOINT";
		String s4="python";
		System.out.println(s1.equals(s2));//true because content and case is same
		System.out.println(s1.equals(s3));//false because case is not same
		System.out.println(s1.equals(s4));//false because content is not same
	}
}
```
o/p: true
false
false

7. String toUpperCase() and toLowerCase() method

```
class Example
{
	public static void main(String args[])
	{
		String s="Sachin";
		System.out.println(s.toUpperCase());//SACHIN
		System.out.println(s.toLowerCase());//sachin
		System.out.println(s);
	}
}
```

o/p:

SACHIN
sachin
Sachin

8. trim() method

```
class TrimExample
{
	public static void main(String args[])
	{
		String s=" Sachin ";
		System.out.println(s);// Sachin
		System.out.println(s.trim());//Sachin
	}
}
```

o/p:

Sachin
Sachin

9. startsWith() and endsWith() method

```
class Example
{
	public static void main(String args[])
	{
		String s="Sachin";
		System.out.println(s.startsWith("Sa"));//true
		System.out.println(s.endsWith("n"));//true
```

```
        }
}
```

o/p:

true
true

10. charAt() method

```
public class CharAtExample
{
        public static void main(String args[])
        {
                String s="Sachin";
                System.out.println(s.charAt(0));//S
                System.out.println(s.charAt(3));//h
        }
}
```

o/p:

S
h

11. String length() method

```
class LengthExample
{
        public static void main(String args[])
        {
                String s="Sachin";
                System.out.println(s.length());//6
        }
}
```

o/p:6

12. valueOf() method

The string valueOf() method converts given types such as int, long, float, double, boolean, char and char array into string.

```
class ValueOfExample
{
        public static void main(String args[])
```

```
        {
                int a=10;
                String s=String.valueOf(a);
                System.out.println(s+10);
        }
}
```

o/p:1010

13. replace() method

```
class LastIndexOfExample
{
        public static void main(String args[])
        {
                String s1="Java is a programming language. Java is an Island.";
                String replaceString=s1.replace("Java","Kava");//replaces all occurrences
        of "Java" to "Kava"
                System.out.println(replaceString);
        }
}
```

o/p: Kava is a programming language. Java is a platform. Java is an Island.

14. replaceAll: replace character

The java string replaceAll() method returns a string replacing all the sequence of characters matching regex and replacement string.

```
class ReplaceAllExample1
{
        public static void main(String args[])
        {
                String s1="javatpoint is a very good website";
                String replaceString=s1.replaceAll("a","e");//replaces all occurrences of "a"
        to "e"
                System.out.println(replaceString);
        }
}
```

o/p: jevetpoint is e very good website

15. replaceAll() example: replace word

```
public class ReplaceAllExample2
{
	public static void main(String args[])
	{
		String s1="My name is Khan. My name is Bob. My name is Sonoo.";
		String replaceString=s1.replaceAll("is","was");//replaces all occurrences of
	"is" to "was"
		System.out.println(replaceString);
	}
}
```

o/p: My name was Khan. My name was Bob. My name was Sonoo.

CONSTRUCTOR

It is very simple and easier to initialize the object when it is created.Java supports this functionality with the help of constructors.Constructors are similar to classname, that enables an object to initialize itself when it is created.They do not specify a return type not even void.This is because they return the instance of the class itself

Syntax:

```
class classname
{
	Variable declaration;
	Constructorname(arguments)
	{ ..... }
}
```

Example program to illustrate constructors:

```
class Rectangle
{
	int length,width;
	Rectangle(int x, int y)
	{
		length=x;
		width=y;
	}
	int rectarea( )
	{
		return(length*width);
	}
```

```
}
class RectangleArea
{
        public static void main(String args[ ] )
        {
                Rectangle rect1=new Rectangle(15,10);
                int area1=rect1.rectarea();
                System.out.println("Area:",+area1);
        }
}
```

Output: Area: 150

Explanation:

Even though there are two classes we have to save the file in the name of the class where we have the main function. So we are going to save the file in the name of "RectangleArea.java". and compilation is performed as follows

Compilation : Z:\>javac rectanglearea.java

Interpretation: Z:\>java rectanglearea

Program execution starts from the Main function. Next line to execute is rectangle rect1=new Rectangle(15,10);

Here object is created as rect1 and as object is created constructor is executed with arguments 15,10.and the control transfers to the class rectangle.Next line to be executed is Here 15,10 is passed as arguments and 15 is assigned to length and 10 is assigned to width. Execution transfers back to class with main function and executes

int area1=rect1.rectarea();

Here method called rectarea in class rectangle is called and area is calculated and area=150 is printed as output.
