String Handling

String is an array of characters in many programming languages.In Java String is also an array of characters.

But mainly a String is a class in Java.It is a predefined class in java.lang package.

Java.lang package is a default package in Java.

There are two String Handling classes available in Java namely

String

StringBuffer

Both String and StringBuffer are final classes.

Both these classes are available in java.lang package.

String :

It is a final class in java.lang package.

A String class is an object in Java.

A String is immutable, which means that it is fixed and the contents of a String are always constant.

String constructors :

We can create an object of a String class by different ways by using different types of constructors.

String s1=new String()

String s2=new String(“program”);

String s3=new String(s2);

String s4=”program”;

char name[]={‘J’,’I’,’T’,’M’};

String s5=new String(name);

There are many methods available in **String** class.

Some of the important methods are:

length():

This method is used to find the length or size of a String.This method returns number of characters present in the given String.

Syntax of the method:

int length()

Eg:

String s=”CoreJava”;

int i=s.length();

length() returns the length of the string and its length is stored in i which is **8**

concat():

This method is used to join or concat two strings.

The return type of this method is String.

This method takes one parameter which is also a String

String concat(String str)

Where str is the String joined to the given String.

Eg:

```
String     s1=”Java”;
String     s2=”Program”;
String s3=s1.concat(s2);
```

Where the output of s3 is **JavaProgram**

Concatenation operator(+):

The operator “+” is called a concatenation operator.

It is similar to concat() method.

In Java **+** is used to join any strings.

Eg:

```
String     s1=”Java”;
String     s2=”Program”;
String s3=s1+s2;
```

Where the output of s3 is **JavaProgram**

toLowerCase():

This method is used to convert upper case letters of a String to lowercase.The return type of this method is String.

String toLowerCase()

Eg:

String s=”JAVA PROGRAM”;

The output of

s.toLowerCase() is **java program**

toUpperCase():

This method is used to convert lower case letters of a String to uppercase.The return type of this method is String.

String toUpperCase()

Eg:

String s=” **java**”;

The output of

s.toLowerCase() is **JAVA**

trim():

This method is used to filter white spaces before and after a given String.

This mehod is not used to filter white spaces between any two words of a String.

The return type of this method is String

String trim()

String s=" Learn Java Basics ";

The output of

s.trim() is **Learn Java Basics**

replace():

This method is used to replace a given character of a String with another new character . The return type of this method is String.

This method replaces the occurrence of the given character through out the String by the new character.

String replace(char original,char new)

Eg:

String s="Liver";

The output of

s.replace('L','R') is **River**

String Comparision Methods:

There are three types of String Comparision in Java.

They are :

(i)equals()

(ii)==

(iii)compareTo()

equals():

This method is used to compare actual contents of the two strings.The return type of this method is a boolean value.

boolean equals(String s)

Eg:

String s1=new String(“rama”);

String s2=new String(“jaya”);

String s3=new String(“rama”);

String s4=new String(“RAMA”);

The output of

if(s1.equals(s2)) is **false**

if(s1.equals(s4)) is **false**

if(s1.equals(s3)) is **true**

equalsIgnoreCase():

This method is used to compare actual contents of the two strings same as equals() except that it neglects case sensitiveness of letters.

Eg:

String s1=new String(“rama”);

String s2=new String(“RAMA”);

The output of

if(s1.equals(s2)) is **false**

if(s1.equalsIgnoreCase(s3)) is **true**

compareTo():

This method is used to compare two strings depending on the ASCII values of the characters of the string.

The return type of this method is int.

int compareTo(String s)

Eg:

String s1=”A123”;

String s2=”C123”;

String s3=”A123”;

It returns,

<0 if s1 is less than s2 and the output of

S1.compareTo(s2) is **-2**

>0 if s1 is greater than s2 and the output of

s2.compareTo(s1) is **2**

=0 if s1 is equal to s2 and he output of

s1.compareTo(s3) is **0**

Character Extraction:

The String class provides a number of ways in which characters can be extracted from a String object.There are many methods used to extract characters from a String.The important method is **charAt()**

charAt():

This method is used to extract a single character from a String.The return type of this method is char.

char charAt(int where)

Here, where is the index of the character.

Eg;

String s=”ABCDEF”;

s.charAt(2);

where the output is **C**,since at location 2 there is character **C**.

Modifying a String:

There are many ways of modifying a String.Mostly used modifying method is substring().

substring():

This method is used to return a substring of the given String starting from the startindex of the given method.

The return type of this method is also a String.

This method is overloaded.

(i)String substring(int startindex)

It returns a substring starting from startindex to the end of the String.

Eg:

String s=”Learn from the Basic Concepts”;

012345---posions of characters in the String

The output of

s.substring(5) is **from the Basic Concepts**

(ii)String substring(int startindex,int endindex):

It returns a substring starting from startindex upto the end of the end index without including endindex.

Eg:

String s=”Learn from the Basic Concepts”;

0123456789

s.substring(5,9) is **from**

In this character in the location of 9 will not be included.

StringBuffer: It is a final class in java.lang package.

A StringBuffer is mutable,which means that which means that it is not fixed and the contents of a StringBuffer may change.It is dynamic in nature,where we can insert any character at any location.

A StringBuffer class stores an additional memory for 16 characters.

There are some methods which are available only in StringBuffer class only.Some of the important methods are:

capacity():

This method is used to return than memory capacity of a StringBuffer.The return type of this method is **int.**

Eg:

StringBuffer sb1=new StringBuffer();

StringBuffer sb2=new StringBuffer(“ABCDE”);

Sb1.length() is **0** and sb2.length() is **5**

sb1.capacity() is **16** where as

sb2.capacity is **21**

append():

This method is similar to **concat()** method in String class.This method is used to join any string to the StringBuffer.

The return type of this method is StringBuffer.

StringBuffer append(String s)

Eg:

StringBuffer sb=new StringBuffer(“Learn”);

Sb.append(“Basics”);

Where the output is **LearnBasics**

reverse():

This method is used to reverse the characters of a given StringBuffer.The return type of this method is again a StringBuffer.

StringBuffer reverse()

Eg:

StringBuffer sb=new StringBuffer(“ABCDEF”);

sb.reverse();

Where the output is **FEDCBA**