String functions

String are immutable objects

indexOf(substr)	It finds the position of the first occurrence of given substring	String s1=”some data” S1.indexOf(“data”) -→5
toLowerCase()	It converts the string into lowercase	String s1=”ABCtest” S1.toLowerCase() -→abctest
toUpperCase()	It converts the string into uppercase	String s1=”ABCtest” S1.toUpperCase() -→ABCTEST
contains(substr)	It returns true, if the substring exists otherwise false	String s1=”some data” S1.contains(“data”)-→true
split(delimiter)	It breaks the string into parts, at the given delimiter	String s1=”xxx,yyy,zzz-rrr” S1.split(“,”)-→ [“xxx”,”yyy”,”zzz-rrr”] S1.split(“-“) [“xxx,yyy,zzz”,”rrr”]
join(delimiter, arr of strings)	It will combine all the values from arr , separated by delimiter	String s1=”xxx,yyy,zzz-rrr” String[] arr=S1.split(“,”) String s2=String.*join*(":", arr); xxx:yyy:zzz-rrr
startswith(substr)	It returns true, if string starts with given substr, false otherwise	String s1=”Happy life” S1.startsWith(“Ha”)-→ true S1.startsWith(“ab”)-→false
endswith(substr)	It returns true, if string ends with given substr, false otherwise	String s1=”Happy life” S1.endsWith(“fe”)-→ true S1.endsWith(“ab”)-→false
charAt(i)	It retrieves the character at given index position	String s1=”Happy”; S1.charAt(1)-→a
matches(regexpression)	It checks whether the given regular expression matches the string or not	String s1=”Happy Life” S1.matches(“.*fe$”) -------true String str="rain in SPAIN in plain"; System.***out***.println("matches:"+str.matches(".*a.*n.*")); ----true
equals(Object ob)	It checks whether the contents of 2 string are same or not, returns true, if the contents are same, false othertwise	String s1=”test” String s2=new String(“test”) S1.equals(s2) -→ true
compareTo(Object ob)	It returns -ve value if s1<s2, 0 if s1==s2; +ve number otherwise	String s1=”aaaa”; String s2=”AAAA”; s1.compareTo(s2)-→+ve number
isBlank()	This method is used to check whether	String str = "testing"; boolean r = str.isBlank(); System.out.println(r);

	the string is blank or not. It returns **true** if the string is empty or contains only white space codepoints, otherwise **false**	str = ""; r = str.isBlank(); System.out.println(r);
lines()	It breaks the line into multiple lines at \n character position	String str = "testing \n is a \r technical \n portal"; Stream<String> lines = str.lines(); Output : testing is a technical portal
strip() stripLeading() stripTrailing()	This method is used to remove all the leading and trailing spaces from a string. If you want to remove only leading spaces then use the stripLeading() me thod and for trailing, spaces use the stripTrailing() me thod	String str1=" hello "; System.***out***.println("using strip "+str1.strip()); System.***out***.println("using stripleading "+str1.stripLeading()); System.***out***.println("using stripTrailing "+str1.stripTrailing());
String repeat(int count)	This method is used to repeat a string at the specified time. It returns a string whose value is the concatenation of this string repeated specified times.	String s1=”123” S1.repeat(3) Output 123123123

If you need a string which is changing frequently, the use StringBuffer or StringBuilder

StringBuilder sb=new StringBuilder(“xxxx”);

StringBuffer sf=new StringBuffer(“aaa”);

StringBuilder class functions are not synchronized, hence while changing data, it does not lock the object, Hence this class is suitable in Single threaded programming.

StringBuffer class functions are synchronized, hence while changing data, it puts lock on the object, Hence this class is suitable in Multithreaded programming.

append()-→ it appends the given string in the original string

StringBuilder sb=new StringBuilder("test")

sb.append(" data")

System.out.println(sb); ////test data

In both these classes, we have functions for deleting portion of the string, inserting portion of the string

Collection class hierarchy

List --→Vector, ArrayList, LinkedList

1. It is ordered collection, it means it stores the data in the same sequence, in which it is entered.
2. Since it is ordered, we can retrieve data by using index.
3. Since we can use index, we can retrieve data randomly.
4. Duplicate values are allowed

Set

1. It is unordered collection, it means it stores the data in the different way than the sequence in which it is entered.
2. Since it is unordered, we cannot retrieve data by using index.
3. Since we cannot use index, we cannot retrieve data randomly.
4. Duplicate values are not allowed

Map

1. It is Ordered collection, it maintains the sequence of keys, in the same order in which it is inserted
2. The data is store as key→ value pair
3. Keys will help us to retrieve data faster and randomly
4. Keys has to be unique, value can be duplicated.