COMPUTER APPLICATIONS

STRING HANDLINGS

FUNCTION NAME	DESCRIPTION	SYNTAX	EXAMPLE	NOTE
int length()	Returns the no.of characters in a string.	int <var> = <String.length()>;	String s1 = “comp”; int s2 = s1.length(); O/P : s2 = 4	It starts from 1(one).
char charAt (int index)	Returns the character present at the given index no.	char <var> = <String.charAt(int index)>;	String s1 = “comp”; char c = s1.charAt(3); O/P : c = p	It starts from 0(zero). If the index is more than length of a string then, O/P - String out of bounce.
String replace(char old_char, char new_char)	Replaces a character or a string by another character or a string.	String <var> = <String.replace(old ch_str, new ch_str)	String s1 = “comp app”; String s2 = s1.replace(‘c’,’k’); String s3 = s1.replace(“app”,”science”); O/P : s2 = ko	It will replace the character everywhere. Eg : MXDXM becomes MADAM when X is replaced by A.
boolean endsWith(String suffix)	Checks if the string ends with the specified suffix.	Boolean <var> =<String.endsWith(Str2)>;	String s1 = “this comp”; boolean b1=s1.endsWith(“this”); boolean b2 = s1.endsWith(“comp”); O/P : b1 = false b2 = true	Returns true if ends with the specified suffix or else false.
boolean startsWith(String prefix)	Checks if the string starts with the specified prefix.	Boolean <var> =<String.startsWith(Str2)>;	String s1 = “this comp”; boolean b1=s1.startsWith(“this”); boolean b2 = s1startsWith(“comp”); O/P : b1 = true b2 = false	Returns true if starts with the specified prefix or else false.

int compareTo(String s)	Compares 2 strings lexicographically. If s1 is string 1 and s2 is string 2, the value is 0 if both the strings are lexicographically equal. The value is less than 0 if s1<s2 and more than 0 if s1>s2.	int <var> = <String1.compareTo(String2)>;	s1=”computer app”;
			s2=”computer app”;
			s3=”app comp”;
			r1=s1.compareTo(s2);
			r2=s2.compareTo(s3);
			r3=s3.compareTo(s1);
			O/P : r1 = 0
			r2 = 2
			r3 = -2
				During comparison, the control keeps on comparing the corresponding characters of both strings. Wherever the corresponding characters are found different, the comparison will terminate resulting in a difference of their ASCII codes.

String toLowerCase()	Converts all the characters in the string to lowercase.	String <var> = <String.toLowerCase()>;	String s1=”COMPUTER”; String s2=s1.toLowerCase(); O/P :s2 = computer	The character which is a special character or is already a lower case, then it will remain same.
String toUpperCase()	Converts all the characters in the string to uppercase.	String <var> = <String.toUpperCase()>;	String s1=”computer”; String s2=s1.toUpperCase(); O/P :s2 = COMPUTER	The character which is a special character or is already uppercase, then it will remain same.

String substring(int begin_Index)	Returns a new string i.e a substring of the given string. The substring begins with a character at a specified index and extends to the end of the given string.	String <var> = <String.substring(int begin_Index)>;	String s1=”This is a computer”; String s2 = s1.substring(6); O/P: s2 = s a computer	
String substring(int begin_Index, int end_Index)	Returns a new string i.e a substring of the given string.. The substring begins with the character at a specified index and extends upto the index of the given string.	String <var> = <String.substring(int begin_Index, int end_Index)>;	String str = “This is a computer”; String s1=str.substring(2,12); O/P: s1 = is is a co	Starts from 0. Does not print the character present at end_Index.
int indexOf(char c)	Opposite of charAt(). Returns the index of the first occurence of the specified character of a string.	int <var> = <String.indexOf(char c)>;	String s1 = “book”; int r1 = s1.indexOf(‘o’); O/P: r1 = 1	Returns -1 if the character is not there.
int indexOf(String s)	Returns the index of the first occurence of the specified substring of a string.	int <var> = <String.indexOf(String s)>;	String s1 = “welcome to java”; int r1 = s1.indexOf(“java”); O/P: r1 = 11	
int lastIndexOf(char c)	Returns the index of the last occurence of the specified character of a string.	int <var> = <String.indexOf(char c)>;	String s1 = “book”; int r1 = s1.lastIndexOf(‘o’); O/P: r1 = 2	
int lastIndexOf(String s)	Returns the index of the last occurrence of the specified substring of a string.	int <var> = <String.indexOf(String s)>;	String s1 = “java is java”; int r1 = s1.lastIndexOf(“java”); O/P: r1 = 8	
String trim()	Function to remove the leading and trailing blanks from a string.	String <var> = <String.trim()>;	String s1=” comp “; String s2=s1.trim(); O/P: s2=comp	The blanks which are between the words will remain unchanged.
String concat(String s)	Concatenates(joins) the given string with the other string.	String <var> = <String.concat(String s)>;	String s1 = “comp”; String s = s1.concat(“app”); O/P: s=compapp	
boolean equals(String s)	Compares the given string with the other string, to check whether they are identical or not. Returns true if identical(same).	boolean <var> = <String.equals(String s)>;	String s1 = “computer”; String s2 = “app”; boolean b =s1.equals(s2); O/P:false	It is case-sensitive.
boolean equalsIgnoreCase(String s)	Compares the given string with the other string but ignores the case, and checks whether they are identical or not. Returns true if identical(same).	boolean <var> = <String.equalsIgnoreCase(String s)>;	String s1=”cOmPUter”; String s2=”COMputeR”; boolean b=s1.equalsIngoreCase(s2); O/P:true	The == sign is never used for checking equality of strings because comparing 2 strings is the same as comparing 2 character arrays which are non-primitive data.
int copmareToIgnoreCase (String s)	Its modified of compareTo(). The only difference is that compareToIgnoreCase() function compares 2 strings after ignoring the cases.	int <var> = <Strring.compareToIgnoreCase(String s)>;	String s1 = “COMP”; String s2 = “comp”; int n = s1.compareToIgnoreCase (s2); O/P:0	
int valueOf(String s)	Function to convert string into primitive datatype of the corresponding wrapper class. This function is available under each wrapper class of java.lang package.	<Primitive Datatype> <var> = <WrapperClass>.valueOf(String s);	String s1 = “124.67”; double n = Double.valueOf(s1); O/P: n = 124.67	This fn() will only convert a string into primitive datatype, provided in the form of number or digits. In case, an alphanumeric string is used as an argument then it will throw a NmuberFormatException