CSE 215: Programming Language II Lab

Lab – 4

Mathematical and String functions in Java

Objective:

- To learn about Mathematical Functions, Characters and Strings
- To learn about using Mathematical functions in Java program

TABLE 4.1 Trigonometric Methods in the Math Class

Method	*Description*
`sin(radians)`	Returns the trigonometric sine of an angle in radians.
`cos(radians)`	Returns the trigonometric cosine of an angle in radians.
`tan(radians)`	Returns the trigonometric tangent of an angle in radians
`toRadians(degree)`	Returns the angle in radians for the angle in degree.
`toDegree(radians)`	Returns the angle in degrees for the angle in radians.
`asin(a)`	Returns the angle in radians for the inverse of sine.
`acos(a)`	Returns the angle in radians for the inverse of cosine.
`atan(a)`	Returns the angle in radians for the inverse of tangent.

TABLE 4.2 Exponent Methods in the Math Class

Method	*Description*
`exp(x)`	Returns e raised to power of x (e^x).
`log(x)`	Returns the natural logarithm of x ($\ln(x) = \log_e(x)$).
`log10(x)`	Returns the base 10 logarithm of x ($\log_{10}(x)$).
`pow(a, b)`	Returns a raised to the power of b (a^b).
`sqrt(x)`	Returns the square root of x ($\sqrt{x}$) for x >= 0.

TABLE 4.3 Rounding Methods in the Math Class

Method	*Description*
`ceil(x)`	x is rounded up to its nearest integer. This integer is returned as a double value.
`floor(x)`	x is rounded down to its nearest integer. This integer is returned as a double value.
`rint(x)`	x is rounded up to its nearest integer. If x is equally close to two integers, the even one is returned as a double value
`round(x)`	Returns (int)Math.floor(x + 0.5) if x is a float and returns (long)Math.floor(x + 0.5) if x is a double.

TABLE 4.8 Comparison Methods for `String` Objects

Method	*Description*
`equals(s1)`	Returns true if this string is equal to string `s1`.
`equalsIgnoreCase(s1)`	Returns true if this string is equal to string `s1`; it is case insensitive.
`compareTo(s1)`	Returns an integer greater than 0, equal to 0, or less than 0 to indicate whether this string is greater than, equal to, or less than `s1`.
`compareToIgnoreCase(s1)`	Same as `compareTo` except that the comparison is case insensitive.
`startsWith(prefix)`	Returns true if this string starts with the specified prefix.
`endsWith(suffix)`	Returns true if this string ends with the specified suffix.
`contains(s1)`	Returns true if `s1` is a substring in this string.

TABLE 4.9 The `String` class contains the methods for obtaining substrings.

Method	*Description*
`substring(beginIndex)`	Returns this string's substring that begins with the character at the specified `beginIndex` and extends to the end of the string, as shown in Figure 4.2.
`substring(beginIndex, endIndex)`	Returns this string's substring that begins at the specified `beginIndex` and extends to the character at index `endIndex` – 1, as shown in Figure 4.2. Note that the character at `endIndex` is not part of the substring.

TABLE 4.10 The `String` class contains the methods for finding substrings.

Method	*Description*
`index(ch)`	Returns the index of the first occurrence of `ch` in the string. Returns `-1` if not matched.
`indexOf(ch, fromIndex)`	Returns the index of the first occurrence of `ch` after `fromIndex` in the string. Returns `-1` if not matched.
`indexOf(s)`	Returns the index of the first occurrence of string `s` in this string. Returns `-1` if not matched.
`indexOf(s, fromIndex)`	Returns the index of the first occurrence of string `s` in this string after `fromIndex`. Returns `-1` if not matched.
`lastIndexOf(ch)`	Returns the index of the last occurrence of `ch` in the string. Returns `-1` if not matched.
`lastIndexOf(ch, fromIndex)`	Returns the index of the last occurrence of `ch` before `fromIndex` in this string. Returns `-1` if not matched.
`lastIndexOf(s)`	Returns the index of the last occurrence of string `s`. Returns `-1` if not matched.
`lastIndexOf(s, fromIndex)`	Returns the index of the last occurrence of string `s` before `fromIndex`. Returns `-1` if not matched.

Task:

1) Compute the area of a hexagon using the following formula:

$$Area = \frac{6 \times s^2}{4 \times \tan\left(\frac{\pi}{6}\right)}$$

Here, s is the side of a hexagon.

- Write a program that prompts the user to enter the side of a hexagon and display its area. If the side length is invalid (<0), display a message showing "The area can not be computed due to invalid side length". Below, a sample run is given:

```
Enter the side: 5.5 Enter
The area of the hexagon is 78.59
```

2) Given a string s1 and another string s2, check if string s1 starts or ends with string s2.

   For example, for s1 = "apple" and s2="app" , display the following output:
   "The string "apple" starts with "app".

   for s1 = "apple" and s2="ple" , display the following output:
   "The string "apple" ends with "ple".

   for s1 = "apple" and s2="ppl" , display the following output:
   "The string "apple" does not start or end with "ppl".

3) Given a string, find the index of the second last occurrence of any vowel. For example, if your string is "corresponding" , then return the index of 'o' which is the second last vowel (after 'i').

4) * (Take Home Task) Modify task 2 to check if two strings s1 and s2 are equal, or equal after ignoring cases, or if s1 starts or ends with s2 (and vice versa), or if s2 is substring of s2 (or vice versa). If none of these happen, display an appropriate message.