Lab 06

Mathematical, Characters and String Methods

Objective:

This lab will give you practical implementation of predefined mathematical, characters and String methods.

Activity Outcomes:

On completion of this lab student will be able

- Use pre-defined **`Math`** class methods
- Use pre-defined **`Character`** class methods
- Use pre-defined **`String`** class methods

Instructor Note:

As a pre-lab activity, read Chapter 06 and Chapter 16 from the text book "Java How to Program, Deitel, P. & Deitel, H., Prentice Hall, 2019".

1) Useful Concepts

Common Mathematical Functions

Java provides many useful methods in the Math class for performing common mathe-matical functions.

Trigonometric Methods

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.

Exponent Methods

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.

The Rounding Methods

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.

The `min`, `max`, and `abs` Methods

The **`min`** and **`max`** methods return the minimum and maximum numbers of two numbers (**`int`**, **`long`**, **`float`**, or **`double`**). The **`abs`** method returns the absolute value of the number (**`int`**, **`long`**, **`float`**, or **`double`**)

The `random` Method

This method generates a random **`double`** value greater than or equal to 0.0 and less than 1.0 (**`0 <= Math.random() < 1.0`**)

`(int)(Math.random() * 10)` Returns a random integer between 0 and 9.

`50 + (int)(Math.random() * 50)` Returns a random integer between 50 and 99.

General Form,

`a + Math.random() * b`	Returns a random number between a and a + b, excluding a + b.

Methods in the Character Class

Method	*Description*
`isDigit(ch)`	Returns true if the specified character is a digit.
`isLetter(ch)`	Returns true if the specified character is a letter.
`isLetterOfDigit(ch)`	Returns true if the specified character is a letter or digit.
`isLowerCase(ch)`	Returns true if the specified character is a lowercase letter.
`isUpperCase(ch)`	Returns true if the specified character is an uppercase letter.
`toLowerCase(ch)`	Returns the lowercase of the specified character.
`toUpperCase(ch)`	Returns the uppercase of the specified character.

The String Type

A string is a sequence of characters. Class **`String`** is used to represent strings in Java.

Declaring and Initializing String Variables

The String type is not a primitive type. It is known as a reference type

```
String message = "Welcome to Java";
```

Any Java class can be used as a reference type for a variable. The variable declared by a reference type is known as a reference variable that references an object. Here, **`message`** is a reference variable that references a string object with contents **`Welcome to Java`**

Simple Methods for `String` Class

Method	*Description*
`length()`	Returns the number of characters in this string.
`charAt(index)`	Returns the character at the specified index from this string.
`concat(s1)`	Returns a new string that concatenates this string with string s1.
`toUpperCase()`	Returns a new string with all letters in uppercase.
`toLowerCase()`	Returns a new string with all letters in lowercase
`trim()`	Returns a new string with whitespace characters trimmed on both sides.

Reading a String from the Console

To read a string from the console, invoke the **`next()`** method on a **`Scanner`** object. For example, the following code reads three strings from the keyboard

```
Scanner input = new Scanner(System.in);
System.out.print("Enter three words separated by spaces: ");
String s1 = input.next();
String s2 = input.next();
String s3 = input.next();
```

```
System.out.println("s1 is " + s1);
System.out.println("s2 is " + s2);
System.out.println("s3 is " + s3);
```

```
Enter three words separated by spaces: Welcome to Java
s1 is Welcome
s2 is to
s3 is Java
```

You can use the **`nextLine()`** method to read an entire line of text. The **`nextLine()`** method reads a string that ends with the Enter key pressed

```
Scanner input = new Scanner(System.in);
System.out.println("Enter a line: ");
String s = input.nextLine();
System.out.println("The line entered is " + s);
```

```
Enter a line: Welcome to Java
The line entered is Welcome to Java
```

Comparing Strings

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.

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.

Finding a Character or a Substring in a String

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.

2) Solved Lab Activites

Sr.No	*Allocated Time*	*Level of Complexity*	*CLO Mapping*
Activity 1	***15 mins***	***Midum***	***CLO-5***
Activity 2	***15 mins***	***Midum***	***CLO-5***
Activity 3	***15 mins***	***Midum***	***CLO-5***
Activity 4	***15 mins***	***Midum***	***CLO-5***

Activity-1:

This activity demonstrate usage of Math class functions. Following program prompts the user to enter the x- and y-coordinates of the three corner points in a triangle and then displays the three angles.

Solution:

```
public class Activity1{
      public static void main(String[] args) {
            Scanner input = new Scanner(System.in);
             // Prompt the user to enter three points
            System.out.print("Enter three points: ");
            double x1 = input.nextDouble();
            double y1 = input.nextDouble();
            double x2 = input.nextDouble();
            double y2 = input.nextDouble();
            double x3 = input.nextDouble();
            double y3 = input.nextDouble();
             // Compute three sides
      double a = Math.sqrt((x2 - x3) * (x2 - x3) + (y2 - y3) * (y2 -
      y3));
            double b = Math.sqrt((x1 - x3) * (x1 - x3) + (y1 - y3) *
      (y1 - y3));
            double c = Math.sqrt((x1 - x2) * (x1 - x2) + (y1 - y2) *
      (y1 - y2));
            // Compute three angles
```

```
        double A = Math.toDegrees(Math.acos((a * a - b * b - c *
    c) / (-2 * b * c)));
        double B = Math.toDegrees(Math.acos((b * b - a * a - c *
    c) / (-2 * a * c)));
         double C = Math.toDegrees(Math.acos((c * c - b * b - a *
    a) / (-2 * a * b)));
        // Display results
        System.out.println("The three angles are " + Math.round(A
    * 100) / 100.0 + " " + Math.round(B * 100) / 100.0 + " " +
    Math.round(C * 100) / 100.0);
    }
}
```

Output

```
Enter three points: 1 1 6.5 1 6.5 2.5
The three angles are 15.26 90.0 74.74
```

Activity-2:

This activity illustrate useful methods of `Character` class.

Solution:

```
public class Activity2{
  public static void main(String[] args) {
    System.out.println("isDigit('a') is " + Character.isDigit('a'));
    System.out.println("isLetter('a') is " +
Character.isLetter('a'));
    System.out.println("isLowerCase('a') is "+
    Character.isLowerCase('a'));
    System.out.println("isUpperCase('a') is "+
    Character.isUpperCase('a'));
    System.out.println("toLowerCase('T') is "+
    Character.toLowerCase('T'));
    System.out.println("toUpperCase('q') is "+
     Character.toUpperCase('q'));
  }
}
```

Output

```
isDigit('a') is false
isLetter('a') is true
isLowerCase('a') is true
isUpperCase('a') is false
toLowerCase('T') is t
toUpperCase('q') is Q
```

Activity-3:

This acitivty uses compareTo() method to display two cities, entered by user, in alphabetical order.

Solution:

```
import java.util.Scanner;
public class Activity3 {
       public static void main(String[] args) {
            Scanner input = new Scanner(System.in);
             // Prompt the user to enter two cities
            System.out.print("Enter the first city: ");
            String city1 = input.nextLine();
            System.out.print("Enter the second city: ");
            String city2 = input.nextLine();
            if (city1.compareTo(city2) < 0)
                  System.out.println("The cities in alphabetical order
                  are " + city1 + " " + city2);16 else
                  System.out.println("The cities in alphabetical order
                  are " +city2 + " " + city1);
      }
}
```

Output

```
Enter the first city: New York
Enter the second city: Boston
The cities in alphabetical order are Boston New York
```

Activity-4:

This program illustrate how various String methods work.

Solution:

```
public class Activity4{
      public static void main(String [] args){
            String sentence;
            String str1;
```

```
        String str2;
        String str3;
        int index;
        sentence = "Now is the time for the birthday party";
        System.out.println("sentence = \"" + sentence + "\"");
        System.out.println("The length of sentence = "+
    sentence.length());
        System.out.println("The character at index 16 in "+
    "sentence = " + sentence.charAt(16));
        System.out.println("The index of first t in sentence = "+
    sentence.indexOf(' t'));
        System.out.println("The index of for in sentence = "+
    sentence.indexOf("for"));
        System.out.println("sentence.substring(0, 6) = \""+
    sentence.substring(0, 6) + "\"");
        System.out.println("sentence.substring(7, 12) = \""+
    sentence.substring(7, 12) + "\"");
        System.out.println("sentence.substring(7, 22) = \""+
    sentence.substring(7, 22) + "\"");
        System.out.println("sentence.substring(4, 10) = \""+
    sentence.substri ng(4, 10) + "\"");
        str1 = sentence.substring(0, 8);
        System.out.println("str1 = \"" + str1 + "\"");
        str2 = sentence.sub string(2, 12);
        System.out.println("str2 = \"" + str2 + "\"");
        System.out.println("sentence in uppercase = \""+ sent
    ence.toUpperCase() + "\"");
        index = sentence.indexOf("birthday");
        str1 = sentence.substring(index, index + 14);
        S ystem.out.println("str1 = \"" + str1 + "\"");
        System.out.println("sentence.replace('t', 'T' ) = \""+
    sentence.replace('t', 'T') + "\"");
    }
}
```

Output

```
sentence = "Now is the time for the birthday party"
The length of sentence = 38
The character at index 16 in sentence = f
The i ndex of first t in sentence = 7
The index of for in sentence = 16
sentence.substring(0, 6) = "Now is"
sentence.substring(7 , 12) = "the t"
sentence.substring(7, 22) = "the time for th"
sentence.substring(4, 10) = "is the"
str1 = "Now is t"
str2 = "w is the t"
sentence in uppercase = "NOW IS THE TIME FOR THE BIRTHDAY PARTY"
str1 = "birthday party"
sentence.replace('t' , 'T') = "Now is The Time for The birThday parTy"
```

3) Graded Lab Tasks

Note: The instructor can design graded lab activities according to the level of difficult and complexity of the solved lab activities. The lab tasks assigned by the instructor should be evaluated in the same lab.

Lab Task 1

The great circle distance is the distance between two points on the surface of a sphere. Let (x1, y1) and (x2, y2) be the geographi-cal latitude and longitude of two points. The great circle distance between the two points can be computed using the following formula:

$$d = radius \times \arccos(\sin(x_1) \times \sin(x_2) + \cos(x_1) \times \cos(x_2) \times \cos(y_1 - y_2))$$

Write a program that prompts the user to enter the latitude and longitude of two points on the earth in degrees and displays its great circle distance. The average earth radius is 6,371.01 km. Note that you need to convert the degrees into radians using the Math.toRadians method since the Java trigonometric methods use radians. The latitude and longitude degrees in the formula are for north and west. Use negative to indicate south and east degrees. Here is a sample run:

```
Enter point 1 (latitude and longitude) in degrees: 39.55, -116.25
Enter point 2 (latitude and longitude) in degrees: 41.5, 87.37
The distance between the two points is 10691.79183231593 km
```

Lab Task 2

a) *Write a program that receives an ASCII code (an integer between 0 and 127) and displays its character. Here is a sample run*

```
Enter an ASCII code: 69
The character for ASCII code 69 is E
```

b) *Write a program that receives a character and displays its Unicode. Here is a sample run:*

```
Enter a character: E
The Unicode for the character E is 69
```

Lab Task 3

a) *Write a program that prompts the user to enter an integer between 0 and 15 and displays its corresponding hex number. Here are some sample runs:*

```
Enter a decimal value (0 to 15): 11
The hex value is B
Enter a decimal value (0 to 15): 5
The hex value is 5
Enter a decimal value (0 to 15): 31
31 is an invalid input
```

b) *Write a program that prompts the user to enter a hex digit and displays its corresponding binary number. Here is a sample run*

```
Enter a hex digit: B
The binary value is 1011
Enter a hex digit: G
G is an invalid input
```

Lab Task 4

Write a program that displays a random uppercase letter using the ***`Math.random()`*** *method.*

Lab Task 5

Write a program that checks whether a string is a palindrome. A string is a palindrome if it reads the same forward and backward. The words "mom," "dad," and "noon," for instance, are all palindromes. Sample run:

```
Enter a string: noon
noon is a palindrome
Enter a string: moon
moon is not a palindrome
```

Lab Task 6

Given a string consisting of exactly two words separated by a space. Print a new string with the first and second word positions swapped (the second word is printed first). ***This task should not use loops and if.***

Sample Run:

```
Input: Hello, world!
Correct Answer: world! Hello,
```

Lab Task 7

Given a string that may or may not contain a letter of interest. Print the index location of the first and last appearance of ***f****. If the letter* ***f*** *occurs only once, then output its index. If the letter* ***f*** *does not occur, then do not print anything.*

```
Input: office
Correct Answer: 1 2
```

Lab Task 8

*Given a string in which the letter **h** occurs at least twice. Remove from that string the first and the last occurrence of the letter **h**, as well as all the characters between them.*

Sample Run:

```
Input: In the hole in the ground there lived a hobbit
Correct Answer: In tobbit
```

Lab Task 9

Given a string. Replace every occurrence of the letter h by the letter H, except for the first and the last ones.

```
Input: In the hole in the ground there lived a hobbit
Correct Answer: In the Hole in tHe ground tHere lived a hobbit
```

Lab Task 10

You are given a string.
In the first line, print the third character of this string.
In the second line, print the second to last character of this string.
In the third line, print the first five characters of this string.
In the fourth line, print all but the last two characters of this string.
In the fifth line, print all the characters of this string with even indices (remember indexing starts at 0, so the characters are displayed starting with the first).
In the sixth line, print all the characters of this string with odd indices (i.e. starting with the second character in the string).
In the seventh line, print all the characters of the string in reverse order.
In the eighth line, print every second character of the string in reverse order, starting from the last one.
In the ninth line, print the length of the given string.
Sample Run:

```
Input: Hello
l
l
Hello
Hel
Hlo
el
olleH
olH
5
```