RELEASE 01:
Chapter 04 & 05.

Exercises explanation document

by María Díaz-Rozas Pantoja

1º GSDAW-Y

2023-11-05

CONTENIDO

CHAPTER 04. **3**

EXERCISE 6: **3**

CHAPTER 05: **4**

EXERCISE 2. **4**
EXERCISE 6. **6**
EXERCISE 7. **8**

CHAPTER 04.

In this chapter 04 to Methods and Testing, I did the following exercise:

Exercise 6:

- **Create a new program called *Multadd.java*.**
- **Write a method called multadd that takes three doubles as parameters and returns a * b + c.**
- **Write a main method that tests multadd by invoking it with a few simple parameters, like 1.0, 2.0, 3.0.**
- **Also, in main, use multadd to compute the following values:**

$$\sin\frac{\pi}{4} + \frac{\cos\frac{\pi}{4}}{2}$$

$$\log 10 + \log 20$$

- **Write a method called** `expSum` **that takes a double as a parameter and uses** `multadd` **to calculate:**

$$x\,e^{-x} + \sqrt{1 - e^{-x}}$$

In this Exercise I defined the public class named *Multadd.java,* I created the methods named *multadd* and *expsum.*

In multadd I declared that when it is invoked the method calculate the value for a*b+c, and return a double type value.

And when you invoke *expSum*, the method assigns the value to variables a, b and c. And then invoke the *multadd* to resolve the equation.

- I declared variables a, b and c to resolve: $x\,e^{-x} + \sqrt{1 - e^{-x}}$

  [a= x] [b=Math.exp(-x)] [Math.sqrt(1.0 - b)]

Note: Math.exp --> is a util by JAVA to resolve e (exponent).

Note: Math.sqrt -- > is a util by JAVA to resolve square root

In the main method:

1 -Invoke *multadd* with values: 1.0, 2.0, 3.0

2- Declare a1, b1, c1 to resolve cos and sin.

3- Invoke multadd again with a1, b1, c1.

4- I declared the variables to a2, b2 and c2 and resolve the logarithms with the help of util `Math.log`.

5- And invoke *multadd* again for this.

6- Finally, the program prompted the user to enter a value for x.

7- And resolve the equation with method *expSum*

In this exercise I didn't have too many problems in this exercise.

CHAPTER 05:

In this chapter 05 to Conditionals and Logic, I did the following exercise:

Exercise 2.

- **The goal of this exercise is to improve the last program: *Guess My Number* game.**

Now I create a new file named *MyGuess2.java* and added some improvements and changes.

In first, I added the Scanner.in to ask the user its name.

```
System.out.println("Hello, what is your name?");
String name=in.nextLine();
System.out.println("Well, "+name+" I'm MAGICJAVA. ");
System.out.println();
```

Then, I modify the program so that it tells the user if the guess is too high or too low and then prompts the user for another guess. For this I created a method named *mirror*

```
    public static int mirror (int suppose, int number) {
        if(suppose<number){
         System.out.println("The guess is too LOW");}

         else if(suppose>number){
             System.out.println("The guess is too HIGH");
         }
         return 0;
     }
```

Before that, I reduced the number of attempts so that it was equal to 3.

```
int k = 3;
```

And I match up the conditions to make it cleaner like this:

```
 else if (guess > 51 || guess<0 || guess == 0 ) {
        System.out.println();
        System.out.println( "Type a number between 1 and 50 (including
both)");
        System.out.println( "PLEASE, TRY AGAIN!");
        System.out.println("Tries left: "+ k + "/"+k);
        }
```

Now if you run it and type the value to convert, the program print and you don´t guess:

Hello, what is your name?
Maria
Well, Maria I'm MAGICJAVA.

And I'm thinking of a number between 1 and 50 (including both), CAN YOU GUESS IT?
You have 3 tries to guess.
Insert a number: 56
Type a number between 1 and 50 (including both)
PLEASE, TRY AGAIN!
Tries left: 3/3
Insert a number:
Insert a number: 0
Type a number between 1 and 50 (including both)
PLEASE, TRY AGAIN!
Tries left: 3/3
Insert a number:
Insert a number: -6
Type a number between 1 and 50 (including both)
PLEASE, TRY AGAIN!
Tries left: 3/3
Insert a number: 10
The guess is too LOW
Tries left: 2/3
Insert a number: 46
The guess is too HIGH
Tries left: 1/3
Insert a number: 5

GAME OVER!

Your number is: 1
I was thinking of: 14
You were off by: -13

Don't worry, maybe next time you will guess!

If you guess and win:

….
You have 2 tries left.
Insert a number: 27
CONGRATULATION!!
Your guess is: 27
My number is: 27

Exercise 6.

- **The goal of this exercise is creating a program named *Quadratic.java*.**

This is the one that cost me the most to do, because I didn't know how to create exceptions.

Write a program named Quadratic.java that finds the roots of ax2 + bx + c = 0 using the quadratic formula:

$$x = \frac{-b \pm \sqrt{b^2 - 4ca}}{2un}$$

First try to create the exceptions with “Validating Input”, but I saw that was a mistake! since I realized that this was just to avoid entering different characters by the user, and this is not what the exercise asked of me. Finally, I create a code simpler.

This program prompts the user to enter values for a, b, and c, calculates the discriminant, and then checks the value of the discriminant to determine the number of solutions to the quadratic equation.

- If the discriminant is negative, it displays a message indicating that the equation has no real roots.
- If the discriminant is zero, calculate and show the only solution to the equation.
- If the discriminant is positive, calculate and show the two solutions of the equation.

The program uses the Math.sqrt() method to calculate the square root and handles division by zero using double precision for calculations.

Finally, I added the format validation:

```
if (!in.hasNextInt()) {
            String word = in.next();
            System.err.println("ERROR IMPUT: "+word + " is not a
number");
            return;}
```

But this no to be work really well, I really don't know how to use other word return to main when get ERROR don't finish the program. But for the moment … this is what I have achieved.

Now when you execute the program:

Ejemplo 1: Ingrese el valor de a: 1, Ingrese el valor de b: -4, Ingrese el valor de c: 4. El discriminante es 0. La ecuación tiene una solución: 2.0

Enter the value of a: 1
Enter the value of b: -4
Enter the value of c: 4
The equation has one root: 2.0

Ejemplo 2: Ingrese el valor de a: 12, Ingrese el valor de b: 25, Ingrese el valor de c: 6. El discriminante es d>0. La ecuación tiene dos soluciones: -0.27676834372142417 and 1.8065649896119094

Enter the value of a: 12
Enter the value of b: 25
Enter the value of c: 6
The equation has two roots: -0.27676834372142417 and -1.8065649896119094

Ejemplo 3: Ingrese el valor de a: 2, Ingrese el valor de b: 2, Ingrese el valor de c: 2. El discriminante es d<0. La ecuación no tiene solución.

Enter the value of a: 2
Enter the value of b: 2
Enter the value of c: 2
The equation has no real roots.

Exercise 7.

Write a program named Triangle.java that inputs three integers, and then outputs whether you can (or cannot) form a triangle from the given lengths. Reuse your code from the previous exercise to validate the inputs. Display an error if any of the lengths are negative or zero.

In this exercise I created different methods to resolve the problem.

In first, a Boolean method named: *validateLengths*

```
public static boolean validateLengths(int side1, int side2, int side3) {
        return side1 > 0 && side2 > 0 && side3 > 0; }
```

In this method the program compares if the value to all sides is true (bigger than 0).

Then, created a boolean method named *isTriangle* to compare the all sides with each others. Because, in every triangle the sum of the lengths of any two sides is always greater than the length of the remaining side.

```
public static boolean isTriangle(int side1, int side2, int side3) {
        return side1 + side2 > side3 && side1 + side3 > side2 && side2 +
side3 > side1; }
```

This logic is named “triangular inequality”.

> *NOTE: TO KNOW THIS RULE I HAD TO SEARCH ON THE INTERNET, YOU CAN SEE THE FOLLOWING LINK:*
>
> *HTTPS://ES.WIKIPEDIA.ORG/WIKI/DESIGUALDAD_TRIANGULAR*
>
> **THE TRIANGULAR INEQUALITY** IS A THEOREM IN GEOMETRY THAT STATES THAT IN ANY TRIANGLE, THE SUM OF THE LENGTHS OF TWO SIDES WILL ALWAYS BE GREATER THAN THE LENGTH OF THE THIRD SIDE. MATHEMATICALLY, IF A, B, AND C ARE THE LENGTHS OF THE SIDES OF A TRIANGLE, THEN IT IS TRUE THAT A + B > C, A + C > B, AND B + C > A. THIS INEQUALITY IS A FUNDAMENTAL PRINCIPLE IN TRIANGLE GEOMETRY AND IS USEFUL FOR DETERMINING IF A SET OF SEGMENTS CAN FORM A VALID TRIANGLE.

In conclusion, mathematically, for a triangle with sides of lengths a, b, and c, the triangular inequality can be written as: (a + b > c b + c > a c + a > b).

In my program, the code compares if the sum of side1 and side2 is greater than side3, and also if the sum of side1 and side3 is greater than side2, and finally if the sum of side2 and side3 is greater than side1.

If all these conditions are met, the result will be true, indicating that the lengths of the sides form a valid triangle. Otherwise, the result will be false, meaning that they do not meet the triangular inequality and therefore do not form a valid triangle.

If the *validateLenggths* returns true, proceed to call the method *isTriangulo.*

- ✓ And if this returns true, print the message:

  `"It is possible to form a triangle with the given lengths."`

- ✓ If this returns false, print the message:
  `"It is not possible to form a triangle with the given lengths."`

To finish, if the method *validateLengths* function returns false, the program printed the message

- ✓ `"Error: the lengths must be positive integers and different from zero”.`

This indicates that the side lengths do not meet the necessary requirements to form a triangle.

Finally, I added the format validation but I had the same problem as in the previous exercise.
