Introduction to Computer Programming (Java A)
Lab 4

[Objectives]

0. Learn how to use the *while* repetition statement to execute statements in a program repeatedly.
1. Learn how to use the *do…while, for* repetition statement to execute statements in a program.
2. Learn how to use the *switch* selection statements to choose among alternative actions.
3. Learn how to use the *break and continue* statements in a program.

[Exercises]

1. Write a program to print 9 x 9 multiplication table, by using the ***while*** repetition statement.

Notice: printf using %02d, "3" will be "03"; %2d, "3" will be " 3",with a blank space on the left.

Sample output:

```
1 * 1 = 1
1 * 2 = 2  2 * 2 = 4
1 * 3 = 3  2 * 3 = 6  3 * 3 = 9
1 * 4 = 4  2 * 4 = 8  3 * 4 = 12 4 * 4 = 16
1 * 5 = 5  2 * 5 = 10 3 * 5 = 15 4 * 5 = 20 5 * 5 = 25
1 * 6 = 6  2 * 6 = 12 3 * 6 = 18 4 * 6 = 24 5 * 6 = 30 6 * 6 = 36
1 * 7 = 7  2 * 7 = 14 3 * 7 = 21 4 * 7 = 28 5 * 7 = 35 6 * 7 = 42 7 * 7 = 49
1 * 8 = 8  2 * 8 = 16 3 * 8 = 24 4 * 8 = 32 5 * 8 = 40 6 * 8 = 48 7 * 8 = 56 8 * 8 = 64
1 * 9 = 9  2 * 9 = 18 3 * 9 = 27 4 * 9 = 36 5 * 9 = 45 6 * 9 = 54 7 * 9 = 63 8 * 9 = 72 9 * 9 = 81
```

2. Create a class called GuessingNumber. In the main method, you should generate a random integer magicNum between 0 and 9, then keep asking the user to input an integer between 0 and 9 until the input number is equal to the attribute magicNum. When the input number is greater than the attribute magicNum, the method should output "Too big!Please try again:".When the input number is less than the attribute magicNum, the method should output "Too small!Please try again:". Then the method wait for the user to input a new integer. When the input number is equal to the attribute magicNum, the method should output "Congratulations!" and terminate.

Sample code:

```
import java.util.Random;
public static void main(String[] args) {

    Random random = new Random();
    int magicNum = random.nextInt(10);
    int inputNum;
    Scanner sc = new Scanner(System.in);

    System.out.println("Please input an Integer in
{0,1,2,...,9}:");
    inputNum = sc.nextInt();

    while(                    ){// to finish it
        if (                  )// to finish it
          System.out.println("Too big!Please try again:");
       else
          System.out.println("Too small!Please try again:");
       inputNum = sc.nextInt();
    }

    System.out.println("Congratulations!");
    sc.close();
}
```

Sample output:

```
Please input an Integer in {0,1,2,...,9}:
3
Too small!Please try again:
5
Too small!Please try again:
7
Congratulations!
```

3. Calculate the value of π from the infinite series

$$\pi = 4 - \frac{4}{3} + \frac{4}{5} - \frac{4}{7} + \frac{4}{9} - \frac{4}{11} + \cdots$$

Input an integer ***n*** which represents the number of terms in the formula above. It is more precise when ***n*** is bigger. Use *do…while* or *while* repetition statements to compute the value of π.

Sample output:

```
Please input n:
10000
The estimatioin of Pi is 3.141498
```

Modify your program as follows:
Input a double value which represents a precision threshold. Your program should terminate when the difference between two successive iterations is smaller than the precision threshold. Print the value of π, and the iteration numbers.

Sample output:

```
Please input the precision:
0.0001
The estimatioin of Pi is 3.141547
It computed 19998 times
```

Tips: use Math.abs()

4. Rewrite exercise 3 above. Use *for* repetition statements to estimate the value of π, according to the specified number of iterations and precision threshold.

Think about this: when to use *for* and when to use *while*?

Calculate the value of π from the infinite series

$$\pi = 4 - \frac{4}{3} + \frac{4}{5} - \frac{4}{7} + \frac{4}{9} - \frac{4}{11} + \cdots$$

(1) Input an integer ***n,*** which represents the number of terms in the formula above. The estimated value is more precise when ***n*** is bigger.

(2) Input a double value, which presents a precision threshold. Your program should terminate when the difference between two successive iterations is smaller than the precision threshold. Print the value of π, and the number of iterations.

5. Rewrite exercise 2 in lab3. Use *switch* to calculate the GPA according to the following table.

Grade	GPA
100~90	4.0
89~80	3.0
79~70	2.0

69~60	1.0
59~0	0

Write a program to calculate the GPA of a student according to the method used by SUSTech. The user can input the credit and score of each course. The process should continue until the user inputs "-1". After receiving all inputs, the program outputs the final GPA of the student.

Think about this: when could *'if…else'* be replaced by *switch*?

Sample output

```
3 95
2 89
3 77
3 67
1 95
-1
final gpa is 2.6
```

6. There are 30 or 31 days in a month except February. There are 28 days in February in a common year, and 29 days in a leap year. Write a program to input year and month by command line and show the days of this month using *switch*.

A year is a leap year if it is:
(1) divisible by 4, but not divisible by 100;
(2) or divisible by 400;

Please use "*DaysofYearMonth*" as the class name and "*DaysofYearMonth.java*" as the file name.
The template code is given to you as follows:

```
public class DaysOfYearMonth {

    public static void main(String[] args) {
        int year = Integer.parseInt(args[0]);
        int month = Integer.parseInt(args[1]);
        String monthName = "";
        int days = 0;
        boolean isLeapYear = false;
        if ( /*fill in the checking case here */     ) {
            isLeapYear = true;
        } else {
            isLeapYear = false;
        }
        switch (month) {
        /* fill in every cases below */
```

```
                case 1:
                    days = 31;
                    monthName = "January";
                    break;
                case 2:
                case 3:
                case 4:
                case 5:
                case 6:
                case 7:
                case 8:
                case 9:
                case 10:
                case 11:
                case 12:
                default:
                    System.out.println("error!!!");
                    break;
            }
            System.out.printf("%s of %d has %d days.\n", monthName, year,
    days);
        }
}
```

Sample inputs and outputs:

```
D:\CS102A>java DaysOfYearMonth 2019 3
March of 2019 has 31 days.

D:\CS102A>java DaysOfYearMonth 2019 2
February of 2019 has 28 days.

D:\CS102A>java DaysOfYearMonth 1900 2
February of 1900 has 28 days.

D:\CS102A>java DaysOfYearMonth 2000 2
February of 2000 has 29 days.
```

7. Recall the `9 x 9 multiplication table` in the previous lab. Modify the program so that

a) The program can display a multiplication table of any given size in [1, 9].
b) The program keeps running until the user inputs 0.
c) The program will warn users for invalid inputs.

Try to use *break* and *continue* statements to complete the task.

Sample output:

```
Please input a number to print the Multiplication Table [0 to terminate]:
-4
Please input a number between [1,9]
Please input a number to print the Multiplication Table [0 to terminate]:
1
1 * 1 =  1
Please input a number to print the Multiplication Table [0 to terminate]:
3
1 * 1 =  1
1 * 2 =  2  2 * 2 =  4
1 * 3 =  3  2 * 3 =  6  3 * 3 =  9
Please input a number to print the Multiplication Table [0 to terminate]:
9
1 * 1 =  1
1 * 2 =  2  2 * 2 =  4
1 * 3 =  3  2 * 3 =  6  3 * 3 =  9
1 * 4 =  4  2 * 4 =  8  3 * 4 = 12  4 * 4 = 16
1 * 5 =  5  2 * 5 = 10  3 * 5 = 15  4 * 5 = 20  5 * 5 = 25
1 * 6 =  6  2 * 6 = 12  3 * 6 = 18  4 * 6 = 24  5 * 6 = 30  6 * 6 = 36
1 * 7 =  7  2 * 7 = 14  3 * 7 = 21  4 * 7 = 28  5 * 7 = 35  6 * 7 = 42  7 * 7 = 49
1 * 8 =  8  2 * 8 = 16  3 * 8 = 24  4 * 8 = 32  5 * 8 = 40  6 * 8 = 48  7 * 8 = 56  8 * 8 = 64
1 * 9 =  9  2 * 9 = 18  3 * 9 = 27  4 * 9 = 36  5 * 9 = 45  6 * 9 = 54  7 * 9 = 63  8 * 9 = 72  9 * 9 = 81
Please input a number to print the Multiplication Table [0 to terminate]:
0
```