Debriefing

Nom	Prenom
BADAROU	Mikdaam

Exercice 1 - Hello Groland

1. Write the following program in `HelloGroland.java` file

```
public class HelloGroland {
 public static void main(String[] args) {
     System.out.println("Hello Groland");
 }
}
```

Done ✅

2. Compile the program using the following command `javac HelloGroland.java`. Done ✅

3. Run the program using the following command `java HelloGroland`. Done ✅

Exercice 2 - Display the command line arguments

1. Display the first argument of the command line

```
public class PrintArgs {
 public static void main(String[] args) {
     System.out.println(args[0]);
 }
}
```

> If we run the program without arguments, the program fails with an exception raise. To avoid this issue, we can put the following statement in the source code :

```
int len = args.length;
if(len == 0) {
 System.err.println("Usage: java PrintArgs <arg1> <arg2> ....");
 return;
}
```

2. Adding to the previous code a loop to display the content of the args array

```
for(int i = 0; i < len; i++) {
 System.out.println(args[i]);
}
```

3. Using foreach loop instead of for loop

```
for(String arg: args) {
 System.out.println(arg);
}
```

> Note : The entire source code is available in `src/PrintArgs.java` file or :

```
public class PrintArgs {
 public static void main(String[] args) {
     int len = args.length;
     for(int i = 0; i < len; i++) {
         System.out.println(args[i]);
     }

     for(String arg: args) {
         System.out.println(arg);
     }
 }
}
```

Exercice 3 - Simple Calculator

1. Copy the previous program and complete it.

```
import java.util.Scanner;

public class Calc {
public static void main(String[] args) {
 Scanner scanner;
 scanner = new Scanner(System.in);
 int value;
 value = scanner.nextInt();
 System.out.println("Interger: " + value);
 }
}
```

2. Identify the variable and their type

Variables	Type
scanner	Scanner
value	int

- Put the variable initialisation in one line

```
import java.util.Scanner;

   public class Calc {
       public static void main(String[] args) {
           Scanner scanner = new Scanner(System.in);
           int value = scanner.nextInt();
           ...
       }
   }
```

3. Why the `nextInt()` is not a function.

   - nextInt()` is not a function because it's a instanciation method. In other words, to use the method we need to instanciate an object.
   - So `nextInt()` is an instanciation method.

4. Explains the following line code `import java.util.Scanner;`

   This line mean that we want to use the `Scanner` class in the `java.util` package. This line will be remplaced by the source code of the `Scanner` class when the program is compiled.

5. Edit the program to display the sum of two input numbers

```
import java.util.Scanner;

public class Calc {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Interger 1 : ");
        int value1 = scanner.nextInt();
        System.out.print("Interger 2 : ");
        int value2 = scanner.nextInt();
        System.out.println("Sum: " + value1 + " + " + value2 + " = " +
(value1 + value2));
    }
}
```

6. Adding to the previous code the display of others operation.

```
import java.util.Scanner;

public class Calc {
    public static void main(String[] args) {
        ...
        System.out.println("Product: " + value1 + " * " + value2 + " = " +
(value1 * value2));
        System.out.println("Substraction: " + value1 + " - " + value2 + " =
" + (value1 - value2));
        System.out.println("Quotient: " + value1 + " / " + value2 + " = " +
(value1 / value2));
        System.out.println("Remainder: " + value1 + " % " + value2 + " = " +
(value1 % value2));
    }
}
```

Exercice 4 - Record et conversion of String to interger

1. Create a Point record with two components, x and y.

```
public record Point(int x, int y) {};
```

> The commande line to compile this code is `javac Point.java` in version 17 of java.

2. Adding a main method to the record

```
 public static void main(String[] args) {
     if(args.length != 2) {
         System.err.println("Usage: java Point x y");
         return;
     }
     int x = Integer.parseInt(args[0]);
     int y = Integer.parseInt(args[1]);
     System.out.println("x="+ x +", y="+ y);
 }
```

3. What does `static` means for a method ? If a method is static, it's means that, it can be call without instanciate an object of the class.

4. What does happen if an argument is not a number ? If an argument is not a number, the program raise the following exception : `java.lang.NumberFormatException`.

5. Adding to the previous code, instructions to creata an instance of the Point record.

```
 public static void main(String[] args) {
     ...

     Point p = new Point(x, y);
     System.out.println(p.toString());
  }
```

6. Add an instanciation methode to the **Point** record named `distance`.

```
/**
* parameter Point point
* return float
*/
public double distance(Point point) {
    float dist;
    dist = Math.sqrt((this.x - point.x()) * (this.x - point.x())  + (this.y
- point.y()) * (this.y - point.y()));
    return dist;
}
```

Exercice 5 - From C to Java

1. Compiles the C program with the command `gcc -o pascal pascal.c`. When we run the program with `time` command, we get the following output:

```
user@host:~/workdir$ time ./pascal
Cn, p = -1742193024

real    0m1.074s
user    0m1.070s
sys     0m0.001s
```

2. Compiling the Java program with the command `javac Pascal.java`. When we run the program with `time` command, we get the following output:

```
user@host:~/workdir$ time java Pascal
Cn, p = -1742193024

real 0m0.297s
user 0m0.276s
sys 0m0.029s
```

Explanation

We can notice that the execution time of both program is pratically the same. This is due to the JIT(Just In Time)