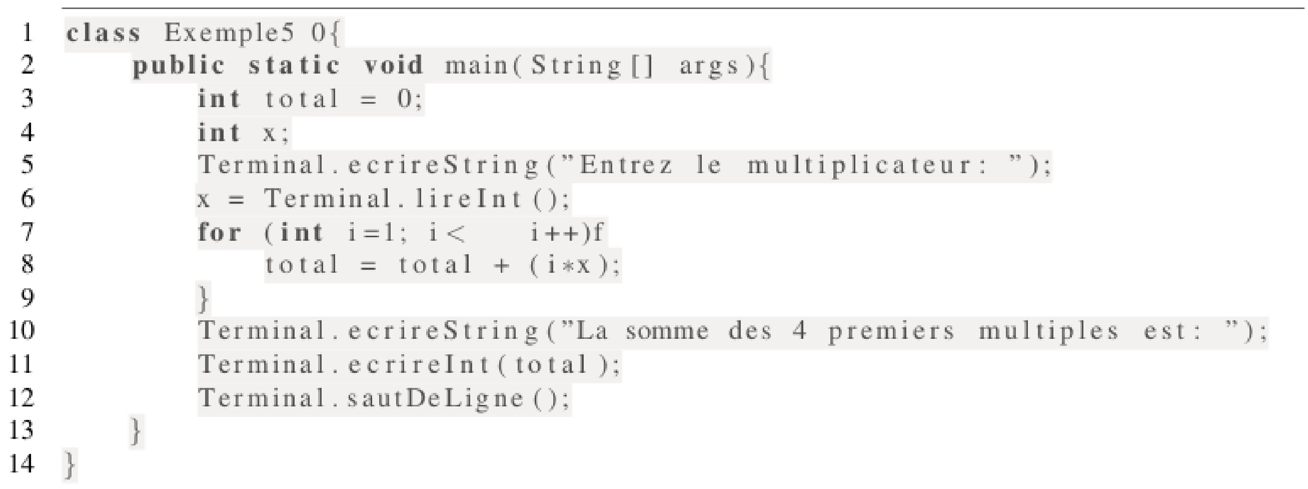
Exercises 4: loops

Example 4.0: Running a program

In this example, we will go step by step through execution of a program. The instructions of a program can change the memory state (values of variables), and have an impact on execution via I/O operations. The declarations have the effect of creating new variables. We will trace the program execution in a summary table with program steps, noting the effect of the declarations and instructions. The first column refers to the line numbers of the program. Next columns are used to record the value of the conditions in conditional statements and loops. Then there is a column for each variable and finally a column for keyboard input and a column for outputs on the screen.

We denote a variable that is not defined by **DNE** (does not exist) in the corresponding column. A question mark **(?)** is used when the variable exists (it was declared), but it has not been initialized.



```
1  class Exemple5 0{
2      public static void main(String[] args){
3          int total = 0;
4          int x;
5          Terminal.ecrireString(”Entrez le multiplicateur: ”);
6          x = Terminal.lireInt();
7          for (int i=1; i<     i++)f
8              total = total + (i*x);
9          }
10         Terminal.ecrireString(”La somme des 4 premiers multiples est: ”);
11         Terminal.ecrireInt(total);
12         Terminal.sautDeLigne();
13     }
14 }
```

Exercise:
Draw the execution table with **loop** steps (1 .. 9) , the variables and the results
The first line of the table shows the initial state before execution. The **for** statement has **three parts** involved at different times : **initialization i=1**, **condition test i <= 4** (when the condition is **true**, the loop continues when it is **false**, the loop stops) and the variable change **i++**. Other types of loop (**while** and **do ... while**) comprise only **condition test** part.

Line 9 ends a block, which puts an end to the existence of the variable i that is **local** to that block. Line 13 terminates variables x and the whole program.

If you change the value of x in the input line 6, you change the whole table. We see in this example that the body of the loop - line 8 - is executed 4 times during the program.

```
> java  Exemple4_0
Entrez le multiplicateur: 3
La  somme  des 4  premiers multiples  est: 30
```

Exercise 4.1: Running a `for` loop

Trace the execution of the following program in which you enter on the keyboard the value 5.

```
class Exo5 1{
    public static void main(String[] args){
        int x;
        Terminal.ecrireString(”Un entier svp: ”);
        x = Terminal.lireInt();
        for (int i = 0; i<4; i++){
            Terminal.ecrireInt(x+i);
            Terminal.sautDeLigne();
        }
        Terminal.ecrireStringln(”Fini”);
    }
}
```

Exercise 4.2: Running a `while` loop

```
class Exo6 2{
    public static void main(String[] args){
        int puis = 1;
        int x, res;
        Terminal.ecrireString(”Un entier svp: ”);
        x = Terminal.lireInt();
        res = x;
        while (res < 1000){
            res = res * x;
            puis = puis+1;
        }
        Terminal.ecrireString(”Le résultat est ”);
        Terminal.ecrireInt(puis);
        Terminal.sautDeLigne();
    }
}
```

1. What calculates this program?
2. Trace the execution of this program in which you enter on the keyboard the value 8.

Exercise 4.3: Calculations

1. Write a program that displays the multiplication table of a number. This value will be entered by the user. For example, if the number is 3, the program displays:

1 x 3 = 3

..

9 x 3 = 27

2. If you have not already done so, change your program to verify that the number entered by the user is a number between 1 and 9.
3. writing a program that computes x^y where x and y are two integers keyed. To do this, multiply x by y times. Eg 2^3 = 1 * 2 * 2 * 2.
4. write a program that displays the value of the function x^2 (x squared) for the first ten positive integers.

Exercise 4.4: Conversion to dollars

The following program is a version of the Conversion program, modified to calculate the conversion of an amount of dollars into euros.

```
public class Exo212 {
    public static void main (String[] args) {
        double euros, dollar, cours;

        Terminal.ecrireStringln(”Cours du dollar (valeur de 1 dollar)? ”);
        cours = Terminal.lireDouble();
        Terminal.ecrireStringln(”Somme en euros? ”);
        euros = Terminal.lireDouble();
        dollar = euros / cours;
        Terminal.ecrireStringln(”La somme en francs: ”+ dollar);
    }
}
```

Modify the program so that, for given exchange rate of dollars (that does not change during the program execution), we can convert many amounts. Provide a means to stop the program execution.

Exercise 4.5: Correct date

Write a program that takes a valid date in the form of three integers (day, month and year). The program must check if the date is correct, and if this is not the case, should indicate the type of error, then ask a new entry.
The program ends when the correct date is finally entered, with the display of it. In case the month of February is the date, your program should calculate whether the year is a leap year. Generally, it will calculate the maximum number of days of the month of the date entered in order to confirm the number of days that has been entered.