TP4: Iterations in Java

Exercise 1 Loops

Recall the loop instructions available in Java:

1. The `while` loop: repeat a sequence of instructions as long as a condition is met:

```
while (condition) {
    // bloc of instructions
}
```

During the execution, we check if `condition` returns `true` and we execute the block of instruction.

2. The `do... while` loop: repeat a sequence of instruction as long as a condition is verified, but the verification takes place after the block of instruction has completed:

```
do  {
// bloc of instructions
} while (condition);
```

3. The `for` loop: a particular case of the `while` loop.

```
for( expression A; expression B; expression C){
    // bloc of instructions
}
```

is equivalent to

```
{
    expression A; // initialization
    while (expression B) {
        // bloc of instructions
        expression C;
        }
    }
}
```

Notice that

- `expression A` corresponds to the initialization of the loop
- `expression B` is the condition of the `while` loop
- `expression C` is ran after the block of instruction corresponding to one iteration of the loop is executed.

We can use a `for` loop to vary a counter between two bounds. Example:

```
int i;
for(i=-10 ; i<=10; i++){
    System.out.println(i);
}
```

Exercise 2 Examples

1. Write a program which:

(i) Asks an integer number n from the user
(ii) Asks n real numbers from the user
(iii) Outputs the sum of the n numbers
(iv) Outputs the average of the n numbers

Write a variant of the same program which:

(i) Asks for a real number from the user **until the number zero is given**
(ii) Updates the average of the numbers and gives it as an output after each iteration.

2. Trees:

Write a program which draws a tree of a given height h which is given by the user as an integer. The following types should be available, illustrated for $h = 5$.

– A "full tree"

```
    *
   ***
  *****
 *******
*********
```

– An "empty tree"

```
    *
   * *
  *   *
 *     *
*********
```

– A "tree on the side"

```
*
**
***
****
*****
****
***
**
*
```

Exercise 3 Guess a number

Write a program which realizes the following game. First, the computer chooses a random number x between 1 and 100. The user will try and guess it and enters successive guesses. After each try, the computer indicates if the given integer is larger or smaller than x. If the input is equal to x then the game ends and the number of given inputs is shown.

Modify the program in order that the user can, if he/she wishes, restart the game. When quitting the program, the best score must be plotted.

Below you have a program which chooses random values using the class `Random`. You can use these commands in your code.

```
/* Example of random number generator */
import java. util .*;

class ChooseNumber {

    static final Random random = new Random();
    public static void main(String [] args) {
        System.out. println ("An integer (int) between 0 included
           and 20 excluded");
        System.out. println (random.nextInt(20));

        System.out. println ("A real (float then double) between 0.0
            included and 1.0 excluded");
        System.out. println (random.nextFloat());
        System.out. println (random.nextDouble());
    }
}
```