TypeScript Exercises – Iteration

Outcomes

By the completion of this exercise you should understand how to:

- Capture input from the user
- Use a while loop

Overview

Introduction

In this exercise we will be looking at how we can iterate (loop) through a block of code using a **while** loop.

Instructions

For the following tasks, you will be provided minimal instructions, and have to rely on the guidance and your own research to complete them.

You have been provided with an existing TypeScript project called **iterationExercises**. Your work should be placed inside of this project.

Before starting on the exercise, open a terminal in the same folder as **package.json** (for example, using Visual Studio Code) and run **`npm install`** to re-install all packages.

You will be required to work with functions in this exercise, as well as exporting/importing functions a module. Some functions have been provided for you, some you will have to create and write yourself. If you're unsure how to create new functions and export/import them, be sure to consult your teacher.

Task 01

Write JavaScript code which displays the numbers 1 to 12 separated by tabular spaces.

Place your work inside the **printWithTabs** function located in the file **exercises.ts**. Call your function from the **main** function of the file **index.ts** by uncommenting the line:

```
printWithTabs()
```

The expected output should be:

```
1	2	3	4	5	6	7	8	9	10	11	12
```

Task 02

Write JavaScript code which displays the numbers 12 to 1 separated by commas ",". Make sure the last number is not displaying the comma after it.

Write a function called **printWithCommas** and place it inside the provided file **exercises.ts**. Call your function from the **main** function of the file **index.ts** by uncommenting the line:

```
printWithCommas()
```

The expected output should be:

```
12,11,10,9,8,7,6,5,4,3,2,1
```

======= MORE TASKS ON THE NEXT PAGE =======

Task 03

Write JavaScript code which starts with the number 1 and then displays the result of halving the previous number until the result is less than 0.001. Your work should be placed in a function called **halveNumbers** inside of the appropriate **.ts** file.

Ensure your function is called when the program runs.

The expected output should be:

```
1.0
0.5
0.25
0.125
0.0625
0.03125
0.015625
0.0078125
0.00390625
0.001953125
```

Task 04

Write JavaScript code which generates and displays 10 random (integer) numbers between 0 and 50, note that each time this program runs the results are different.

Your work should be placed in a function called **generateRandomNumbers** and be placed in the appropriate **.ts** file. Ensure your function is called when the program runs.

Task 05

Modify your code from the previous task. In addition to generating the 10 random numbers, display the lowest of the 10 numbers.

Task 06

Modify your code from the previous task. In addition to the lowest number, display the highest and the average of the 10 numbers.

======= MORE TASKS ON THE NEXT PAGE =======

Task 07

Write a program that simulates the roll of two six sided dice.

The program should ask the user to guess the total sum of the dice. The user has to keep guessing until they get the total correct, and the program should tell them how many tries it took.

Save your work in a new function called **rollDice**.

An example of the expected output is shown below.

```
What do you think the total of two six-sided die will be? 6

The total was 8.  Try again!

What do you think the total of two six-sided die will be? 4

The total was 11.  Try again!

What do you think the total of two six-sided die will be? 10

The total was 10.  Congratulations! It took you 3 tries to guess successfully.
```