Programming 2 - C Language - HW1

General Instructions

Please read the questions thoroughly and make sure your programs run as specified by the question statements. Remember to follow the submission rules given on the Moodle. You can find the submission rules in Hebrew and in English. Please note that in order to view these instructions you will have to be logged in to the Moodle website.

In addition to this pdf, you can find three files named **`1.c`**, **`2.c`** and **`3.c`** that you will use for solving the three questions of this homework. Each of the files contains a **`main()`** function with the first print that the exercise requires. Additionally, the file contains various print functions that you can use. You will learn about functions used for standard output on tutorial 4 and in the meantime, you only need to use these functions on the correct places in your programs.

In order to solve this homework, you should first **create a new project** and add the files to the project.

Remember: Each file can contain only one **`main()`** function, so in order to complete all three questions you need to create three projects in the same solution. For additional information go to the manual page on working with multiple files.

Submission: After completing the assignment, change the 3 filenames according to the following pattern: ex1_q<num>_123456789.c (with the correct question number and your ID number instead of 123456789), zip the 3 files to a zip file with the following name:
`ex1_123456789.zip` (with your ID number instead of 123456789)
Once you have created the archive, you should submit it on the Moodle website.

Notice: This assignment has to be solved alone, not in pairs!

Basic calculations

In the current coronavirus days, we want to write a program that will help worried people to decide whether to call MDA (Magen David Adom) or not.

The program will receive three numbers in the following order:

- The number of coughs in the last 24 hours
- The number of sneezes in the last 24 hours
- The age of the person (between 0 and 120)

The program should print:

- The average of coughs, sneezes and the age (sum of these three numbers divided by three)
- The symptom with a bigger number (sneezes or coughs). If the number of sneezes and coughs is equal, do not print anything.
- If the average is more than 40, the program should print `"Call MDA!"`, and otherwise the output should be `"Don't call MDA!"`

The file **`1.c`** contains the function **`main()`**.
The first line in the function prints the following message:

```
Please enter three whole numbers:
```

After this line, the user is expected to enter the input, as described above.

After calculating the average, call the function **`avgOut`** with a parameter that is the average that you have computed. This function should output the average. After deciding which symptom occurred more times, call the responding function **`sneezeOut`** or **`coughOut`** accordingly, if the number of sneezes and coughs are equal, don't print anything.
After deciding whether to call MDA, call the responding function **`callMda`** or **`dontCallMda`** accordingly.

Example 1 (Program, User): Please enter three whole numbers: 50 12 78 The average is: 46.667 You cough more than you sneeze Call MDA!	Example 2 (Program, User): Please enter three whole numbers: 5 40 35 The average is: 26.667 You sneeze more than you cough Don't call MDA!

Sort a 3-digit number

Write a program that gets a 3-digit number and outputs the number with the digits sorted from the lowest to the highest. For example, if the input number is 386, the output should be 368.

The file **`2.c`** contains the function **`main()`**.
The first line in the function prints:

```
Please enter a 3-digit number:
```

After this line, you should take input from the user, and then calculate the output as needed in the question. It is necessary to make sure that the input is indeed a 3-digit number. If it is not a 3-digit number, the program should pass -1 to the function **`printSorted`**.

You can assume that the input is a positive integer and you do not need to check it.

Example 1 (Program, User): Please enter a 3-digit number: 393 The sorted number is: 339	Example 2 (Program, User): Please enter a 3-digit number: 7224 This is not a 3-digit number!

Handshakes

In the current coronavirus days, we want to write a program that will calculate the number of handshakes in a room with N people. Despite the official suggestions, each person in the room shakes hands **once** with every other person! For example, if there are 4 people in the room, the first will shake hands with the other three, the second will shake hands with the third and the fourth, and the third only has to shake hands with the fourth so that every two persons have shaken their hands. This example leads to a total of 6 handshakes.

The file **`3.c`** contains the function **`main()`**.
The first line in the function prints:

```
Please enter the number of people in the room:
```

After this line, you should take input from the user, and then calculate the output as needed in the question. The program should pass the sum of handshakes to **`printSum`**.

You can assume that the input is an integer and you do not need to check it. You do need to check that N is positive, and if it is not, you should pass -1 to **`printSum`**.

Example 1 (Program, User): Please enter the number of people in the room: -4 The number you entered is not positive!	Example 2 (Program, User): Please enter the number of people in the room: 50 The number of handshakes: 1225