Follow this tutorial only if you do not have access to the LIP machines and are using Docker and the ROOT container in your own computer

1. A Linux terminal is a text interface that allows you to write and execute commands. The goal of this exercise is to learn a few simple commands.
    a. Open a new terminal and start the ROOT docker container (you should have already followed the instructions here to set-up Docker and start the ROOT container)
    b. Go to the shared folder inside the container: `cd /shared`
    c. Create a new directory named test: `mkdir test`
    d. Go into the directory: `cd test`
    e. Start a text editor software in your native system to write a program in C++ (e.g. Windows: notepad++; Linux: emacs, gedit; MacOS: XCode, Sublime Text)
    f. Type some text and save it as a new file (`test.txt`) under the shared folder, inside the folder you just created (Linux/MacOS: $HOME/docker/tutorials/test, Windows: the folder you selected when starting the container in the Docker interface)
    g. On the terminal, the new file should be visible. List all the files in the directory: `ls`
    h. Print the file content in the terminal: `cat test.txt`
    i. A few other useful commands:

Command	**Description**
pwd	Show path of current directory
mkdir name	Create directory name
cd name	Go into directory name
ls	List all files and directories in current directory
ls -l	Include additional information in the listing
cp, mv	Copy/move files
man command	Open the manual of a command
command -h	Get help on the syntax of a command
g ++	Compile C and C++ programs
exit	Exit the terminal
tar -xvzf name.tar.gz	Uncompress file with extension tar.gz
tar -cvzf name.tar.gz	Create zipped file with extension tar.gz

2. **Introduction to C++** | The goal of this task is to write a first C++ program.
    a. Open a terminal and start the ROOT container (as in the previous step) -- not needed if you already have one open
    b. Go to the shared folder and create a new directory called `hello`
    c. Go into this directory
    d. Open a text editor and create a C++ file called `hello.cpp` in the folder you just created
    e. Type the following code, and save the file:

    ```
    #include <iostream>
    using namespace std;

    int main(){
        cout<<"Hello␣World"<<endl;
        return 0;
    }
    ```

    f. Compile the code in the terminal: `g++ hello.cpp -o hello.x`
    g. Run the code: `./hello.x`
    h. Try changing the output text, compile and run the program again
    (hint: you can use the up/down error keys in the terminal to access previous commands)

3. **Another C++ example** | The goal is to write a code that generates a table with the values given by a parabola.
    a. Start a new file called `parabola.cpp` and write the following code:

    ```
    #include <iostream>
    using namespace std;

    int main(){
        for(int i = 1; i<=10; i++){
            double y = i*i; // Create new variable
            cout<<i<<"\t"<<y<<endl;
        }
        return 0;
    }
    ```

    b. Save the file inside the shared folder and compile it following the instructions of the previous example
    c. Run the new program saving the output to a file parabola.dat:
    `./parabola.x > parabola.dat`
    d. You can see the content of this new file directly in the terminal:
    `cat parabola.dat`

4. **Simple arrays** | Implement a new program that defines an array with the following values

   {10.5, 9.3, 11.4, 10.9, 13, 8.4, 9.2, 8.9, 10.3, 11.2, 12.1, 8.4, 9.2, 9.9, 10.1}

   The program should run over all values and print them to the screen. Then it should ask the user to enter a number between 1 and 15 and print the corresponding number of the array.

5. **Calculate mean values and standard deviation** | Change the program you wrote on the previous exercise to calculate the following quantities for the same array:
   a. Mean value of the numbers in the array

      $$< x >= \frac{1}{N}\sum_{i=1}^{N} x_i$$

   b. Standard deviation

      $$\sigma = \sqrt{var}, \qquad var = \frac{1}{N}\sum_{i=1}^{N}(x_i - < x >)^2$$

6. **Conditional statements** | Using the same program as in the previous exercises, define the following new array:

   {1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 1, 0, 1}

   Loop over the entries of the array and whenever you find an entry with the value 1, print the corresponding entry of the initial array. Then for all entries marked with 0 (or 1) calculate the mean value and the standard deviation.