Assignment #2

CSC111: Introduction to Programming

Learning Outcomes

When you have completed this assignment, you should understand:

- How to define and call functions that take arguments.
- How to write condition statements using relational operators (<, <=, ==, etc)
- How to write conditional statements (if, if/else)
- How to compose condition statements using logical operators (and, or, etc)

Assignment Overview

1. Create a single new file called `assignment2.c` in Jupyter Hub
2. Design the functions according to the specifications in the Function Specifications section.
3. Test to make sure your code produces the expected output.
4. Ensure that the filename is `assignment2.c` (files with an incorrect name will not be marked).
5. Download your `assignment2.c` file from Jupyter Hub (File→Download) and submit this single file to the Assignment 2 dropbox (under Assignments) on the CSC111 BrightSpace Page.
6. Download your file from BrightSpace to confirm that name and file contents are as expected.

Reminder: Your code is to be designed and written by only you and not to be shared with anyone else. See the Course Outline for details explaining the policies on Academic Integrity. Submissions that violate the Academic Integrity policy will be forwarded directly to the Computer Science Academic Integrity Committee.

Grading

1. Late submissions will not be accepted
2. You must submit a single file with the name `assignment2.c` or you will be given a **zero grade**.
3. Your function names must match the specification **exactly** or you will be given a **zero grade**.
4. Any code written in the main function may not be marked.
5. We will do spot-check grading for code quality. All assignments are graded BUT only a subset of your code may be graded for code quality. You will not know which portions of the code will be graded, so all of your code must be complete and adhere to specifications to receive marks.
6. Your code must run without errors in the Computer Science department's Jupyter Hub environment using the compilation command provided in this document.
7. We reserve the right to replace your main function with our own, allowing it to better interact with our testing software.
8. It is the responsibility of the student to submit any and all correct files. Only submitted files will be marked. Submitting an incorrect file or forgetting a file is not grounds for a regrade.

Marks will be given for:

- Your code producing the correct output.
- Your code following good coding conventions
  - Proper indentation
  - Documentation (comments)
  - Use of whitespace to improve readability
  - Names of variables should have meaning relevant to what they store

Compiling and Running

Use the following commands to compile and run your code when you are testing your work.

```
gcc -Wall -Werror -pedantic -std=c18 -o assignment2 assignment2.c
```

and the following to run the code:

```
./assignment2
```

NOTE: Where examples are provided for clarity in the problem descriptions, they do not necessarily consider all edge cases – your testing should take into account all edge cases as we will when grading your submissions.

Functions Specifications

When testing your code type the following command to compile your code. Every line of output should have a new line after it. This section will describe the functions that you are tasked to create:

1. Design a function called `print_qualification_status()`, that meets the following criteria:
   - The function should take in two floating point numbers, 1) the time needed to qualify for the Olympic event, and 2) the time a racer achieved in a trial event. You may assume that the function arguments will be larger than 0.
   - **Remember:** the arguments MUST be in this order.
   - If the time achieved by the racer is at least as fast, or faster than the time needed to qualify for the Olympic event, the function should print:

     ```
     You qualified at X second below the qualifying time
     ```

     Where **X** is the number of seconds (to two (2) decimal places) that the racer's time was under the qualifying time.

   - If the time achieved by the racer is over the qualifying time, the function should print:

     ```
     You missed qualifying by Y seconds
     ```

     Where **Y** is the number of seconds (to two (2) decimal places) that the racer's time was over the qualifying time.
2. Design a function, called `print_median()`, that meets the following criteria:
   - The function should take as arguments three floating point numbers and print the median of these three values.
   - The function should print the value to four decimal places.
   - **RECALL:** the median of a set of numbers is the middle number of that set in sorted order. Your function should print only the median number with nothing before or after it.
   - **Example:** If the function is called with `print_median(3.2, 20.22, 3.2)` it should print:

     ```
     3.2000
     ```

3. Design a function called `print_triangle_type()`, that meets the following criteria:
   - The function should take three floating point numbers that represent the side lengths of a triangle. The function should print the type of triangle that these side lengths represent as:
     - Equilateral Triangle
     - Isosceles Triangle
     - Scalene Triangle
   - Your function can assume the arguments passed are that of a valid triangle.
   - **RECALL:** a triangle with all sides with the same length is equilateral, a triangle with two of the same sides is an isosceles and a triangle with no sides the same is a scalene triangle.

- Example: If the function is called with `print_triangle_type(2,2,2)` it should print:

```
Equilateral Triangle
```

4. Design a function called `is_multiple_of ()` that meets the following criteria:

   - The function should take two integers n1 and n2 as arguments (in this order).
   - The function should determine whether the second argument (n2) is a multiple of the first argument (n1).
   - If n2 is a multiple of n1, the function should print the following message:

   ```
   The value of n2 is a multiple of the value of n1
   ```

   - If n2 is not a multiple of n1, the function should print the following message:

   ```
   The value of n2 is not a multiple of the value of n1
   ```

   - **NOTE:** n1 and n2 above must be replaced with the values that are passed into the function as arguments (see the example below).
   - **RECALL:** The definition of a multiple can be found here
   - **Example:** If the function is called as: `is_multiple_of(3,12)`. 3 divides evenly into 12, 4 times, 12 is therefore a multiple of 3. The function prints:

   ```
   The value of 12 is a multiple of the value of 3
   ```

5. Design a function called `process_breathalyzer_results()` that meets the following criteria:

   - **Background:** In British Columbia, if you are caught driving with too much alcohol in your system, you can be charged an administrative fee and be prohibited from driving for a specified number of days. A breathalyzer test is given to a driver and if their blood alcohol content is 0.08 mg/ml or more they fail the test. If the blood alcohol content is less than 0.08 mg/ml but still 0.05 mg/ml or more they receive a warning but if it is less than 0.05 mg/ml they will pass.
   - **Background:** The administrative penalties and driving prohibitions can be found here. We have summarized this on the next page for your convenience. **If any discrepancy is found between the summary and the webpage, the assignment specification document's summary will be taken as correct**.

   - The function should take two arguments, in the following order: a double for the person's blood alcohol content and an integer, for the number of times the person has been warned.
   - Your function should print a message that status the result of the breathalyzer test as `PASS`, `WARNING`, or `FAIL`.
   - If the person did not pass the test, the function should also print the amount of the **administrative penalty** the person must pay and the **number of days they are prohibited from driving**. **Examples**

     (a) If the function is called as: `process_breathalyzer_results(0.049, 0);` it should print:

     ```
     PASS
     ```

     (b) If the function is called as: `process_breathalyzer_results(0.06, 1);` it should print:

     ```
     WARNING penalty: $300, driving suspension: 7 days
     ```

     (c) If the function is called as: `process_breathalyzer_results(0.09, 0);` it should print:

     ```
     FAIL penalty: $500, driving suspension: 90 days
     ```

Summary of BC administrative penalties for intoxicated driving

If, after being asked by a police officer to provide a breath sample, you register a `WARNING` using an Approved Screening Device, you will receive an Immediate Roadside Prohibition. If you fall into this range, your Prohibition will be 3 days, 7 days or 30 days in length, depending on your driving record as follows:

- If it is the **first time** that you have registered a `WARNING`, you will receive a 3-day Immediate Roadside Prohibition. You will face the following consequences:
  - 3 day prohibition from operating a motor vehicle
  - 3 day impoundment of the vehicle that you were operating
  - Cost of towing the vehicle to the impoundment lot
  - Cost of storing the vehicle at the impoundment lot
  - $200.00 administrative penalty
  - $250.00 license reinstatement fee
- If it is the **second time** that you have registered a `WARNING`, you will receive a 7-day Immediate Roadside Prohibition. This means that you will be prohibited from operating a motor vehicle for a period of seven days. You will face the following consequences:
  - 7 day prohibition from operating a motor vehicle
  - 7 day impoundment of the vehicle that you were operating
  - Cost of towing the vehicle to the impoundment lot
  - Cost of storing the vehicle at the impoundment lot
  - $300.00 administrative penalty
  - $250.00 license reinstatement fee
- If it is the **third or greater time** that you have registered a `WARNING`, you will receive a 30-day Immediate Roadside Prohibition. This means that you will be prohibited from operating a motor vehicle for a period of seven days. You will face the following consequences:
  - 30 day prohibition from operating a motor vehicle
  - 30 day impoundment of the vehicle that you were operating
  - Cost of towing the vehicle to the impoundment lot
  - Cost of storing the vehicle at the impoundment lot
  - $400.00 administrative penalty
  - $250.00 license reinstatement fee
  - Completion of the Responsible Driver's Program, with an associated cost of $880.00 plus tax
  - Installation of an Ignition Interlock device in any vehicle that you operate, for a minimum period of at least one year, with an associated cost of $1,730.00 plus tax

If, after being asked by a police officer to provide a breath sample, you register a 'fail' using an Approved Screening Device, you will receive an Immediate Roadside Prohibition. You will face the following consequences:

- 90 day prohibition from operating a motor vehicle
- 30 day impoundment of the vehicle that you were operating
- Cost of towing the vehicle to the impoundment lot
- Cost of storing the vehicle at the impoundment lot
- $500.00 administrative penalty
- $250.00 license reinstatement fee
- Discretionary order by the Superintendent of Motor Vehicles that you complete the Responsible Drivers Program, with an associated cost of $880.00 plus tax
- Discretionary order by the Superintendent of Motor Vehicles that you install an Ignition Interlock Device in any vehicle that you operate, for a minimum period of at least one year, with an associated cost of $1730.00 plus tax