Background

The purpose of this assignment is to gain some experience working with pointers in a fairly simple program. Pointers are powerful data types, but becoming comfortable with their syntax requires practice.

Problem Statement

In this assignment, you are to write a short program that uses four float variables and pointers to those variables in order to perform mathematical operations.

There is not an extensive amount of programming in this assignment, the development of which will be split into two parts, **Part A** and **Part B**. However, there are some special requirements.

- Your code must include the six statements listed below at the start of `main()`.
- The variables declared in these statements will be the only variables used in your program.
- **Part A** will only use the variables `ptr1`, `ptr2`, `ptr3`, and `ptr4`.
- **Part B** will only use the variables `a`, `b`, `c`, and `avg`.

```
float a, b, c, avg;
float *ptr1, *ptr2, *ptr3, *ptr4;
ptr1 = &a;
ptr2 = &b;
ptr3 = &c;
ptr4 = &avg;
```

Instructions

Represent

- Consider creating a flowchart, algorithm, or pseudocode for solving the problem.

Plan

- Create a file named **APP_C28_1.cpp** .
- Consider outlining the steps your program will take by adding comment statements to your file based on the flowchart, algorithm, or pseudocode.

Implement

- **Part A** – Write a complete C/C++ program **APP_C28_1.cpp** to perform the following tasks using only the variables `ptr1`, `ptr2`, `ptr3`, and `ptr4`:
  - Prompt the user to input three numbers, either positive or negative.
  - Compute their average and print it to the screen.
  - Compute the absolute value of each number and print them to the screen. (Use the library function `fabs()` which requires the header file `math.h` . `fabs()` returns the absolute value of the floating point number passed into it. For more information on `fabs()` enter `man fabs` at the prompt in a Linux terminal window.)
  - Compute the average of the absolute values and print it to the screen.
  - Print the value stored in `a` and the address of `a` using only `ptr1`. (Use the `%p` format specifier for the address and the `%f` format specifier for the value. The `%p` will cause the address to be displayed in hexadecimal format (base 16).)
- Compile, link, and run your program using the test conditions below.
- Verify that **Part A** of your program works correctly

- **Part B** – Once **Part A** is working, modify the program by adding code to **`main()`** <u>after</u> **Part A** to perform the following tasks using only the variables **`a`**, **`b`**, **`c`**, and **`avg`**:
  - Prompt the user a second time to input three numbers, either positive or negative.
  - Compute their average and print it to the screen.
  - Compute the absolute value of each number and print them to the screen.
  - Compute the average of the absolute values and print it to the screen.
  - Print the value stored in **`a`** and the address of **`a`** using only the variable **`a`**. (Use the **`%p`** format specifier for the address and the **`%f`** format specifier for the value. The **`%p`** will cause the address to be displayed in hexadecimal format (base 16).)
- Compile, link, and run your program using the test conditions below.
- Verify that **Part B** of your programs works correctly.

Evaluate

- Perform hand calculations to verify and check your results. Additionally, verify that the results from **Part A** and **Part B** match.
- Briefly discuss the relationship between the values printed in **Part A** and **Part B.**

Document

- Create a single PDF that includes your code, output to the terminal, verification, and discussion.
- Submit the PDF to Carmen according to the DAL.

Test Conditions

- Input **`a`**: -5
- Input **`b`**: 15
- Input **`c`**: -7

Include the standard comment and **`printf()`** statements indicating name, seat number, etc.