Name :	**Student ID :**
Date:	**Professor Signature:**

Lab 2 – Floating-Point Number Conversions

The number system used in the computer system is the binary number system. But we have different data types and need to use different formats to save different data types. In this lab, we will see how the floating-point numbers and signed whole numbers can be processed with a C program.

Part 1: Programming

Use the lab2.c file provided, finish function options 1 and 2 in this lab.

Option 1: When you input an 8-bit binary number ***character*** ***string***, you should be able to print its decimal number in a signed format. The size of the binary string should be 8-bit. For example, if your input is 11111111, you should output -1 on the screen. If your input is 10000001, you should print -127 on screen.

Option 2: When you input a 32-bit binary number ***character*** ***string***, you should be able to print its floating-point number. The size of the binary string should be at least 8-bit. For example, if your input is 11000001010010000000000000000000, you should print -12.5 on screen. If your input is 01000001010101000000000000000000, you should print 13.25 on the screen.

(Hint: if the 32-bit input of this option is not easy, try to predefine the above strings as constants in #define derivative and used it directly in your code. i.e.,

#define number1 "11000001010010000000000000000000"

#define number2 "01000001010101000000000000000000"

… and then use `strcpy(inputs, number1)` to replace statement `scanf("%s", &inputs)` in option 2 to test.)

Question #1: Did you use `strcpy` to solve the problem? If so, was this a best practice? What makes you say so? If this is not a best practice, what alternative solution should be used?

Part 2: Software Practice:

Download the free version of the software called "Novabench" (google search it). This program is used as benchmark software to test the overall computer system performance. Run the program and perform a test. After the test is done, fill in the following information.

Overall Score	
CPU Model	
CPU Score	
GPU Score	
RAM Score	
Disk Score	

Disk Write Speed	
Disk Read Speed	
RAM Read Speed	
GPU Model	

Report Submission Requirements

1. Please use the lab report template and follow its instructions to submit:
    a. Please include the solutions to the part 2 questions.
    b. Must have your own comments to explain your source code.
    c. The source code and the screenshots of the running results from part 1. These resources (code and images) can be submitted as a link to a GitHub repo.
    d. Must include the screenshots of at least two test cases for options 1 (one positive number and one negative number) and 2 (two 32-bit test cases provided) in the report.
    e. The answer to Question 1.
    f. Please submit the lab report on the blackboard by the due date.