Lab 10 Pointers (I)

Please test the correctness of your program in **Q2** and **Q3** on **PASS**.

Q-1.

Write a program that asks the user to enter integers as inputs to be stored in the variables **A** and **B** respectively. There are also two integer pointers named **ptrA** and **ptrB**. Assign the addresses of **A** and **B** to **ptrA** and **ptrB** respectively and display the values in **A** and **B** using **ptrA** and **ptrB**.

Note:
Outputs may be different in different runs as we print addresses in program, the same hereinafter.
To manipulate the stream to print **foo** in hexadecimal use the hex manipulator:

```
cout << hex << foo;
```

Expected Outputs:

```
Enter value of A: 10
Enter value of B: 20
Value of ptrA is 10 stored in address 002EF83C
Value of ptrB is 20 stored in address 002EF830
```

Q-2. **[will be marked]**

Given any two non-zero values from user, both can represent voltage (**V**) or current (**I**) and if one is **V**, the other will be **I**. Download the **resistance.cpp**, and modify this program to compute the possible values of resistance(**R**) from the user input, where **R=V/I**. For example, when the two input values are 5.1 and 2, **R** can be 5.1/2 = 2.55 or 2/5.1 = 0.39.

The program should be made up of the following four functions:

1. **getInput()**: get two values from user using **call by pointer**, where the first one is **V** and the second is **I**. The return type of this function is **void**.
2. **toResistance()**: calculate the resistance **R** given **V** and **I**. The function should return a real number for the value of **R**.
3. **swap()**: swap the values of **V** and **I** to obtain the other possible pair of voltage and current using **call by reference**.
4. **main()**: call **getInput()** to obtain **V** and **I**. After that, the program should pass **V** and **I** to the function **toResistance()** to obtain **R**. Then, swap the values of **V** and **I** by calling **swap()** and calculate **R** again using **toResistance()**. Print the value of **R** in both cases (to 2 decimal places, with **cout << fixed << setprecision(2) << I << endl;**).

Expected output:

Example
Please enter two values: 5.1 2 The value of one resistance is 2.55 The value of the other resistance is 0.39

Q-3.

Write a function `stringCompare()` to implement the comparison among strings. The rules for string comparison between string **s1** and **s2** are the followings:

1. Compare **s1** and **s2** character by character from the beginning of both strings.
2. When the i-1 characters of the two strings are identical, then:
   a. If i^{th} character of **s1** is larger than **s2**, **s1** is larger than **s2** regardless the remaining part;
   b. If i^{th} character of **s1** is smaller than **s2**, **s1** is smaller than **s2** regardless the remaining part;
   c. If i^{th} character of **s1** is equal to **s2**, continue the operation on $(i+1)^{th}$ character until one string ends;
3. When one string ends, and there is still no result for character comparison, the string with longer length is larger. If the lengths of the two strings are identical, the two strings are equal.
4. The function returns `1` if **s1** is larger than **s2**, `-1` if **s1** is smaller than **s2** and `0` if they are identical.
5. Call the function `stringCompare()` in the `main()` function to compare the two input strings.

Note.

I. Assume the maximum length of strings is less than 100.

II. You should only use library `<cstring>`, NOT `class <string>`.

Expected output:

Example 1

```
Enter the first string:
qwert
Enter the second string:
qwer
The first string is larger.
```

Example 2

```
Enter the first string:
Qwer
Enter the second string:
awer
The second string is larger.
```

Example 3

```
Enter the first string:
Qwer
Enter the second string:
Qwer
The two strings are equal.
```