CS216: Introduction to Software Engineering Techniques (Summer, 2020)
Lab Assignment 3
(20 points)

Today's Date: Tuesday, June 16

Due Date: Friday, June 19

The purpose of this lab assignment is

- Getting familiar with Unix/Linux environment
- Learning how to compile and run a C++ program with Unix/Linux environment
- Learning how to use command line arguments
- Reviewing recursion and using recursion to solve problems

Requirement:

1. In the terminal window (either from "terminal" application in the GUI-based Linux or from connecting your Virtual Machine through SSH client), make the CS216 directory, which you created in `Lab1`, your current working directory.

2. Create a directory underneath the CS216 directory named `Lab3` and make the `Lab3` directory your current working directory.

3. Without opening a browser, use command `curl` to download a zip file named `lab3Demo.zip` from the link (http://www.cs.uky.edu/~yipike/CS216/Lab3Source_sr.zip) and save the file into your current working directory **`~/CS216/Lab3`** by typing:

```
$ curl -O http://www.cs.uky.edu/~yipike/CS216/Lab3Source_sr.zip
```

4. Unzip the file you downloaded from step 3 using the command:

```
$ unzip Lab3Source_sr.zip
```

It contains four files: **`lab3input.txt`**, **`Lab3input.txt, Lab3.cpp, smartReverse.h, and smartReverse.cpp`**.

5. You can type the following command at the prompt **$** to check the content of the file `lab3input.txt`:

```
$ cat lab3input.txt
```

It will display the following text lines:

```
CS215 or CS216
race car
evil olive
stack cats
625 1225 2025 3025
```

```
   was it a car or a cat I saw
   never odd or even
   a nut for a jar of tuna
```

6. Then, type the following command at the prompt **$**:

```
$ rev lab3input.txt
```

It will display the following lines, which contains the text lines from the input file but in the reversed order at each line:

```
   612SC ro 512SC
   rac ecar
   evilo live
   stac kcats
   5203 5202 5221 526
   was I tac a ro rac a ti saw
   neve ro ddo reven
   anut fo raj a rof tun a
```

Type the following command at the prompt **$**:

```
$ rev lab3input.txt > lab3output.txt
$ cat lab3output.txt
```

This time, the reversed text lines from lab3input.txt did not display on the screen, instead they were written into a file named lab3output.txt by using the "**>**", which redirects the standard output to a file.

7. Then, type the following command at the prompt **$**, then click "return" key.

```
$ rev
```

The shell will be expecting your input from the keyboard. Repeatedly input some text at each line, then click "return" key. The command "**rev**" will reverse the order of characters in every line you just input from the keyboard. Until you click "**Ctrl-d**"(click both control key and the key for letter d at the same time), it will quit the **rev** command. (Note, Under Unix, the terminal will interpret **Ctrl+d** (on an empty line) as the end of input; further reading from **standard input stream** will set **EOF**, which stops the while loop in Lab3.cpp you downloaded from step 3). You can type the following command at the prompt **$** to check the manual page of the command **rev**:

```
$ man rev
```

The **rev** utility copies the specified files to the standard output, reversing the order of characters in every line. If no files are specified, the standard input (from keyboard) is read, and then reversed.

8. Next, you need to take a look at the definition of a class, named **`smartReverse`**. Its declaration is in header file named **`smartReverse.h`**, and its implementation is in .cpp file named **`smartReverse.cpp`**. You can open these two files at the same time to get a complete "picture by typing the following command at the prompt **`$`**:

```
$ vim –O smartReverse.h smartReverse.cpp
```

Please note that the option is "dash uppercase O" (it is not a zero) when you type the above command.

Now you can see that **`smartReverse.cpp`** is not complete, and this is the only source file you need to work on for this Lab Assignment: provide the implementation of each member function of the class named **`smartReverse`**. There are different ways to reverse a string, for example, you can use a loop to scan the string from last character back to the first character(as the instructor did during Monday's Lecture); or you can define a recursive function to reverse a string; or you can do it with the **`Last-In-First-Out`** feature of a stack: pushing the characters of a string into the stack, then popping off the stack to reverse the string. Each of the following **`THREE`** member functions is going to return a string, which is in the reverse order of the string stored in the private data member, named str. And you are required to implement them using **`THREE`** different ways:

```
// return a reversed string of str
// using a loop to implement
// Note that str has not been changed
string rev() const;

// return a reversed string of str
// using recursion to implement
// Note that str has not been changed string
rev_recursive() const;

// return a reversed string of str
// using a stack to implement
// Note that str has not been changed
string rev_stack() const;
```

9. Note that for this Lab assignment, **DO NOT** modify **`smartReverse.h`**!!! The SECOND source file you need to work on for this Lab Assignment: complete the Lab3.cpp. Open the Lab3.cpp file in your preferred text editor, and start to write the code following the comments of "// Your code starts here…", which is at the line number **`40`** in the original source code you download at step 3. And the code block, which you need to work on, end at the line number **`52`** in the original source code, with the comments of "// Your code ends here.". (Please note that

you can write as many lines of code as you need, the line numbers in the original source code do not have any impact on the lines you need to write on). The code block should help you generate the output, which can match the sample output at step 10. Take a look at the sample output at step 10 before you complete this block of code, and this is the only code block you need to complete in the source file, named **`Lab3.cpp`**.

After you finish writing the code, try to compile it by typing the command at the prompt $:

```
$ g++ Lab3.cpp smartReverse.cpp -o Lab3
```

If your source file passes the compilation, it will generate an executable program named **`Lab3`** under your current working directory Lab3.

10. Try to run your program after it passes compilation:

```
$ Lab3
```

You will get the message "Lab3: Command not found."

Try to execute your program again, but specify it is under your current working directory:

```
$ ./Lab3
```

The following is a sample output of running your program **`./Lab3`**:

```
Please input a string to watch its magic:
A↲
The original string you type is: A
It has been reversed using a loop: A
It has been reversed by recursion: A
It has been reversed using a stack: A

Please input a string to be reversed:
CS 216↲
The original string you type is: CS 216
It has been reversed using a loop: 612 SC
It has been reversed by recursion: 612 SC
It has been reversed using a stack: 612 SC


Please input a string to watch its magic:
Ctrl-d
```

Note that the user input is shown in blue and ↲ represents the return key. Ctrl-d represents when user clicks both control key and the key for letter d at the same time. And ↲ and Ctrl-d do not display on the screen, they are the keys the user clicks. (How do you know the user clicks "Ctrl-d" from the C++ program? Remember, you can watch for it by checking the EOF flag of your standard input stream: `cin.eof()`)

Congratulations! You have executed your C++ program from the command line.

11. You can also run your executable program, named **`Lab3`** with one command line argument, which represents the input file name: **`lab3input.txt`**: (compare the result from step 5)

```
$ ./Lab3 lab3input.txt

   Original (from file)  --> CS215 or CS216
   Reverse using a loop  --> 612SC ro 512SC
   Reverse by recursion  --> 612SC ro 512SC
   Reverse using a stack --> 612SC ro 512SC

   Original (from file)  --> race car
   Reverse using a loop  --> rac ecar
   Reverse by recursion  --> rac ecar
   Reverse using a stack --> rac ecar

   Original (from file)  --> evil olive
   Reverse using a loop  --> evilo live
   Reverse by recursion  --> evilo live
   Reverse using a stack --> evilo live

   Original (from file)  --> stack cats
   Reverse using a loop  --> stac kcats
   Reverse by recursion  --> stac kcats
   Reverse using a stack --> stac kcats

   Original (from file)  --> 625 1225 2025 3025
   Reverse using a loop  --> 5203 5202 5221 526
   Reverse by recursion  --> 5203 5202 5221 526
   Reverse using a stack --> 5203 5202 5221 526

   Original (from file)  --> was it a car or a cat I saw
   Reverse using a loop  --> was I tac a ro rac a ti saw
   Reverse by recursion  --> was I tac a ro rac a ti saw
   Reverse using a stack --> was I tac a ro rac a ti saw

   Original (from file)  --> never odd or even
   Reverse using a loop  --> neve ro ddo reven
   Reverse by recursion  --> neve ro ddo reven
   Reverse using a stack --> neve ro ddo reven
```

```
Original (from file)  --> a nut for a jar of tuna
Reverse using a loop  --> anut fo raj a rof tun a
Reverse by recursion  --> anut fo raj a rof tun a
Reverse using a stack --> anut fo raj a rof tun a
```

12. Then zip together the files: Lab3.cpp, smartReverse.h, smartReverse.cpp, Lab3input.txt and Lab3output.txt by typing the command at the prompt $:

```
$ zip Lab3.zip Lab3.cppsmartReverse.h smartReverse.cpp
Lab3input.txt Lab3output.txt
```

Submission

Open the link to Canvas LMS (https://uk.instructure.com/), and log in to your account using your linkblue user id and password. Please submit your file (Lab3.zip) through the submission link for "Lab3". You may transfer your filenamedLab3.zip from your VM to your local computer via BitVise, or NoMachine, or scp command☺

Grading (20 points + Demo Bonus 3 points)

1. Attend the lab session or have a documented excused absence. (5 points)
2. Finish step 5andgenerate the file named Lab3output.txtcorrectly (1 point)
3. Complete the C++ source files named smartReverse.cpp and Lab3.cpp, which correctly solves the problem. (14points.You get zero if your program cannot pass compilation)
   - implement the constructor of smartReverse class correctly (1 point)
   - implement the getString()member function of smartReverse class correctly (1 point)
   - implement the setString()member function of smartReverse class correctly (1 point)
   - implement the rev()member function of smartReverse class correctly (2 points)
   - implement the rev_recursive()member function of smartReverse class correctly (3 points)
     (if your function is correct, however you did not use recursion, you lose 3 points)
   - implement the rev_stack()member function correctly (3 points)
     (if your function is correct, however you did not use stack, you lose 3 points)
   - implement the code block specified in main() function correctly (3 points)

Bonus: Demonstrate your program to your TA and answer TA's questions. (3 points)

(Late assignment will be reduced 10% for each day that is late. The assignment will not be graded (you will receive zero) if it is more than 3 days late. Note that a weekend counts just as regular days. For example, if an assignment is due Friday and is turned in Monday, it is 3 days late.)

Reference:

If you are going to choose **`Vim`** as your editor to work for your C++ source code, you can download a runtime configuration file for your `Vim`, which contains basic configuration for working with C++ code. Without opening a browser, use command **curl** to download the configuration file, named **`.vimrc`** from the link (http://www.cs.uky.edu/~yipike/CS216/.vimrc) and save the file into your home directory by typing the following command from your home directory **~**:

```
$ curl –O http://www.cs.uky.edu/~yipike/CS216/.vimrc
```

Note that the configuration file name starts with a dot, you can check if this file is actually under your home directory by typing the command **`ls`** with the option of **`–a`** as shown below:

```
$ ls -a
```