Lab 9

Submit Assignment

Due Sunday by 11:59pm **Points** 100 **Submitting** a file upload **File Types** zip

Stack and Queue STL Containers

Goals

- Implement linear data structures using STL containers

In this lab, we will create two data structures using STL containers: “stack” and “queue”.

Requirements

Note: “deque” STL container is not allowed, for the simplicity for graders.

Queue

Queue is a first-in-first-out (FIFO) data structure.

For the Queue container, we will implement a program that simulates a buffer. The simulation needs a function that randomly generate number. And the buffer simulation starts with no value in the buffer.

At the start of simulation, the menu should ask user:

- how many rounds will be simulated.
- the percentage chance to put a randomly generated number at the end of buffer.
- the percentage chance to take out a randomly generated number at the front of buffer.

Note: For all the percentages, the “%” character is not required from user inputs. For example, if user input 25, it means 25%, and the number “25” is stored in the variable, not “25%”, or “0.25”.

Once all the user input is validated and stored in variables, the buffer simulation will start.

The following is the flow of each round in the simulation:

1. Generate a random number from 1 – 1000 called **N**.
2. Appending number: Generate another random number from 1 – 100, if the outcome is less than or equal to the user specified percentage for adding value (Ex: 25), then append the number **N** into the queue.
3. Removing numbers: Generate another random number from 1 – 100, if the outcome is less than or equal to the user specified percentage for removing value (Ex: 25), then remove a number from the front of queue.
4. Output the values in the buffer, and then output the length of the buffer.
5. Output the average length of buffer.

Equation for the average length of buffer:

$AL_i = (AL_{i-1} * (i - 1) + L_i) / i$

where AL_i and AL_{i-1} are the average length in the i^{th} and $(i-1)^{th}$ round, L_i is buffer length in the i^{th} round.

Note: The average length should be double/float data type, so when printed, it should have decimals

Note: A general rule of thumb, if the buffer has higher percentage of adding value than removing value, the average length of buffer should increase; while if the buffer has lower percentage of removing value than adding value, the average length of buffer should decrease.

Stack

Use a stack to create a function that creates a **palindrome**, which is a string that is the same forwards and backwards. For example, “racecar” is a palindrome. The function should receive a string from user input, and return the string with the palindrome, by first adding the values of the original string sequentially to the stack, then popping off characters one by one and printing the character. The popped off string would be in reverse order when you pop them off the stack. For example, if user inputs "hello", your program will print the palindrome "helloolleh".

Menu

Create a menu for the user to test the buffer and create a palindrome.

For the queue, prompt the user to enter the two percentages, and the total number of rounds. Then, display the results to the console in each round.

For the stack, prompt the user to enter a string. Create the palindrome and then display it.

What to submit

- All the program files including header and source files (.cpp/.hpp)
- Makefile

Important: Put all the files in a single .zip file and submit it on Canvas.

Grading

- Programming style -10%
- Implement buffer simulation using <queue> - 30%
- Implement palindrome function using <stack> - 30%

- Create a menu program – 30%

Lab 9		
Criteria	**Ratings**	**Pts**
Programming style		10.0 pts
Implement buffer simulation using <queue>		30.0 pts
Implement palindrome function using <stack>		30.0 pts
Create a menu program		30.0 pts
		Total Points: 100.0