Assignment # 03
CS 2001 – Data Structures
Fall 2021

Topics: Stack and Queue ADT

General Guidelines

1. Write neat and clean code. Avoid any memory leaks and dangling pointers while implementing the scenarios required in this assignment.
2. You can lose the marks if conventions are not strictly followed.
3. Peer plagiarism and the late submissions are strictly not allowed. In case, zero marks will be awarded for whole assignment
4. Total Marks: 100

Submission Guidelines

1. Your assignment submission must be in hardcopy (i.e., handwritten, or printed form) and a scanned version should also be uploaded on the SLATE within the given deadline.
2. Analytical and mathematical questions can be handwritten, while code questions should preferably be computer-typed [however, handwritten solutions are also allowed for this assignment only]
3. **Deadline (Tuesday, 09th November till 04:00 pm)**

Task1: Sequence Mutation using Stack ADT [20 marks]

Consider a scenario where a communication App changes the sequences of words in the message that a sender sends to receiver. However, the changing mechanism is simple though, it removes the punctuation (if any) from the message. For example, a message like below:

"Kindness is what defines humanity!!" would get changed to *"Humanity defines what is kindness"*

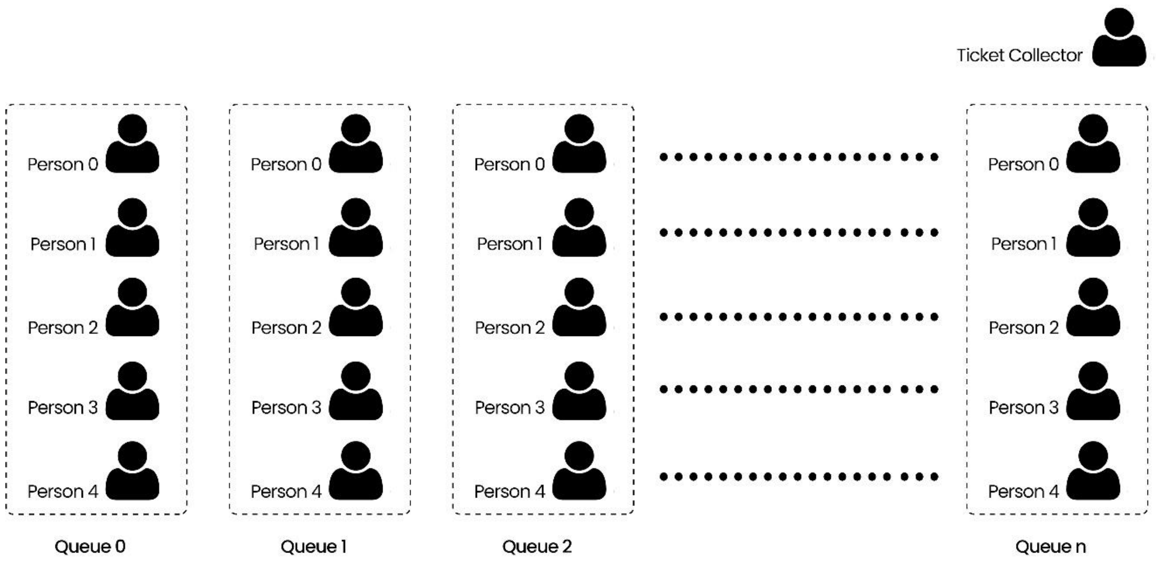
Write a C++ snippet to implement the scenario. Your implementation must use the Stack ADT based on pointer implementation in separate file such as stack.h

Task2: Tower of Hanoi using Stack ADT [30 marks]

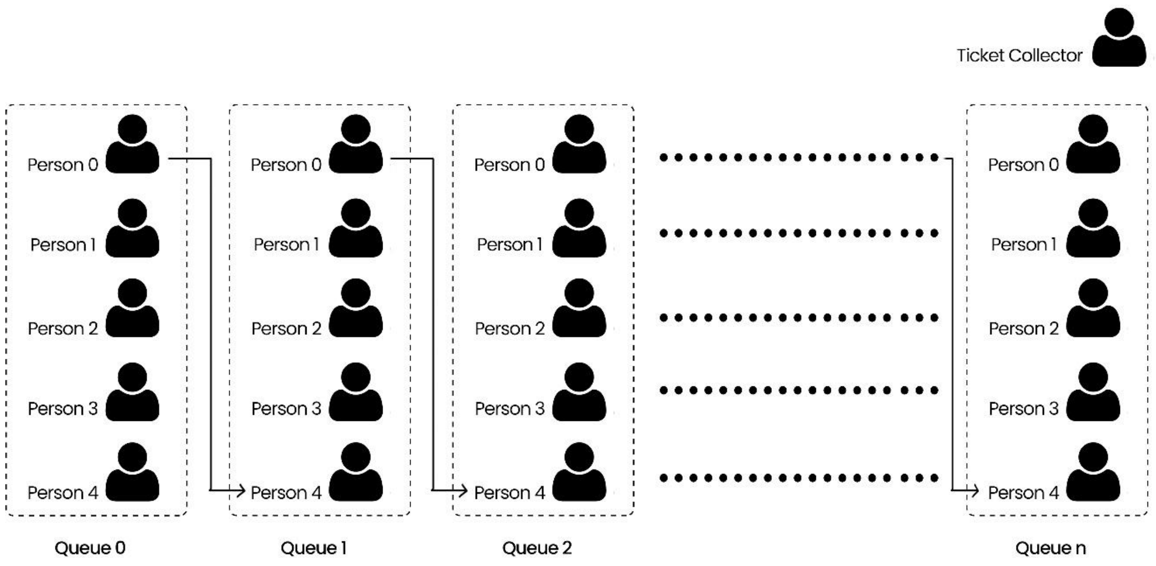
Tower of Hanoi is a mathematical puzzle where we have three rods and n disks. The objective of the puzzle is to move the entire stack to another rod, obeying the following simple rules: Only one disk can be moved at a time. Consider this problem using three stacks representing the rod at a time. For further understanding, visit the following link and play it out using 5 disks and three rods.
https://www.mathsisfun.com/games/towerofhanoi.html
In this task, you need to implement the game using three stacks and 5 numbers (any sequence) representing the rods and disk required to play the game. Clearly show the state of each stack on console while solving this problem.

Task3: Movie Ticket Bookings using Queue ADT [50 marks]

In a world without digital bookings when people had to buy movie tickets. Imagine it's 1990s and a blockbuster movie has just released. To avoid longer queues and to keep track of queues outside the cinema, they have decided to divide the queue into sub-queues but due to limited resources the cinemas can only afford one ticket collector who is standing at the front of the last sub-queue. The ticket collector takes 2 seconds to process a person. The queues look like this:



The queues work in the following way:

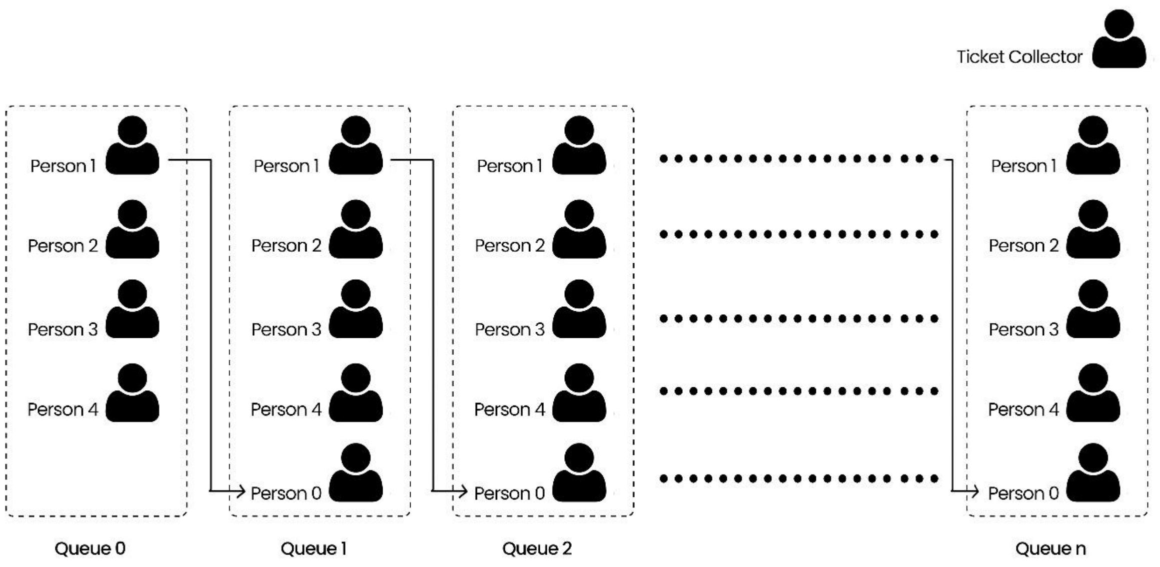


i.e: After processing "Person 0" of "Queue n", the queue moves forward like this:

1. "Person 0" of "Queue n-1" leaves his/her queue and enters in "Queue n".
2. "Person 0" of "Queue n-2" leaves his/her queue and enters in "Queue n-1".
3. "Person 0" of "Queue n-3" leaves his/her queue and enters in "Queue n-2".

4. This keeps happening all the way to queue 0 and at the end, “Person 0” of “Queue 0” leaves his/her queue and enters in “Queue 1”.

After the queue is moved one-step forward, here’s how it looks:



By the end, the ticket collector processes all the people in the queues so that everyone can enjoy the movie.

You are going to simulate the above explained process using Queues ADT in C++. To implement this program, you will also be creating a Template Queue Class all by yourself to keep it generic and to create queues of any-types. The flow of the program will be like this:

User inputs an integer and N queues (<int>) are created. After creating N queues, you will enqueue some number of persons. For your ease, you can use same number of persons in every queue (minimum number of people in a queue must be 7). The ticket collector starts to process persons in “Queue n” one by one until all the persons are processed OR all the queues are empty.

Note: You’re not allowed to any built-in libraries like vectors or queues.