# Lab 6: Stacks and Queues

CS2028C (003)
Group 10

Brendan Pollak
Spencer Kleeh
Jacob Butler
Aaliyah Loechner

March 17, 2021

**Objectives/Concepts**

The objective of this lab was to learn about stacks and queues, and their specific uses in C++. In task 1, we created a stack class using an array. This also required us to utilize our knowledge of C++ templates which we learned from last week's lab, as the class we created had to be a template. Task 2 asked us to create a single person version of the game "Towers of Hanoi". We used stacks to perform different actions within the game. Finally, in task 3 we created a queue class which is also a template, similarly to the stack class from task 1. The queue class uses a vector to track and store data pertaining to moves made in the game.

**Task 1**

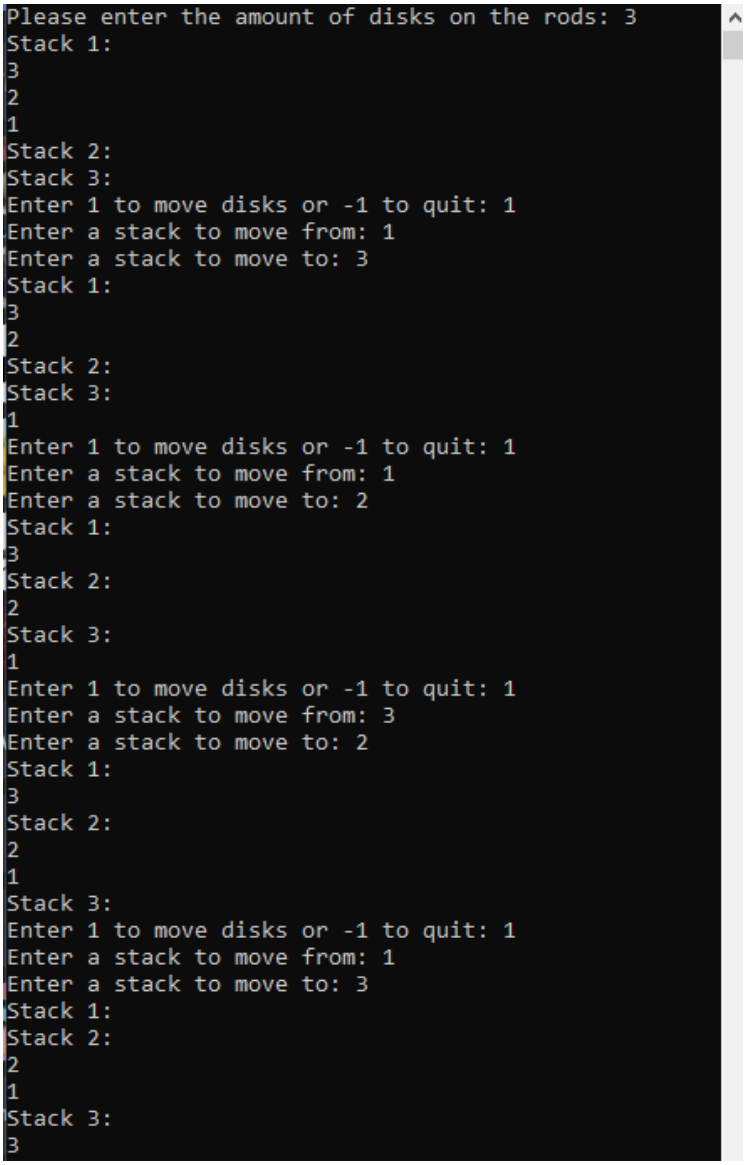
In this task, we created a stack class which was used throughout the entirety of this lab. This class was created using an array, which is initialized to have 3 stacks of n disks. Additionally, we were asked to write push, pop, top, length, and empty functions into the program in order for the implementation code to run the game properly. The code for implementing these functions was created in task 2 which can be found below.

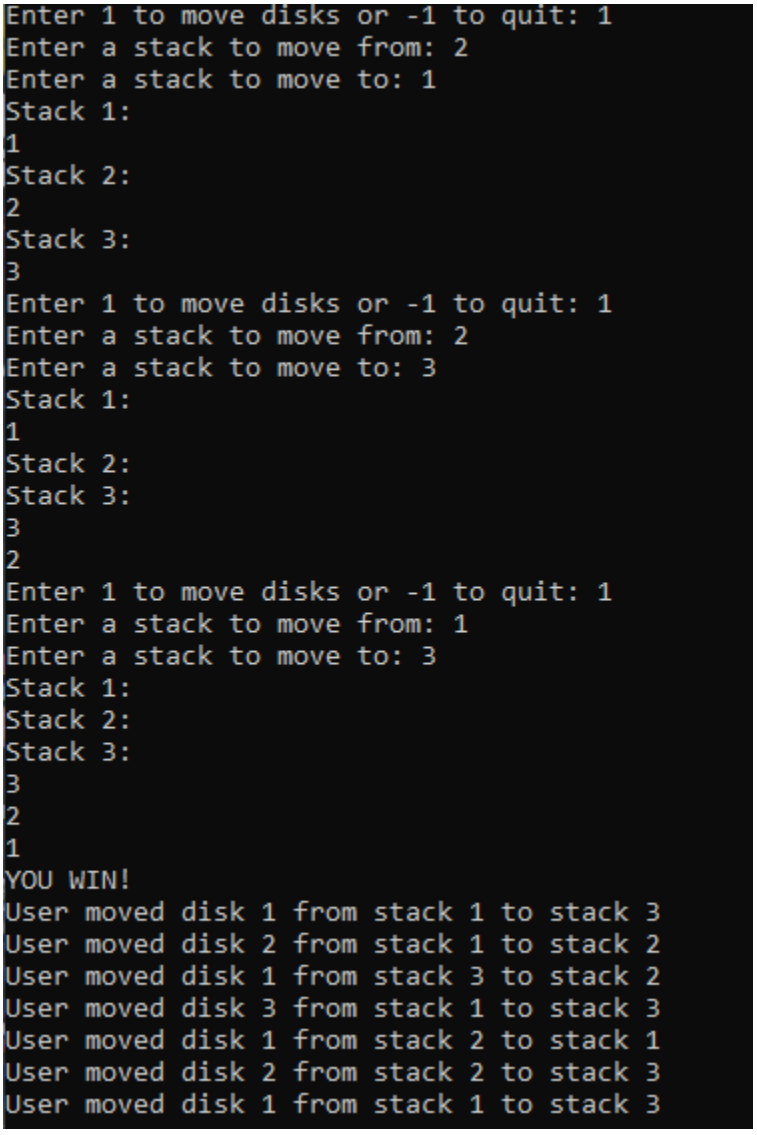
**Task 2**

```
Please enter the amount of disks on the rods: 3
Stack 1:
3
2
1
Stack 2:
Stack 3:
Enter 1 to move disks or -1 to quit: 1
Enter a stack to move from: 1
Enter a stack to move to: 3
Stack 1:
3
2
Stack 2:
Stack 3:
1
Enter 1 to move disks or -1 to quit: 1
Enter a stack to move from: 2
Enter a stack to move to: 3
Exception: cannot pop from empty array
Invalid move.
Stack 1:
3
2
Stack 2:
Stack 3:
1
Enter 1 to move disks or -1 to quit: 1
Enter a stack to move from: 1
Enter a stack to move to: 3
Not a valid move.
Stack 1:
3
2
Stack 2:
Stack 3:
1
Enter 1 to move disks or -1 to quit:
```

The image above thoroughly shows that invalid moves cannot be made. For example, the user cannot move anything from an empty stack, nor can the user move a value larger than what is on their source stack.

**Task 3**




The move data structure was designed in a fairly simple manner. The structure was created with 3 members, the source stack that the user pulls from, the target stack that the user moves to, and the pointer of the value being moved. This way, addition of strings allows for an easy and understandable output as seen above. Fortunately, adding this to the main code was easy. A queue instance was created and the created functions made holding and outputting each move by the user easy to output.