**Fully functional Queue class (50pts, +10 ec)**

**Project Description & Goal**

The goal of this project is to understand the basics of a Queue data structure. We will be building upon this project in the next iteration when we talk about Generics and Interfaces.

**Project Specifications**

* Create a class that acts as a Queue for integers
* Implement all required methods
* Create a menu in Main() that allows for each of the methods to be called on the queue.

**Required Methods**

* **public int DeQueue()**
  + Removes and returns the first element of the queue.
* **public void EnQueue (int val)**
  + Pushes val to the back of the queue.
* **public int Peek()**
  + Returns the front element of the queue. (do not remove it!)
* **public void Print()**
  + Prints the entirety of the queue (essentially for debugging, in practice we keep to the rule that a queue is FIFO only and that we cannot access the rest of the queue).

**Project Learning Objectives**

* Understand a queue data structure.
* Learn what it takes to create a data structure that can be used as a utility for your projects.

**Project Demonstrated Competencies**

1. Queue class is correct and all methods work appropriately.
2. Menu loops and allows the user to access each of the Queue methods.
3. Optional: Implement the search method.
   * public int Search(int val): Returns the position in the queue.
     + Front element = 0
     + Second element = 1
     + etc
4. Optional: Implement error handling for all methods.
   * What happens if I try to DeQueue () or Peek() on an empty queue?

**Rubric**

|  |  |  |
| --- | --- | --- |
|  | **Description of perfect implementation** | **Score** |
| Competency #1 | The queue class has all 4 methods and they work exactly as they should. | \_\_\_  25 |
| Competency #2 | The main menu allows the user to directly access all 4 methods. EnQueue() asks the user for input and then calls EnQueue(), Peek() and DeQueue () will print what they return, and print will print the entire queue. | \_\_\_  25 |
| Competency #3 | Search method correctly returns the position in the queue to the user. | \_\_\_  +7 |
| Competency #4 | Error handling prevents program from crashing when they try to remove an element from an empty queue. | \_\_\_  +3 |