Logo

Description automatically generated

**22/SU-COP-2805C-72035 Java Programming**

**Exercise 10.10**

Document Version: 0.1

Version Date: May 26, 2022

Created By: Johnathan Webster

# Document Version Control

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author | Rationale |
| 0.1 | 2022 May 26 | Johnathan Webster | First Draft |

# Document Purpose

The purpose of this document is to define the Queue() class.

# Technical Specifications

## Purpose of Technical Implementation

The purpose of this implementation is to have a class named Queue() that creates a queue that can be managed by the user.

## Technical Implementation Components

The class has two constructors. The first takes no arguments but will set the default queue size to a value of 8.

*public* *Queue*(){

        this(8);

    }

The second constructor takes one argument. The provided argument will specify the capacity the user would like for the queue.

*Queue*(int capacity){

        this.*capacity* = capacity;

        this.*elements* = *new* int[capacity];

    }

The class also has multiple methods that can be used to:

Add items to the queue and, if the array is not large enough, increase the size of the array:

*public* void *enqueue*(int addNum){

*if* (this.*size* >= this.*elements*.*length*){

            int[] temp = *new* int[this.*size* \* 2];

            System.*arraycopy*(this.*elements*, 0, temp, 0, this.*size*);

            this.*elements* = temp;

        }

        this.*elements*[size++] = addNum;

    }

Remove elements from the queue and arrange the array in the correct orientation:

*public* int *dequeue*(){

        int remvNum = this.*elements*[0];

        this.*size*--;

        //*assignes a loop control var of 0 to loop through array*

        //*then assignes the element at the next index location to the current index*

*for* (int items = 0; items < size; items++){

            this.*elements*[items] = this.*elements*[items + 1];

        }

*return* remvNum;

    }

Check to see if the queue is empty:

*public* boolean *isEmpty*(){

*return* this.*size* == 0;

    }

## Queue Implementation

The implementation of this class allows the user to create a new instance of a queue. The queue will store the values the user inputs and will keep them organize in a queue fashion. The class will also allow the user to keep track of the queue in the order the values are entered.