Bahria University, Islamabad

Department of Software Engineering

Data Structures & Algorithms Lab

(Spring-2024)

Teacher: RAHEELA AMBRIN

Student : Abdul Rafay

Enrollment : 01-131232-004

Lab Journal: 4

Date: 08 / 10 / 24

Comments:

Signature

Code:

All the code files are uploaded on GitHub: https://github.com/CharlieFour/DSA_Lab

You can check out the code on GitHub in Lab_03 folder.

Class file (used in both task):

```
#include <iostream>

template <class itemType>
class Queue
{
  private:
    int front;
    int rear;
    int capacity;
    int count;#include <iostream>

template <class itemType>
class Queue
{
    private:
        int front;
        int rear;
        int capacity;
        int count;#include <iostream>

template <class itemType>
class Queue
{
    private:
        int front;
        int rear;
        int capacity;
        int count;#include <iostream>

template <class itemType>
class Queue
{
    private:
        int front;
        int rear;
        int capacity;
        int count;
        itemType* items;

    public:
        Queue();
        Queue(int size);
        ~Queue();
        bool isEmpty() const;
```

```
template <class itemType>
Queue<itemType>::~Queue()
{
    delete[] items;
}

template <class itemType>
bool Queue<itemType>::isEmpty() const
{
    return (count == 0);
}

template <class itemType>
bool Queue<itemType>::isFull() const
{
    return (count == capacity);
}

template <class itemType>
void Queue<itemType>::insert(itemType item)
{
    if(!(isFull()))
    {
        rear = (rear + 1) % capacity;
        items[rear] = item;
        count++;
    }
    else
    {
        std::cerr << "Error: Queue Overflow" << std::endl;
    }
}

template <class itemType>
itemType Queue<itemType>::remove()
{
    if(!(isEmpty()))
    {
        front = (front + 1) % capacity;
        itemType item = items[front];
        count--;
        return item;
    }
else
    {
        std::cerr << "Error: Queue Underflow" << std::endl;
        return itemType();
    }
}
```

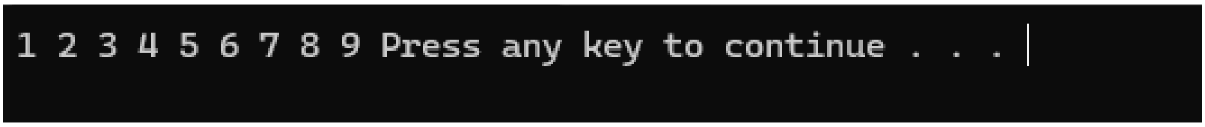
Task 01:

```
#include <iostream>
#include "../libraries/Queue.h"

using namespace std;

int main()
{
    Queue<int> queue;
    int number = 1;
    while (number < 10)
    {
        queue.insert(number);
        number++;
    }
    while (!queue.isEmpty())
    {
        cout << queue.remove() << " ";
    }
    system("pause");
    return 0;
}
```

Screen Shots:



```
1 2 3 4 5 6 7 8 9 Press any key to continue . . .
```

Task 02:

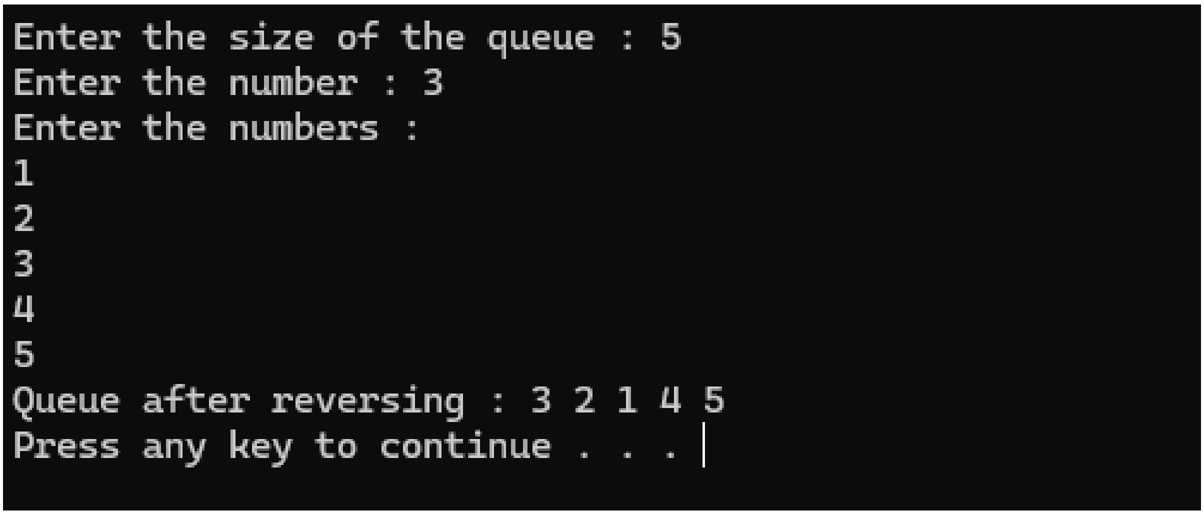
```
#include <iostream>
#include <algorithm> // for sorting array
#include "../libraries/Queue.h"

using namespace std;
int main()
{
    //Taking inputs
    int size, number;
    cout << "Enter the size of the queue : ";
    cin >> size;
    cout << "Enter the number : ";
    cin >> number;
    Queue<int> queue(size);
    int input;
    cout << "Enter the numbers : " << endl;
    for(int i = 0; i < size; i++)
    {
        cin >> input;
        queue.insert(input);
    }

    // Inserting item into the array
    int array[size];
    for (int i = 0; i < size; i++)
    {
        array[i] = queue.remove();
    }

    // Reversing the array
    int array2[number];
    for (int i = 0; i < number; i++)
    {
        array2[i] = array[i];
    }
    reverse(array2, array2 + number);
    for (int i = 0; i < number; i++)
    {
        array[i] = array2[i];
    }
// Inserting the reversed array back into the queue
    for (int i = 0; i < size; i++)
    {
        queue.insert(array[i]);
    }
    cout << "Queue after reversing : ";
    while (!queue.isEmpty())
    {
        cout << queue.remove() << " ";
    }
    cout << endl;
    system("pause");
    return 0;
}
```

Screen Shots:



```
Enter the size of the queue : 5
Enter the number : 3
Enter the numbers :
1
2
3
4
5
Queue after reversing : 3 2 1 4 5
Press any key to continue . . .
```