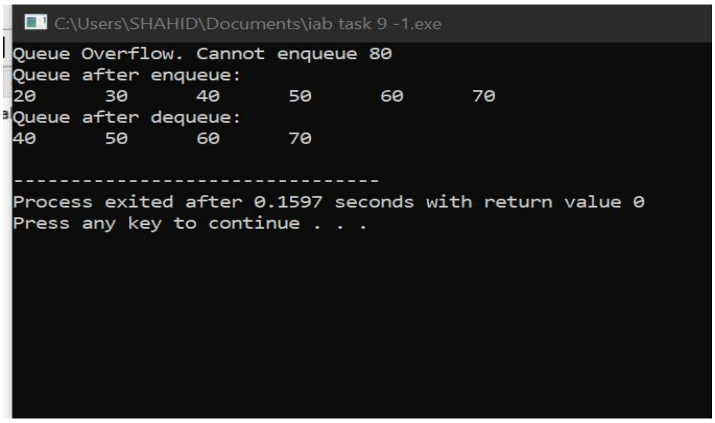
NAME	**SHAHID ZAMAN**
SAP ID	**55123**
LAB TASK	**(09)**
SUBJECT	**(DAS)**

QUESTION NO:01

```
#include<iostream>
using namespace std;

struct Node {
   int data;
   Node* next;
};

class Queue {
   Node *front, *rear;
   int currentSize;
   int maxSize; // Removed const qualifier
public:
   // Constructor to set the maximum size
   Queue(int max) {
      maxSize = max; // Set maxSize in the constructor body
      front = rear = 0;
      currentSize = 0;
   }

   void Enqueue(int data) {
      if (currentSize == maxSize) {
         cout << "Queue Overflow. Cannot enqueue " << data << endl;
         return;
      }

      Node* newnode = new Node;
      newnode->data = data;
      newnode->next = 0;

      if (front == 0) {
         front = rear = newnode;
      } else {
         rear->next = newnode;
         rear = newnode;
      }

      currentSize++;
   }

   void Dequeue() {
      if (front == 0) {
         cout << "Queue is empty\n";
         return;
      }

      Node* temp = front;
      front = front->next;
      delete temp;

      currentSize--;

      if (front == 0) { // Queue is now empty
         rear = 0;
      }
   }

   void Display() {
      Node *temp = front;
      while (temp != 0) {
         cout << temp->data << "\t";
         temp = temp->next;
      }
      cout << endl;
   }
};

int main() {
   Queue Q1(6); // Maximum size of the queue is set to
   Q1.Enqueue(20);
   Q1.Enqueue(30);
   Q1.Enqueue(40);
   Q1.Enqueue(50);
   Q1.Enqueue(60);
   Q1.Enqueue(70);
   Q1.Enqueue(80); // This will trigger an overflow
   cout << "Queue after enqueue:" << endl;
   Q1.Display();
   Q1.Dequeue();
   Q1.Dequeue();
   cout << "Queue after dequeue:" << endl;
   Q1.Display();

   return 0;
}
```

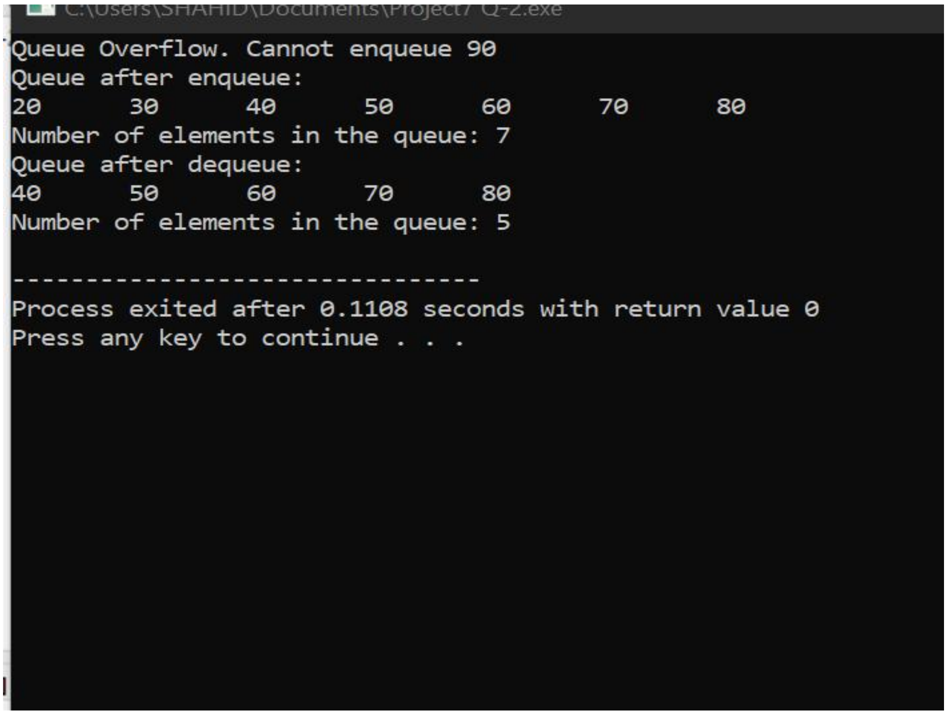


```
Queue Overflow. Cannot enqueue 80
Queue after enqueue:
20      30      40      50      60      70
Queue after dequeue:
40      50      60      70

--------------------------------
Process exited after 0.1597 seconds with return value 0
Press any key to continue . . .
```

QUESTION NO :02

```
#include<iostream>
using namespace std;

struct Node {
   int data;
   Node* next;
};

class Queue {
   Node *front, *rear;
   int size;
   int maxSize;
public:
   // Constructor to set the maximum size
   Queue(int max) {
      maxSize = max;
      front = rear = 0;
      size = 0;
   }

   void Enqueue(int data) {
      if (size == maxSize) {
         cout << "Queue Overflow. Cannot enqueue " << data << endl;
         return;
      }

      Node* newnode = new Node;
      newnode->data = data;
      newnode->next = 0;

      if (front == 0) {
         front = rear = newnode; // The queue was empty, new node is both front and
rear
      } else {
         rear->next = newnode;
         rear = newnode;
      }

      size++; // Increase the current size count
   }

   void Dequeue() {
      if (front == 0) {
         cout << "Queue is empty\n";
         return;
      }

      Node* temp = front;
      front = front->next;
      delete temp;

      size--;

      if (front == 0) { // Queue is now empty
         rear = 0;
      }
   }

   void Display() {
      Node *temp = front;
      while (temp != 0) {
         cout << temp->data << "\t";
         temp = temp->next;
      }
      cout << endl;
   }


   int getSize() {
      return size;
   }
};

int main() {
   Queue Q1(7); // Maximum size of the queue is set to 7
   Q1.Enqueue(20);
   Q1.Enqueue(30);
   Q1.Enqueue(40);
   Q1.Enqueue(50);
   Q1.Enqueue(60);
   Q1.Enqueue(70);
   Q1.Enqueue(80);
   Q1.Enqueue(90); // This will trigger an overflow
   cout << "Queue after enqueue:" << endl;
   Q1.Display();


   cout << "Number of elements in the queue: " << Q1.getSize() << endl;

   Q1.Dequeue();
```

```
   Q1.Dequeue();
   cout << "Queue after dequeue:" << endl;
   Q1.Display();


   cout << "Number of elements in the queue: " << Q1.getSize() << endl;

   return 0;
}
```

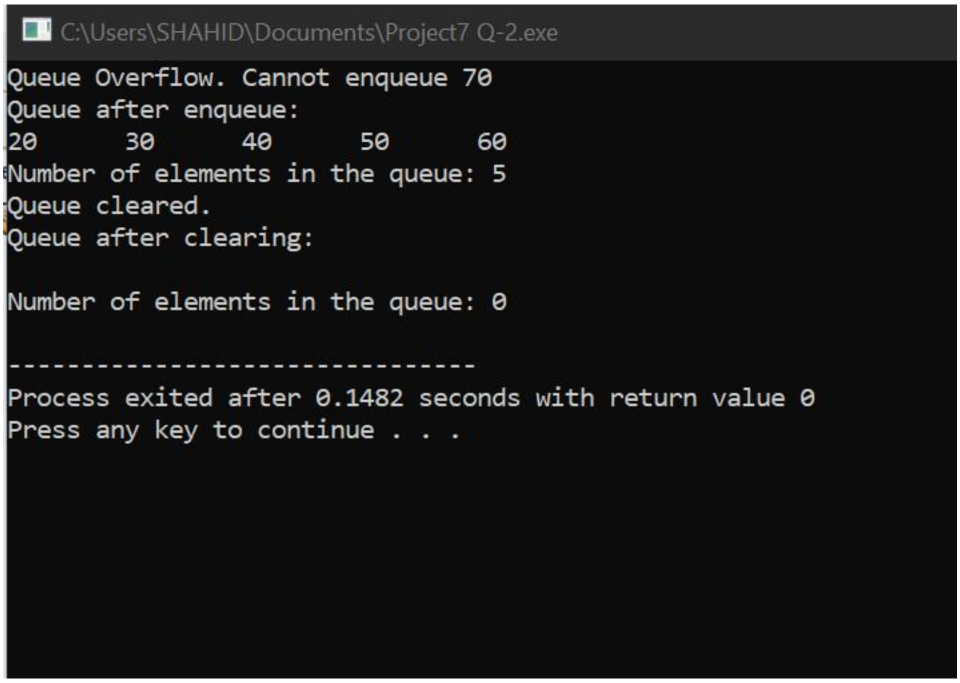


```
Queue Overflow. Cannot enqueue 90
Queue after enqueue:
20      30      40      50      60      70      80
Number of elements in the queue: 7
Queue after dequeue:
40      50      60      70      80
Number of elements in the queue: 5

--------------------------------
Process exited after 0.1108 seconds with return value 0
Press any key to continue . . .
```

QUESTION NO :03

```
#include<iostream>
using namespace std;

struct Node {
   int data;
   Node* next;
};

class Queue {
   Node *front, *rear;
   int size;
```

```
  int maxSize;
public:
  // Constructor to set the maximum size
  Queue(int max) {
    maxSize = max;
    front = rear = 0;
    size = 0; //  the queue is empty
  }

  void Enqueue(int data) {
    if (size == maxSize) {
      cout << "Queue Overflow. Cannot enqueue " << data << endl;
      return;
    }

    Node* newnode = new Node;
    newnode->data = data;
    newnode->next = 0;

    if (front == 0) {
      front = rear = newnode;
    } else {
      rear->next = newnode; /
      rear = newnode;
    }

    size++; // Increase the current size count
  }

  void Dequeue() {
    if (front == 0) {
      cout << "Queue is empty\n";
      return;
    }

    Node* temp = front;
    front = front->next;
    delete temp;

    size--; // Decrease the current size count

    if (front == 0) { // Queue is now empty
      rear = 0;
    }
  }


  void Clear() {
    while (front != 0) {
      Dequeue();
    }
    cout << "Queue cleared.\n";
  }

  void Display() {
    Node *temp = front;
    while (temp != 0) {
            cout << temp->data << "\t";
            temp = temp->next;
        }
        cout << endl;
    }


    int getSize() {
        return size;
    }
};

int main() {
    Queue Q1(5); // Maximum size of the queue is set to 5
    Q1.Enqueue(20);
    Q1.Enqueue(30);
    Q1.Enqueue(40);
    Q1.Enqueue(50);
    Q1.Enqueue(60);
    Q1.Enqueue(70); // This will trigger an overflow
    cout << "Queue after enqueue:" << endl;
    Q1.Display();

    cout << "Number of elements in the queue: " << Q1.getSize() << endl;

    Q1.Clear(); // Clear the entire queue
    cout << "Queue after clearing:" << endl;
    Q1.Display();

    cout << "Number of elements in the queue: " << Q1.getSize() << endl;

    return 0;
}
```



```
Queue Overflow. Cannot enqueue 70
Queue after enqueue:
20      30      40      50      60
Number of elements in the queue: 5
Queue cleared.
Queue after clearing:

Number of elements in the queue: 0

--------------------------------
Process exited after 0.1482 seconds with return value 0
Press any key to continue . . .
```