Name: Muhammad Zohaib khan

Reg# BSE203003

Subject: DS Lab

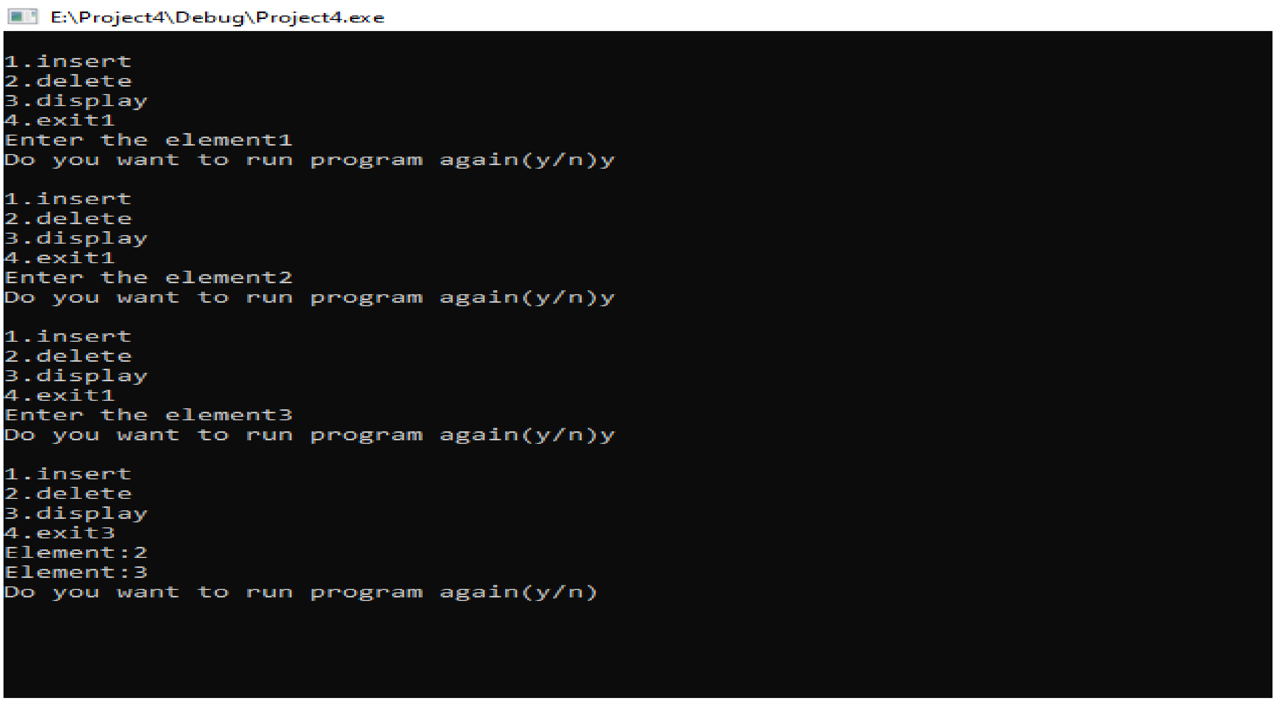
Instructor: Mam Sara Ibrahim

Date:15 -11-2021

Practice Task 1:

```
#include<iostream>
using namespace std;
class Queue
{
	int *elements;
	int size;
	int rear, front;
	int maxsize;
public:
	Queue(int msize)
	{
		maxsize = msize;
		size = maxsize;
		elements = new int[size];
		rear = -1, front = -1;
	}
	/////////////////////////////////// Destructor
	~Queue()
	{
		delete[]elements;
	}
	/////////////////////////////////// Output the Queue structure
	void showStructure()
	{
		if (!isEmpty())
		{
			for (int i = 0; i < rear; i++)
			{
				cout << "Element:" << elements[i] << endl;
			}
		}
		else
		{
			cout << "Display: No items to be displayed. Queue is empty\n";
		}
	}
	////////////////////////////////// Queue manipulation operations
	// Insert in queue
	void Enqueue(int newDataItem)
	{
		if (!isFull())
		{
			if (front == -1)
			{
				front = 0;
			}
			elements[rear] = newDataItem;
			rear++;
		}
		else
		{
			cout << "Insert: Cannot insert more items. List is full\n";
		}
	}
	// Remove data item
	void Dequeue()
	{
		if (!isEmpty())
		{
			cout << "deleted" << elements[++front];
```

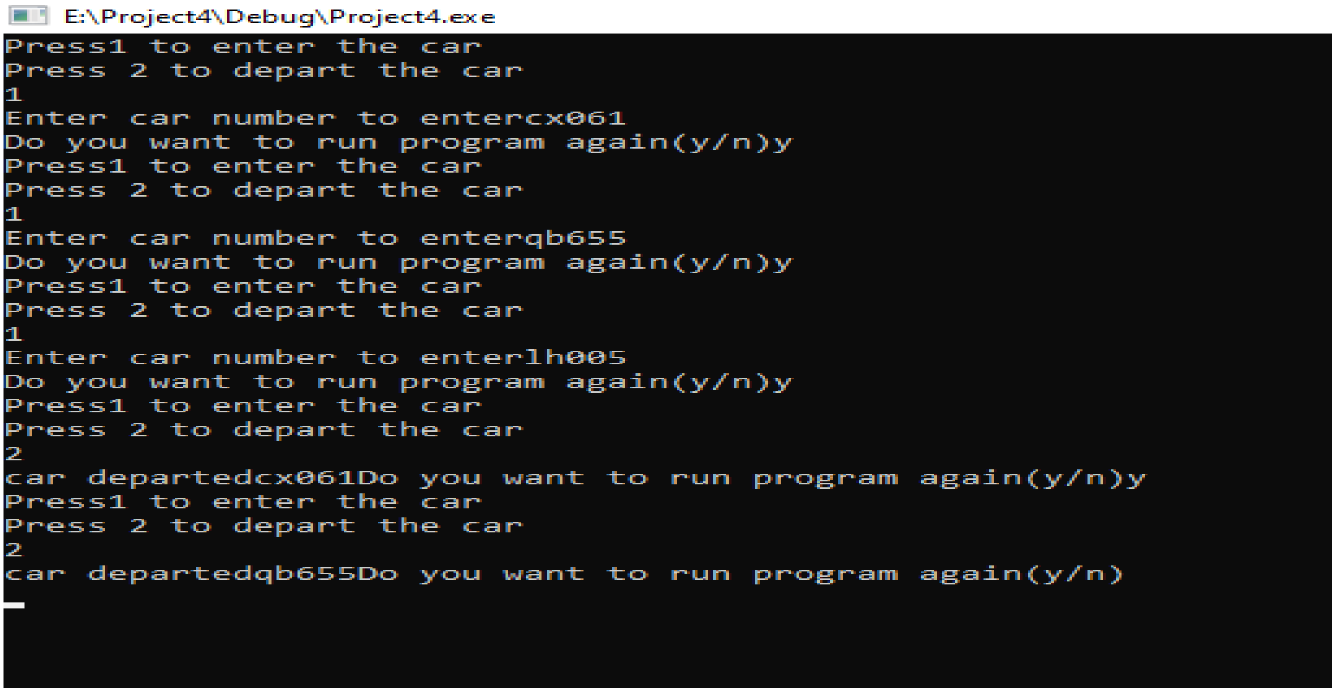
```
                }
                else
                {
                        cout << "Remove: Cannot remove the item. List is empty\n";
                }
        }

        bool isEmpty()
        {
                if (front == -1 && rear == -1)
                {
                        return true;
                }
                else { return false; }
        }

        bool isFull()
        {
                if (rear == maxsize) { return true; }
                else { return false; }
        }
};
int main()
{
        Queue q(10);
        char ch;
        int choice;
        do
        {
                cout << "\n1.insert \n2.delete \n3.display \n4.exit";
                cin >> choice;
                switch (choice)
                {
                        int add;
                case 1:cout << "Enter the element";
                        cin >> add;
                        q.Enqueue(add);
                        break;
                case 2:
                        q.Dequeue();
                        break;
                case 3:
                        q.showStructure();
                        break;
                case 4:
                        exit(0);
                        break;
                }
                cout << "Do you want to run program again(y/n)";
                cin >> ch;
        } while (ch == 'y');
}
```


```
E:\Project4\Debug\Project4.exe
1.insert
2.delete
3.display
4.exit1
Enter the element1
Do you want to run program again(y/n)y

1.insert
2.delete
3.display
4.exit1
Enter the element2
Do you want to run program again(y/n)y

1.insert
2.delete
3.display
4.exit1
Enter the element3
Do you want to run program again(y/n)y

1.insert
2.delete
3.display
4.exit3
Element:2
Element:3
Do you want to run program again(y/n)
```

Practice Task 2:

```
#include<iostream>
#include<string>
using namespace std;
class parking {
	string *car_id;
	int maxsize;
	int rear, front;
public:
	parking(int mxsize)
	{
		car_id = new string[mxsize];
		maxsize = mxsize;
		front = rear = -1;
	}
	void enqueue(string car)
	{
		if (!isfull())
		{
			car_id[++rear] = car;
		}
		else { cout << "Parking is full"; }
	}
	void dequeue()
	{
		if (!isempty())
		{
			cout << "car departed" << car_id[++front];
		}
		else { cout << "Parking is empty"; }
	}
	bool isempty()
	{
		if (front == -1 && rear == -1)
		{
			return true;
		}
		else { return false; }
```

```
        }
        bool isfull()
        {
                if (rear == maxsize)
                {
                        return true;

                }
                else { return false; }
        }
};
int main()
{
        parking p(10);
        int choice;
        string addcar;
        char ch;
        do {
                cout << "Press1 to enter the car " << endl;
                cout << "Press 2 to depart the car " << endl;
                cin >> choice;
                switch (choice)
                {
                case 1:

                        cout << "Enter car number to enter";
                        cin >> addcar;
                        p.enqueue(addcar);
                        break;
                case 2:
                        p.dequeue();
                        break;
                }
                cout << "Do you want to run program again(y/n)";
                cin >> ch;
        } while (ch == 'y');
}
```



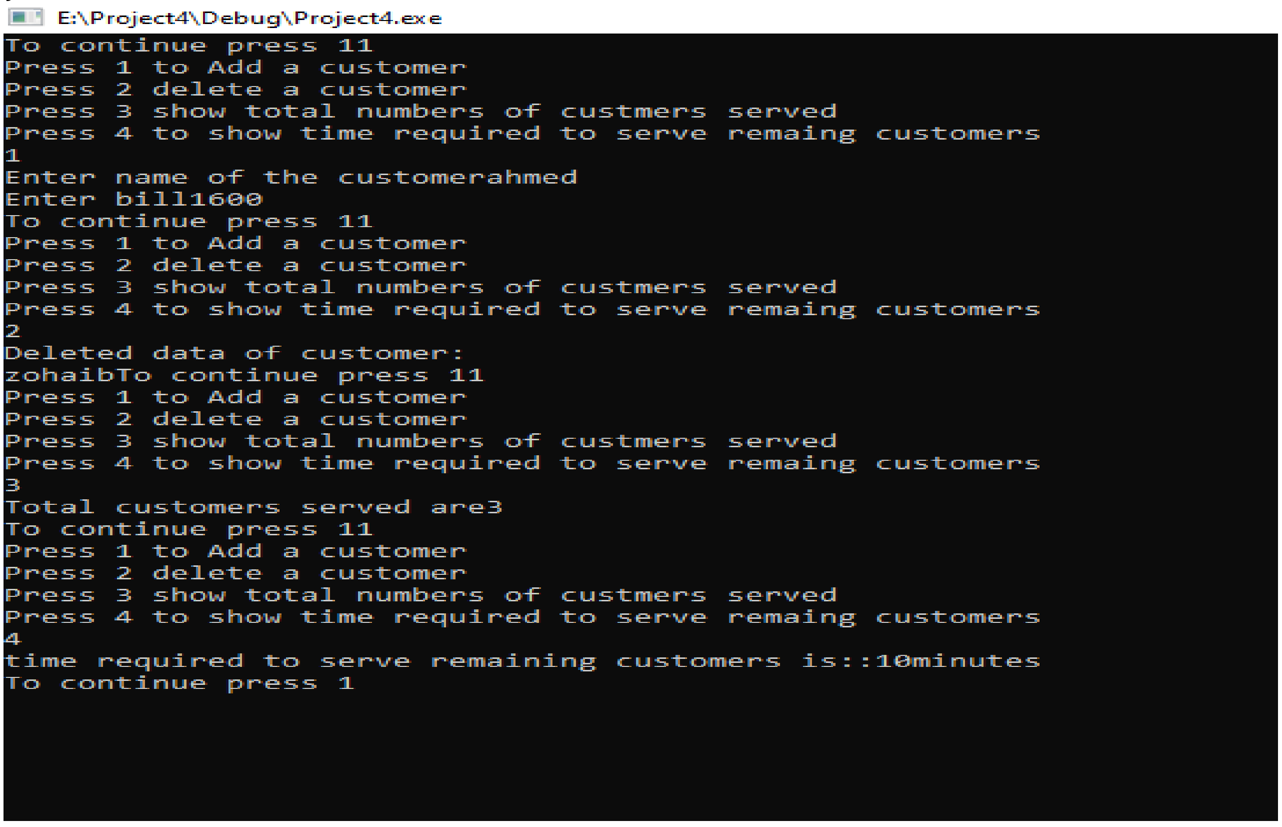
```
E:\Project4\Debug\Project4.exe
Press1 to enter the car
Press 2 to depart the car
1
Enter car number to entercx061
Do you want to run program again(y/n)y
Press1 to enter the car
Press 2 to depart the car
1
Enter car number to enterqb655
Do you want to run program again(y/n)y
Press1 to enter the car
Press 2 to depart the car
1
Enter car number to enterlh005
Do you want to run program again(y/n)y
Press1 to enter the car
Press 2 to depart the car
2
car departedcx061Do you want to run program again(y/n)y
Press1 to enter the car
Press 2 to depart the car
2
car departedqb655Do you want to run program again(y/n)
```

<u>**Practice Task 3:**</u>

```
#include<iostream>
#include<string>
using namespace std;
class customer
{
	string name;
	int id;
	int bill;
	int time;
	int count;
public:
	customer()
	{
		id = 0;
		time = 0;
		count = 0;
	}
	void setdata()
	{
		cout << "Enter name of the customer";
		cin >> name;
		cout << "Enter bill";
		cin >> bill;
		id++;


}
	void getdata()
	{
		cout << "Name:" << name << endl;
		cout << "ID" << id << endl;
		cout << "Bill" << bill << endl;
	}
	string getname()
	{
		return name;
	}


};
class queue {
	customer *queue1;
	int front, rear;
	int maxsize;
	int count, time;
public:
	queue(int mxsize)
	{
		queue1 = new customer[mxsize];
		maxsize = mxsize;
		front = rear = -1;
		count = time = 0;
	}
	void enque()
	{
		if (!isfull())
		{
			if (front == -1)
			{
				front = 0;
			}
			queue1[++rear].setdata();
```

```
                        time = time + 5;
                        count++;

                }
                else { cout << "Queue is full"; }
        }
        void dequeue()
        {
                if (!isempty())
                {
                        cout << "Deleted data of customer:\n" <<
queue1[front++].getname();
                        time = time - 5;
                }
                else { cout << "queue is empty"; }
        }
        bool isempty()
        {
                if (front == -1 && rear == -1)
                {
                        return true;
                }
                else { return false; }
        }
        bool isfull()
        {
                if (rear == maxsize)
                {
                        return true;
                }
                else { return false; }
        }
        void gettime()
        {
                cout << "time required to serve remaining customers is::" << time
<<"minutes"<< endl;
        }
        void gettotal()
        {
                cout << "Total customers served are" << count << endl;
        }
};
int main()
{
        queue custommer(10);
        int choice, ch;
        do {
                cout << "Press 1 to Add a customer " << endl;
                cout << "Press 2 delete a customer" << endl;
                cout << "Press 3 show total numbers of custmers served" << endl;
                cout << "Press 4 to show time required to serve remaing customers" <<
endl;
                cin >> choice;
                switch (choice)
                {
                case 1:
                        custommer.enque();
                        break;
                case 2: custommer.dequeue(); break;
                case 3:custommer.gettotal(); break;
                case 4: custommer.gettime(); break;
                }cout << "To continue press 1";
```

```
            cin >> ch;

      } while (ch == 1);
}
```



```
E:\Project4\Debug\Project4.exe
To continue press 11
Press 1 to Add a customer
Press 2 delete a customer
Press 3 show total numbers of custmers served
Press 4 to show time required to serve remaing customers
1
Enter name of the customerahmed
Enter bill1600
To continue press 11
Press 1 to Add a customer
Press 2 delete a customer
Press 3 show total numbers of custmers served
Press 4 to show time required to serve remaing customers
2
Deleted data of customer:
zohaibTo continue press 11
Press 1 to Add a customer
Press 2 delete a customer
Press 3 show total numbers of custmers served
Press 4 to show time required to serve remaing customers
3
Total customers served are3
To continue press 11
Press 1 to Add a customer
Press 2 delete a customer
Press 3 show total numbers of custmers served
Press 4 to show time required to serve remaing customers
4
time required to serve remaining customers is::10minutes
To continue press 1
```