Department of IT and Computer Science
Pak-Austria Fachhochschule: Institute of Applied Sciences
and Technology, Haripur, Pakistan

COMP-201L Data Structures and Algorithms Lab

Lab Report 07

Class: Computer Science

Name: Yaseen Ejaz Ahmed

Registration No.: B20F0283CS014

Semester: Third

Submitted to: Engr. Rafi Ullah

_____

Instructor Signature

Lab No. 7

Queues

Objectives:

To learn about Queues and its applications using Arrays and Linked Lists

Tools/Software Required:

C++ Compiler

Introduction:

A queue is an ordered collection of items where the addition of new items happens at one end, called the “rear,” and the removal of existing items occurs at the other end, commonly called the “front.” As an element enters the queue it starts at the rear and makes its way toward the front, waiting until that time when it is the next element to be removed.

Lab Tasks:

Lab Task 01: Enter a name character by character in a queue. Your program should Enqueue, Dequeue, and have flags to show whether the queue is empty or full. ASCII value of dequeue should also be displayed.

USING LINKED LIST:

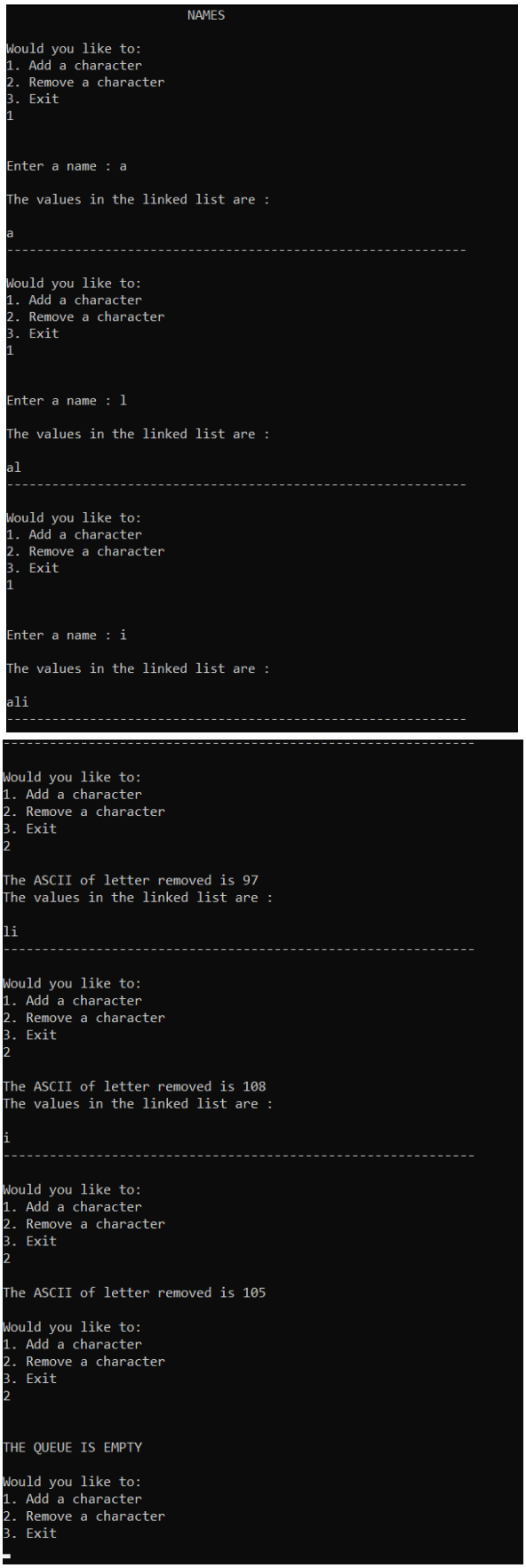
Code:

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

class node
{
	private:
		char data;
		node* next;
		node* head=NULL;
		node* ptr;
		node* tail;

	public:
		void Enqueue(char s)
		{
			node*temp=new node();
			temp->data=s;
			temp->next=NULL;

			if(head==NULL)
			{
				head=temp;
				ptr=head;
			}

			else
			{
				ptr->next=temp;
				ptr=ptr->next;
				tail=ptr;
			}
		}
```

```
int Dequeue()
{
        node*temp=head;
        int ASCII;
        ASCII=int(temp->data);
        head=head->next;
        delete temp;

        return ASCII;
}

bool isEmpty()
{
        if(head==NULL)
        return true;

        return false;
}

int show()
{
        if(head!=NULL)
        {
                cout<<"\nThe values in the linked list are :\n\n";
           ptr=head;
           while(ptr->next!=NULL)
           {
                cout<<ptr->data;
                ptr=ptr->next;
           }
           cout<<ptr->data;
           cout<<"\n------------------------------------------------------------";
        }

        else return 0;
}
};
```

```
int main()
{
	node n;
	int opt;
	char s;
	cout<<"\t\t\tNAMES";

	do
	{
		cout<<"\n\nWould you like to:\n1. Add a character\n2. Remove a
character\n3. Exit\n";
	    cin>>opt;

		if(opt==1)
		{
			cout<<"\n\nEnter a name : ";
	    cin>>s;
	    n.Enqueue(s);
	    n.show();
		}

		else if(opt==2)
		{
			if(!n.isEmpty())
			{
				cout<<"\nThe ASCII of letter removed is "<<n.Dequeue();
			}

			else
			{
				cout<<"\n\nTHE QUEUE IS EMPTY";
			}
			n.show();
		}

	}
	while(opt!=3);
}
```

Output:



```
                    NAMES

Would you like to:
1. Add a character
2. Remove a character
3. Exit
1


Enter a name : a

The values in the linked list are :

a
------------------------------------------------------------

Would you like to:
1. Add a character
2. Remove a character
3. Exit
1


Enter a name : l

The values in the linked list are :

al
------------------------------------------------------------

Would you like to:
1. Add a character
2. Remove a character
3. Exit
1


Enter a name : i

The values in the linked list are :

ali
------------------------------------------------------------
------------------------------------------------------------

Would you like to:
1. Add a character
2. Remove a character
3. Exit
2

The ASCII of letter removed is 97
The values in the linked list are :

li
------------------------------------------------------------

Would you like to:
1. Add a character
2. Remove a character
3. Exit
2

The ASCII of letter removed is 108
The values in the linked list are :

i
------------------------------------------------------------

Would you like to:
1. Add a character
2. Remove a character
3. Exit
2

The ASCII of letter removed is 105

Would you like to:
1. Add a character
2. Remove a character
3. Exit
2


THE QUEUE IS EMPTY

Would you like to:
1. Add a character
2. Remove a character
3. Exit
```

Lab Task 02: Enter a name character by character in a queue. Your program should Enqueue, Dequeue, and have flags to show whether the queue is empty or full. ASCII value of dequeue should also be displayed.

USING ARRAY:

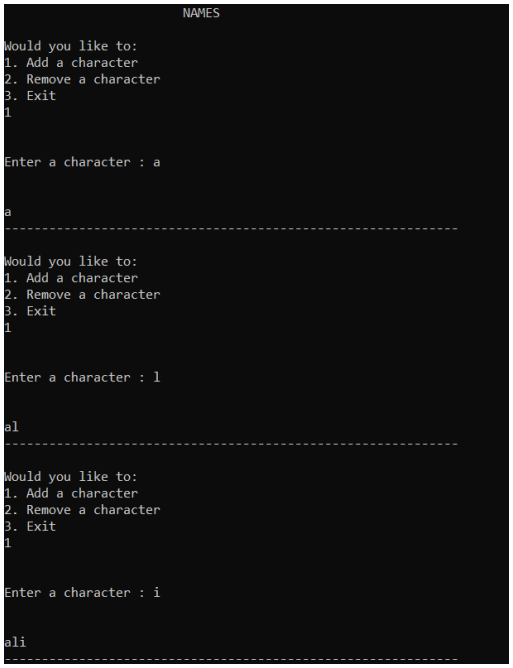
Code:

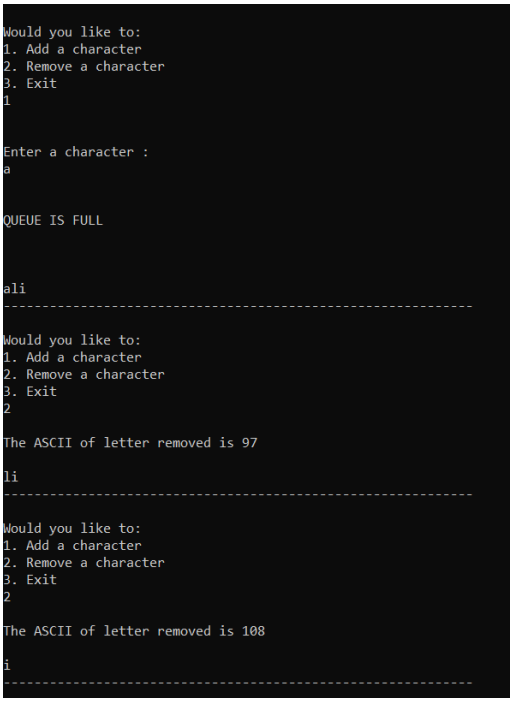
```
#include <iostream>
const int size=6;
using namespace std;


class node
{
        private:
                int first=0;
                int last=-1;
                int count=0;
                char a[size];

        public:

                void Enqueue(char s,int in)
                {
                        if(!isFull())
                        {
                                a[in]=s;
                           count++;
                        }
                        else cout<<"\n\nQUEUE IS FULL\n\n";

                }

                int Dequeue()
                {
                        int ascii;
                        if(!isEmpty())
                        {
                                ascii=int(a[0]);
                                for(int i=0;i<count;i++)
                           {
                                   a[i]=a[i+1];
                           }
```

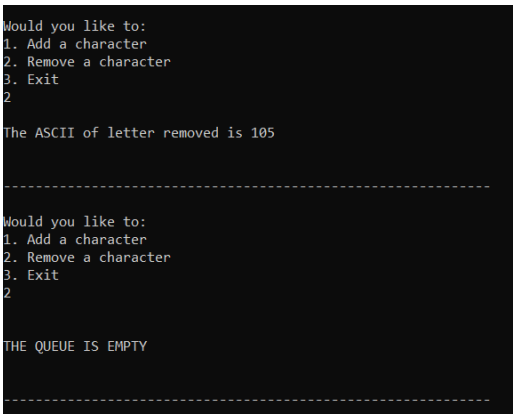
```
                count--;
                return ascii;
            }
            else
            cout<<"\n\nQUEUE IS EMPTY\n\n";

        }

        bool isEmpty()
        {
            if(a[0]==NULL)
    {
      return true;
    }
    return false;
        }

        bool isFull()
        {
            if(count==size)
             {
               return true;
              }
    return false;
        }


        int show()
        {
            cout<<endl<<endl;
            for(int i=0;i<=count;i++)
            {
                cout<<a[i];
            }
        }
};

int main()
{
    node n;
    int opt,in=0;
    char s;
```

```
	cout<<"\t\t\tNAMES";

	do
	{
		cout<<"\n\nWould you like to:\n1. Add a character\n2. Remove a
character\n3. Exit\n";
	   cin>>opt;

		if(opt==1)
		{
			cout<<"\n\nEnter a character : ";
		cin>>s;
		n.Enqueue(s,in);
		in++;
		n.show();
		}

		else if(opt==2)
		{
			if(!n.isEmpty())
			{
				cout<<"\nThe ASCII of letter removed is "<<n.Dequeue();
			}

			else
			{
				cout<<"\n\nTHE QUEUE IS EMPTY";
			}
			n.show();
		}

	}
	while(opt!=3);


}
```

Output:


NAMES
Would you like to:
1. Add a character
2. Remove a character
3. Exit
1
Enter a character : a
a
Would you like to:
1. Add a character
2. Remove a character
3. Exit
1
Enter a character : l
al
Would you like to:
1. Add a character
2. Remove a character
3. Exit
1
Enter a character : i
ali


Would you like to:
1. Add a character
2. Remove a character
3. Exit
1
Enter a character :
a
QUEUE IS FULL
ali
Would you like to:
1. Add a character
2. Remove a character
3. Exit
2
The ASCII of letter removed is 97
li
Would you like to:
1. Add a character
2. Remove a character
3. Exit
2
The ASCII of letter removed is 108
i


Would you like to:
1. Add a character
2. Remove a character
3. Exit
2
The ASCII of letter removed is 105
Would you like to:
1. Add a character
2. Remove a character
3. Exit
2
THE QUEUE IS EMPTY

Lab Task 03: Create a Queue and add values that are being dequeued.

USING LINKED LIST:

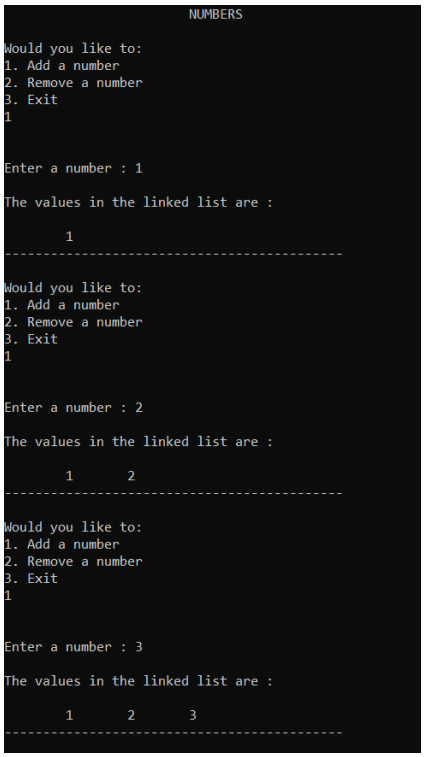
Code:

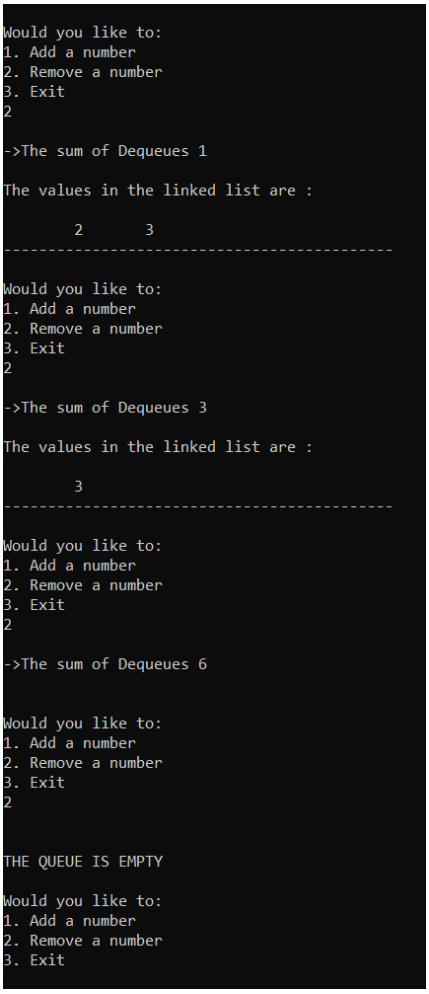
```
#include <iostream>
using namespace std;

class node
{
	private:
		int data;
		node* next;
		node* head=NULL;
		node* ptr;
		int s=0;

	public:

		void Enqueue(int s)
		{
			node*temp=new node();
			temp->data=s;
			temp->next=NULL;

			if(head==NULL)
			{
				head=temp;
				ptr=head;
			}

			else
			{
				ptr->next=temp;
				ptr=ptr->next;
			}
		}
```

```
        int Dequeue()
        {
                node*temp=head;
                s=s+temp->data;
                head=head->next;
                delete temp;

                return s;
        }

        bool isEmpty()
        {
                if(head==NULL)
                return true;

                return false;
        }

        int show()
        {
                if(head!=NULL)
                {
                        cout<<"\nThe values in the linked list are :\n\n";
                   ptr=head;
                   while(ptr->next!=NULL)
                   {
                        cout<<"\t"<<ptr->data;
                        ptr=ptr->next;
                   }
                   cout<<"\t"<<ptr->data;
                   cout<<"\n----------------------------------------";
                }

                else return 0;
        }
};

int main()
{
        node n;
        int opt;
        int s;
```

```
        cout<<"\t\t\tNUMBERS";

        do
        {
                cout<<"\n\nWould you like to:\n1. Add a number\n2. Remove a number\n3.
Exit\n";
            cin>>opt;

                if(opt==1)
                {
                        cout<<"\n\nEnter a number : ";
                cin>>s;
                n.Enqueue(s);
                n.show();
                }

                else if(opt==2)
                {
                        if(!n.isEmpty())
                        {
                                cout<<"\n->The sum of Dequeues "<<n.Dequeue()<<endl;
                        }

                        else
                        {
                                cout<<"\n\nTHE QUEUE IS EMPTY";
                        }
                        n.show();
                }

        }
        while(opt!=3);
}
```

Output:


NUMBERS
Would you like to:
1. Add a number
2. Remove a number
3. Exit
1
Enter a number : 1
The values in the linked list are :
1
Would you like to:
1. Add a number
2. Remove a number
3. Exit
1
Enter a number : 2
The values in the linked list are :
1 2
Would you like to:
1. Add a number
2. Remove a number
3. Exit
1
Enter a number : 3
The values in the linked list are :
1 2 3


Would you like to:
1. Add a number
2. Remove a number
3. Exit
2
->The sum of Dequeues 1
The values in the linked list are :
2 3
Would you like to:
1. Add a number
2. Remove a number
3. Exit
2
->The sum of Dequeues 3
The values in the linked list are :
3
Would you like to:
1. Add a number
2. Remove a number
3. Exit
2
->The sum of Dequeues 6
Would you like to:
1. Add a number
2. Remove a number
3. Exit
2
THE QUEUE IS EMPTY
Would you like to:
1. Add a number
2. Remove a number
3. Exit

Lab Task 04: Create a Queue and add values that are being dequeued.

USING ARRAY:

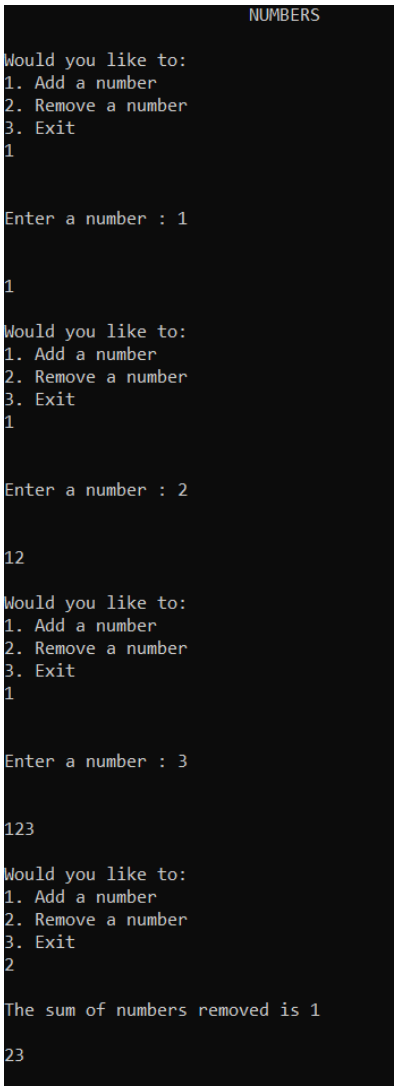
Code:

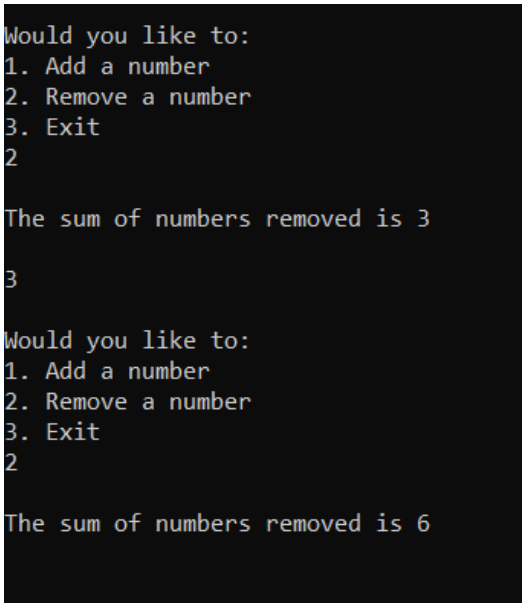
```
#include <iostream>
const int size=6;
using namespace std;

class node
{
        private:
                int count=0;
                int a[size];
                int s=0;

        public:

                void Enqueue(int s,int in)
                {
                        if(!isFull())
                        {
                                a[in]=s;
                           count++;
                        }
                        else cout<<"\n\nQUEUE IS FULL\n\n";

                }

                int Dequeue()
                {
                        if(!isEmpty())
                        {
                                s=s+a[0];
                                for(int i=0;i<count;i++)
                           {
                                   a[i]=a[i+1];
                           }
                           count--;
                           return s;
                        }
                        else
                        cout<<"\n\nQUEUE IS EMPTY\n\n";
```

```
        }

    bool isEmpty()
    {
        if(a[0]==NULL)
          {
           return true;
          }
    return false;
    }

    bool isFull()
    {
                if(count==size)
                {
                  return true;
                  }
    return false;
    }


        int show()
        {
                cout<<endl<<endl;
                for(int i=0;i<count;i++)
                {
                        cout<<a[i];
                }
        }
};

int main()
{
    node n;
    int opt,in=0;
    int s;
    cout<<"\t\t\tNUMBERS";
```

```
	do
	{
		cout<<"\n\nWould you like to:\n1. Add a number\n2. Remove a number\n3. Exit\n";
	   cin>>opt;

		if(opt==1)
		{
			cout<<"\n\nEnter a number : ";
		cin>>s;
		n.Enqueue(s,in);
		in++;
		n.show();
		}

		else if(opt==2)
		{
			if(!n.isEmpty())
			{
				cout<<"\nThe sum of numbers removed is "<<n.Dequeue();
			}

			else
			{
				cout<<"\n\nTHE QUEUE IS EMPTY";
			}
			n.show();
		}

	}
	while(opt!=3);


}
```

Output:


NUMBERS
Would you like to:
1. Add a number
2. Remove a number
3. Exit
1
Enter a number : 1
1
Would you like to:
1. Add a number
2. Remove a number
3. Exit
1
Enter a number : 2
12
Would you like to:
1. Add a number
2. Remove a number
3. Exit
1
Enter a number : 3
123
Would you like to:
1. Add a number
2. Remove a number
3. Exit
2
The sum of numbers removed is 1
23


Would you like to:
1. Add a number
2. Remove a number
3. Exit
2
The sum of numbers removed is 3
3
Would you like to:
1. Add a number
2. Remove a number
3. Exit
2
The sum of numbers removed is 6

Results & Observations:

In this lab, we have learnt about the basics and implementations of the queue data structure. We can use this data structure through linked lists as well as arrays. The queue is used in computers for data processing and managing. Any data that comes first, should be managed first then move to the next instruction. This is the logic of the queue data structure.