Assignment 1

Problem Statement:

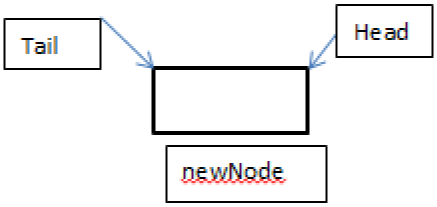
Write a program to implement the queue functions to
1. enqueue
2. dequeue.
Use lists to implement the program (not array)
Run the program as follows:
1. enqueue 1 items
2. dequeue 2 items (show the queue empty message)
3. enqueue 4 items
4. dequeue 2 items
submit the results showing the screen captures. Also submit the code.

Solution:

Enqueue operation:

1. Initializing the nodes Head = Tail = NULL
2. Creating first node

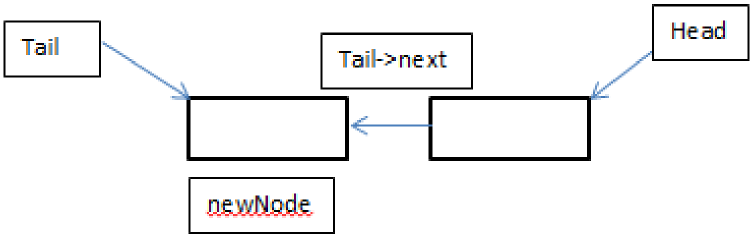
   Head = Tail = newNode



3. Adding subsequent nodes upon enqueue operation.

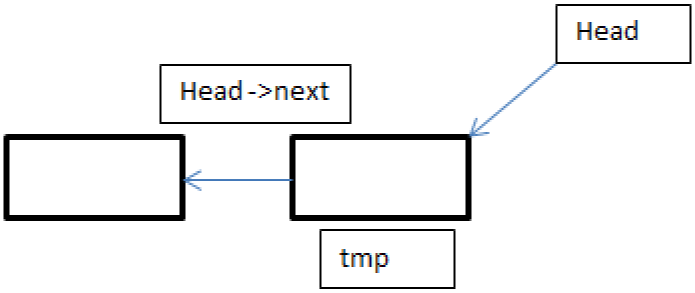
   Tail->next = newNode

   Tail = newNode



Dequeue operation:

```
1. If Head = NULL ; return error
2. If Head = Tail ; return (head->data)
   Head = Tail = NULL
3. If Head ≠ Tail ≠ NULL ;
   Tmp = head
   Head = head->next
   return (head->data)
   delete(tmp)
```



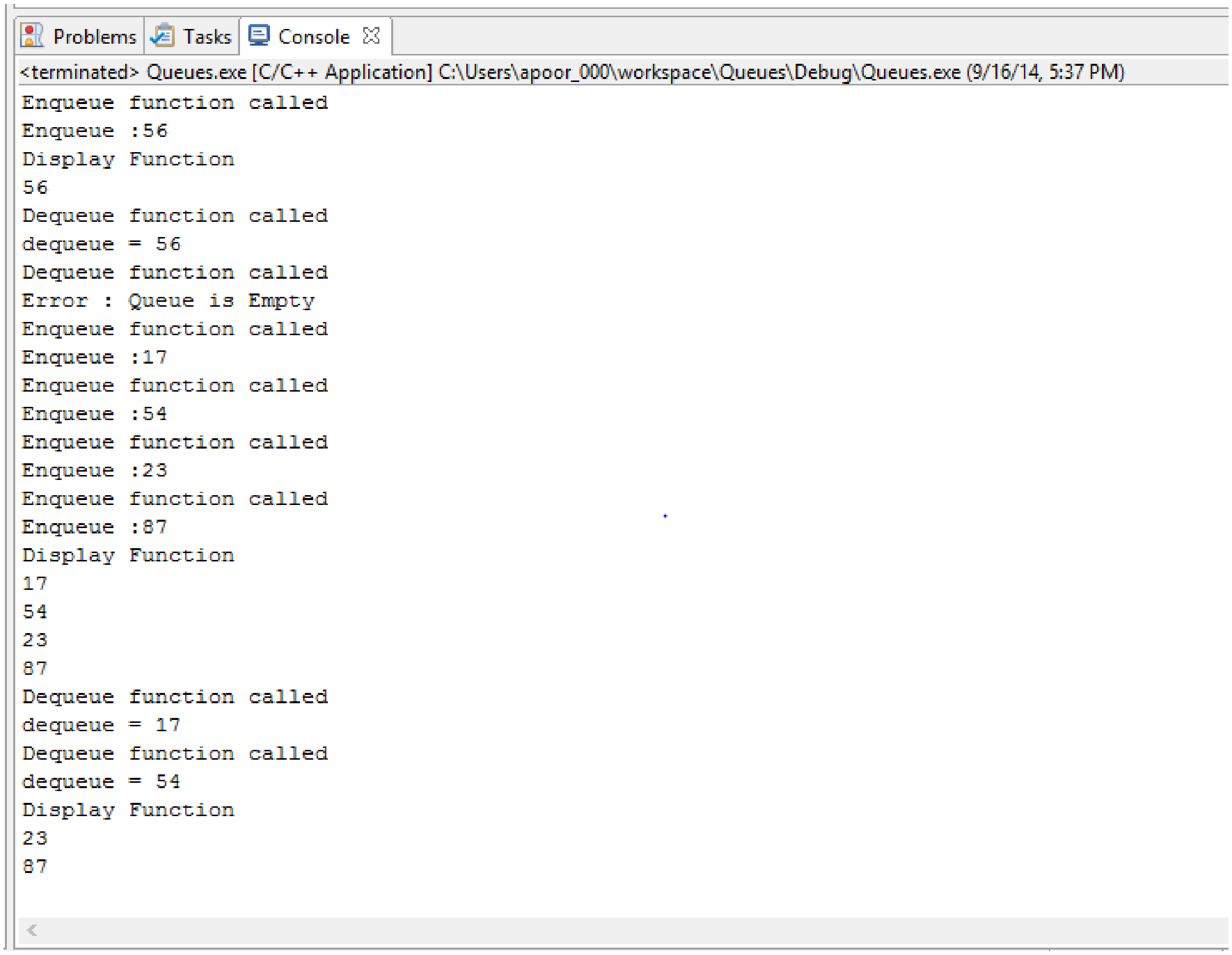
CODE : Written in CPP

```
/*
 *  queue.cpp
 *  Created on: Sep 16, 2014
 *  Author: Apoorva.D.A
 */
#include <cstdlib>
#include <iostream>
using namespace std;

struct node //one element of stack
{
   int data; //data item
   node* next; //pointer to next link
};
```

```
class queue //a list of links
{
        private:
                node* head; //pointer to first link
                node* tail; //pointer to last link
        public:

        queue() //no-argument constructor
        {
           head = NULL;
           tail = NULL;
        } //no first link
        void enqueue(int d); //add data item (one link)
        void display(); //display all link
        int dequeue();
};

void queue::enqueue(int d) //add data item
{
        cout << "Enqueue function called" << endl;
        cout << "Enqueue :" << d << endl;
        node *newNode = new node;
        newNode->data = d;
        newNode->next = NULL;

        if ( head == NULL && tail == NULL ) {
                head = tail = newNode;
                return;
        }
        tail->next = newNode;
        tail = newNode;
}

int queue::dequeue() {
        cout << "Dequeue function called" << endl;
        node *tmp = head;
        int d;
        if(head == NULL) //If the list is already empty
        {
                cout << "Error : Queue is Empty " << endl;
                return(-1);
        }
        if(head == tail) //If the list gets empty after multiple dequeues
        {
                d = head->data;
                head = tail = NULL;
        } else {
                d = head->data;
                head = head->next;
        }
        delete tmp;
        return(d);
}

void queue::display() //display all links
{
        cout << "Display Function" << endl;
        node* current = head; //set ptr to first link
        while (current != NULL) //quit on last link
        {
           cout << current->data << endl; //print data
           current = current->next; //move to next link
        }
}

int main(int argc, char** argv) {
   queue q1;
   int ret;
//Enqueue first element
```

```
    q1.enqueue(56);
    q1.display();

//Dequeue two times
        ret = q1.dequeue();
        if (ret >=0 ) {
            cout << "dequeue = " << ret << endl;
        }

        ret = q1.dequeue();
        if (ret >=0 ) {
            cout << "dequeue = " << ret << endl;
        }

//Enqueue four elements
        q1.enqueue(17);
        q1.enqueue(54);
        q1.enqueue(23);
        q1.enqueue(87);
        q1.display();

//Dequeue two times
    ret = q1.dequeue();
    if (ret >=0 ) {
        cout << "dequeue = " << ret << endl;
    }

    ret = q1.dequeue();
    if (ret >=0 ) {
        cout << "dequeue = " << ret << endl;
    }
    q1.display();
    return 0;
}
```

Tested and Executed on Eclipse IDE. The screenshot of the output is as below:



<terminated> Queues.exe [C/C++ Application] C:\Users\apoor_000\workspace\Queues\Debug\Queues.exe (9/16/14, 5:37 PM)

```
Enqueue function called
Enqueue :56
Display Function
56
Dequeue function called
dequeue = 56
Dequeue function called
Error : Queue is Empty
Enqueue function called
Enqueue :17
Enqueue function called
Enqueue :54
Enqueue function called
Enqueue :23
Enqueue function called
Enqueue :87
Display Function
17
54
23
87
Dequeue function called
dequeue = 17
Dequeue function called
dequeue = 54
Display Function
23
87
```