NAME :

GHULAM JILLANI

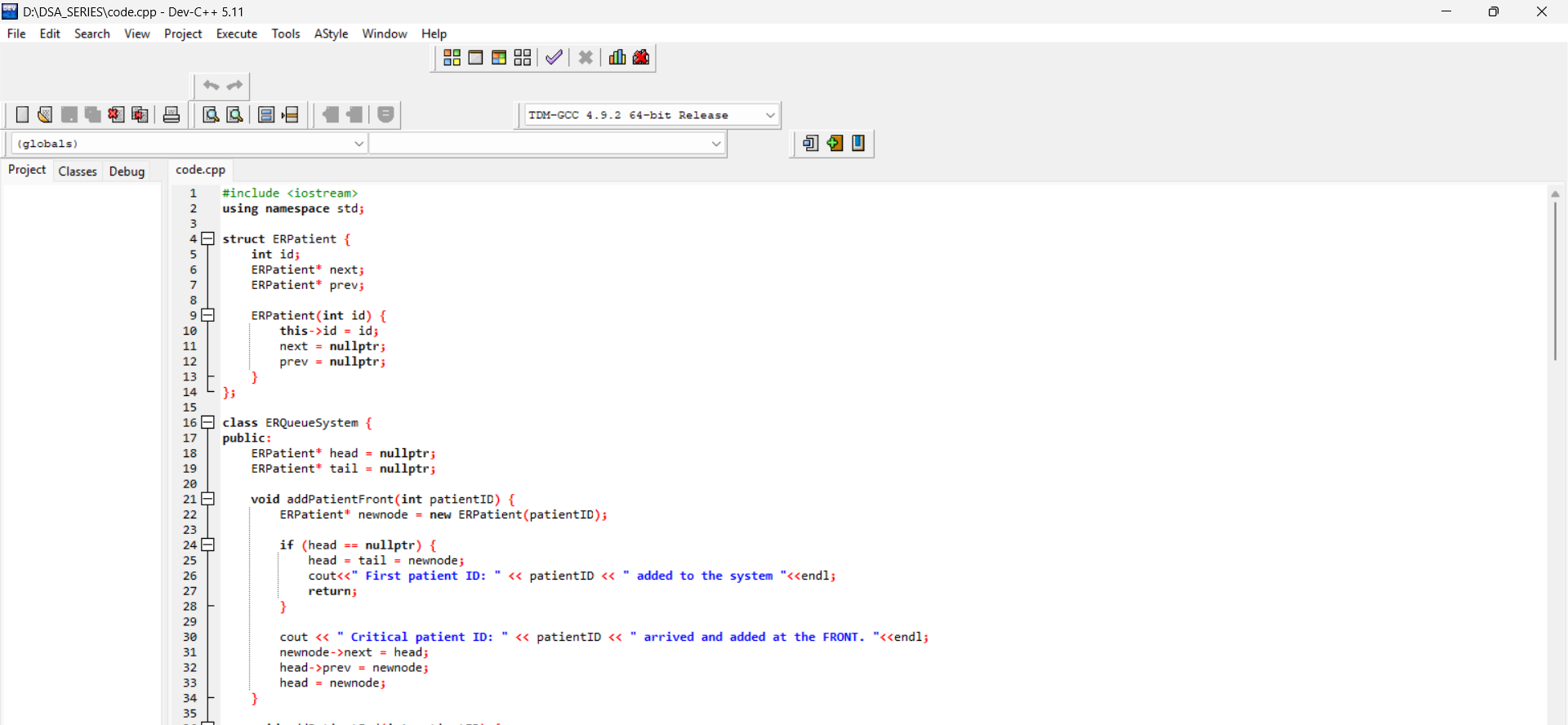
Program : BS Artificial Intelligence

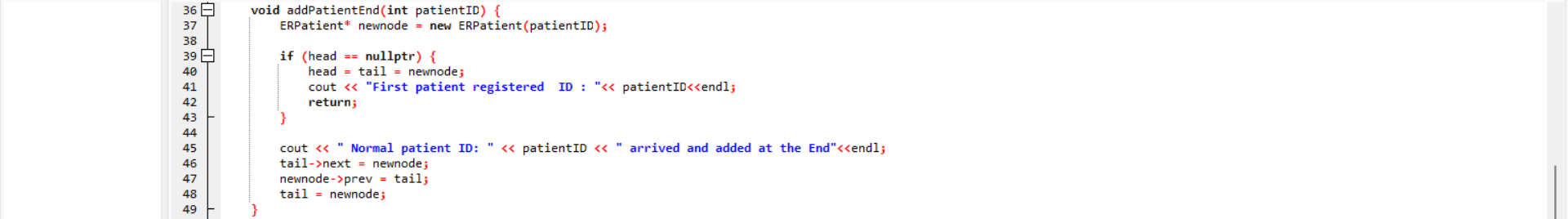
Section: B

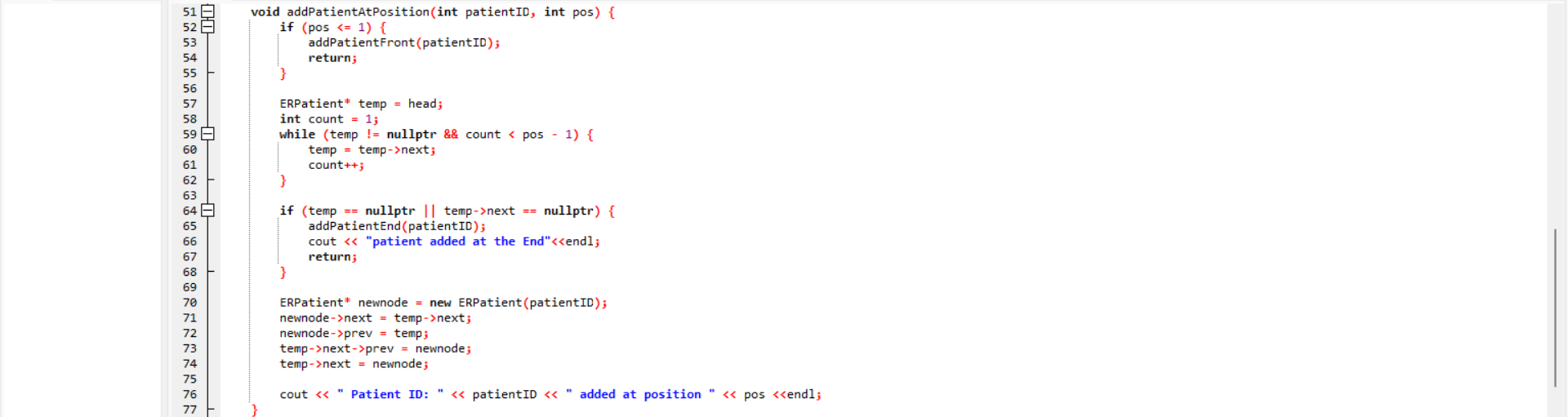
Mid Term Final Assignment

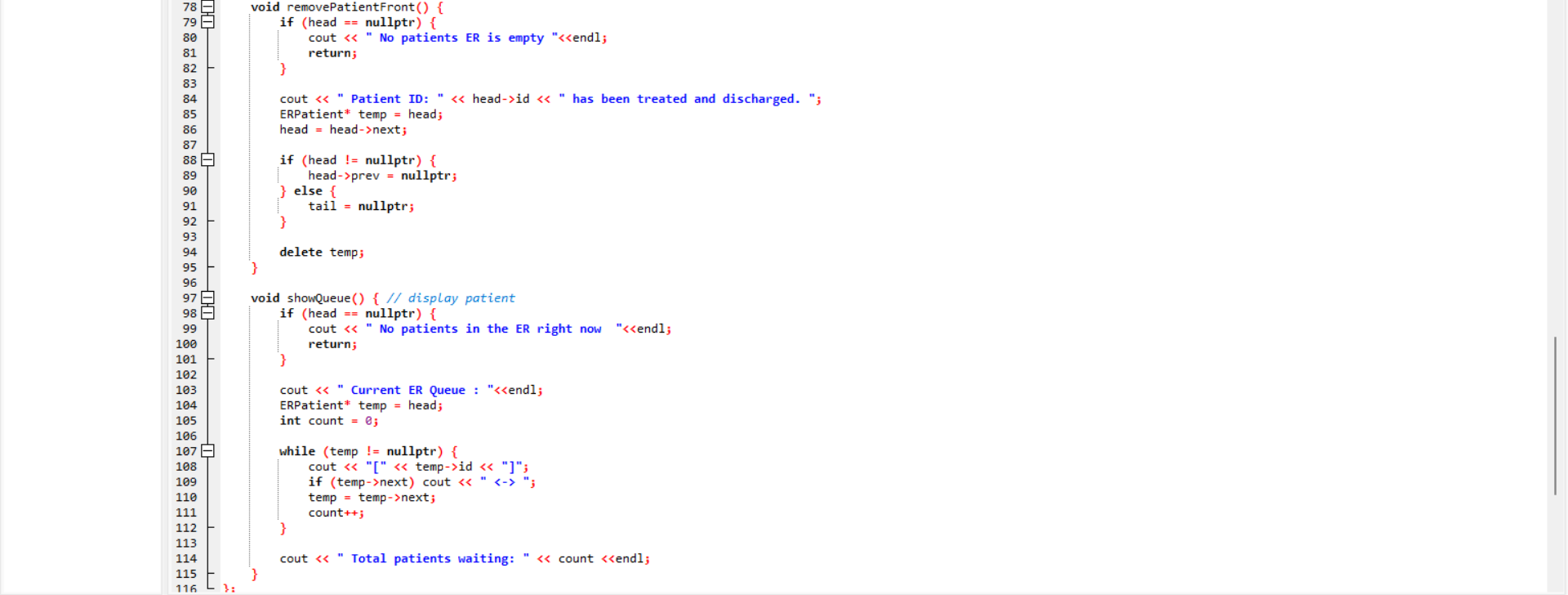
Submitted To: Syed Shayan Ali

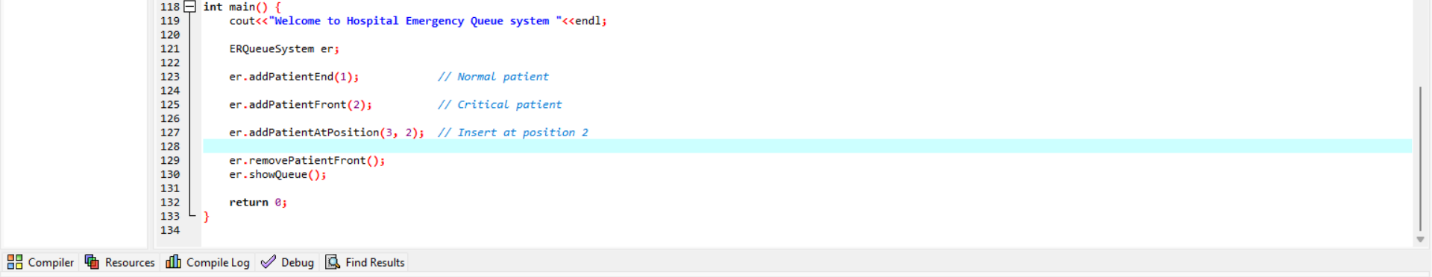
Code











# DRY RUN

**1 : insertAtBeginning(value)**

**Patient**

**HEAD -[101] <-> [102]-TAIL**

Critical patient — must be treated first.

Step 1: check if the list is empty if empty then create a new node insert at the head

ERPatient\* newnode = new ERPatient(patientID);

head = tail = newnode;

this is the first node so tail and head point to the same node

if list is not empty then

ERPatient\* newnode = new ERPatient(patientID);

step 2 : The new node’s **next** points to the head, the head’s **prev** points to the new node, and then we set the head to this new node.

newnode->next = head;

head->prev = newnode;

head = newnod

AFTER INSERTION :

**HEAD-[0]<-> [1] <-> [2]-TAIL**

**2 : insertAtEnd(value)**

Step 1: check if the list is empty if empty then create a new node insert at the head

ERPatient\* newnode = new ERPatient(patientID);

head = tail = newnode;

this is the first node so tail and head point to the same node

**Before :**

**HEAD-[0]<-> [1] <-> [2]-TAIL**

If the list is not empty, the tail’s **next** points to the new node, the new node’s **prev** points to the tail, and then we move the **tail** to the new node.

Step 1 : Create a newnode

ERPatient\* newnode = new ERPatient(patientID);

We have tail in the end so :

Tail->next points to newnode

Newnode->prev points to tail

And in last we move tail to newnode

Tail = newnode

After insertion :

**HEAD-[0]<-> [1] <-> [2]<->[3]-TAIL**

3 : InsertAtposition(pos,id)

Step 1: check if the list is empty if empty then create a new node insert at the head

ERPatient\* newnode = new ERPatient(patientID);

head = tail = newnode;

* we set a condition If the **position is greater than the list length**, we call insert at end.

If the position is in the middle, we move through the list to find the correct place. The new node’s **next** points to the current node’s next, and its **prev** points to the current node. Then we connect the current node’s **next** to the new node, and the next node’s **prev** back to the new node.

List Before :

**HEAD-[0]<-> [1] <-> [2]<->[3]-TAIL**

ERPatient\* newnode = new ERPatient(patientID);

newnode->next = temp->next;( The new node we want to add will point its **next** to temp’s **next**, which means the new node will come after temp in the list.)

newnode->prev = temp;

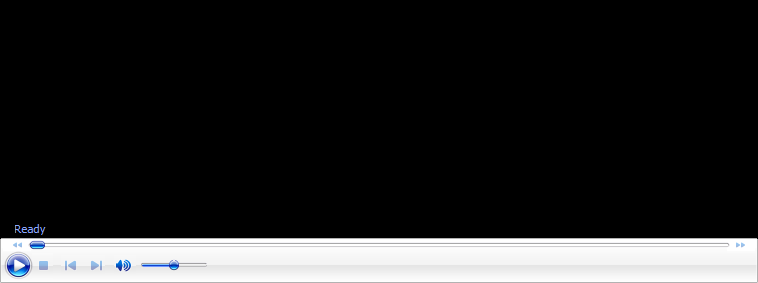
temp->next->prev = newnode;( he new node’s **prev** points to **temp** to connect it properly in the list.)

temp->next = newnode;

List After : (insert at 2 position)

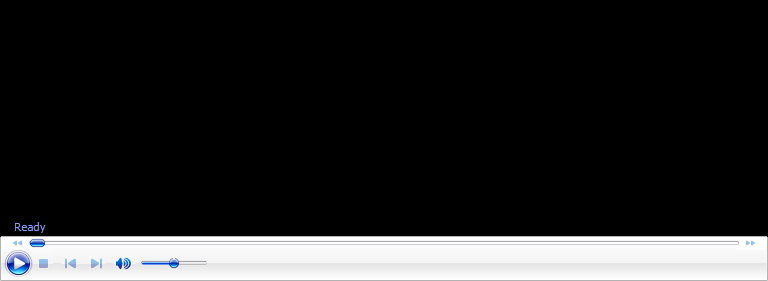
**HEAD-[0]<->[1]<->[1] <-> [2]<->[3]-TAIL**

INSERT NODE AT BEGINNING

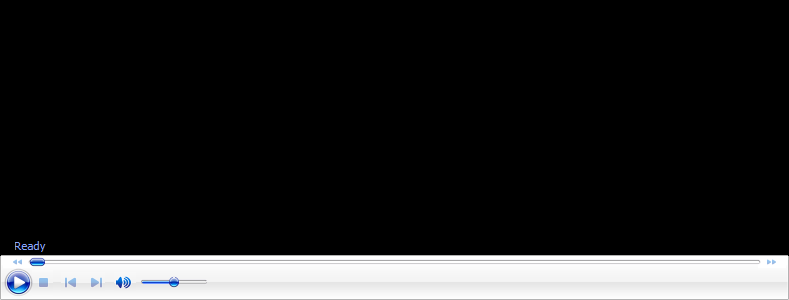


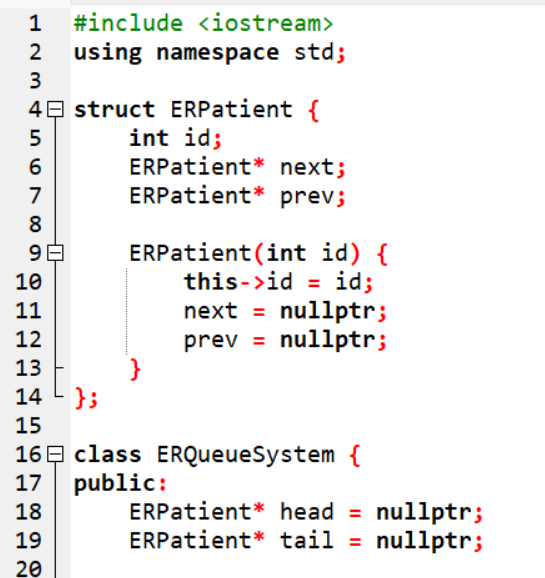
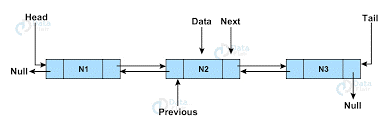
INSERT NODE AT END   


INSERT NODE AT GIVEN POSITION



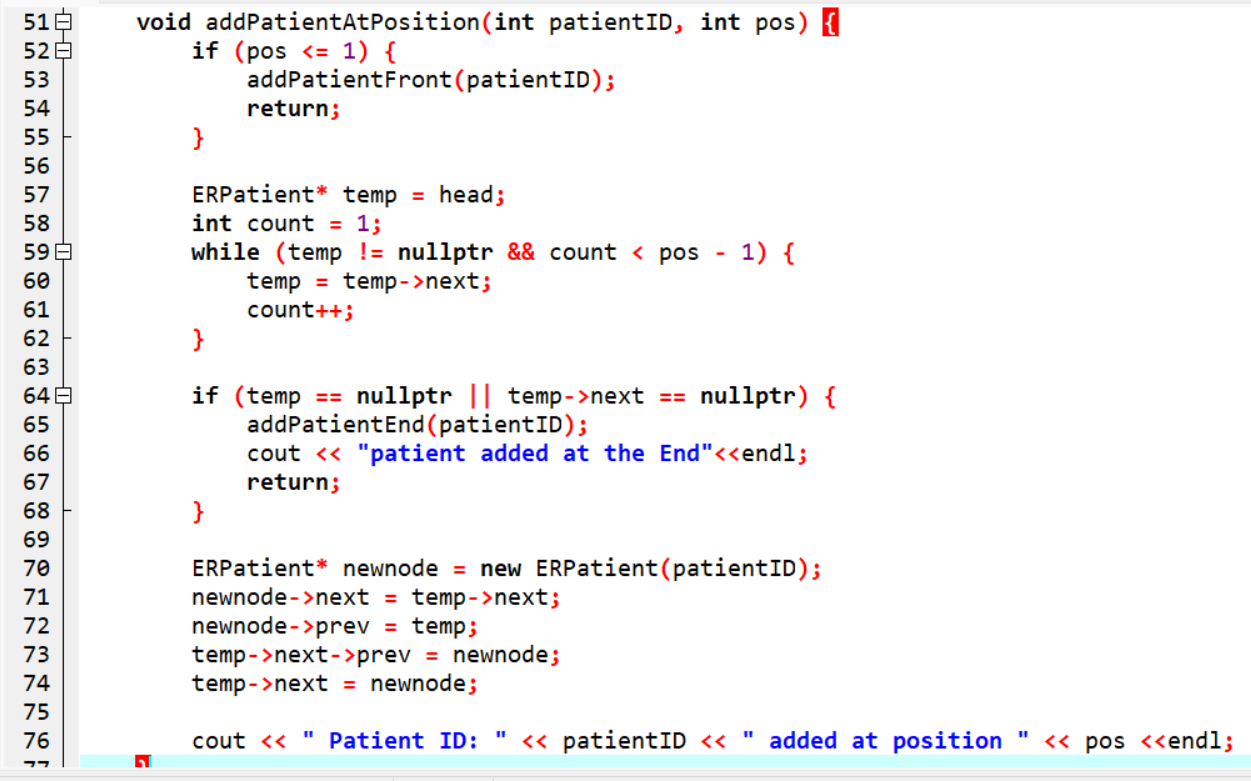
DELETE NODE FROM BEGINNING





NAME : Ghulam jillani

BS AI

 Mid Term Exam

