Lab Task 5 Singly Linked List[Patient ID System]

Task:

You are a software developer working for a hospital that manages patient check-ins.
Patients arrive, register, get treated, and leave. The hospital needs a dynamic system that can:

- Add new patients
- Remove treated patients
- Search a patient by ID
- Display all current patients

Since the number of patients changes continuously, the hospital wants a linked list–based system.

Your Task
Write a C++ program using a Singly Linked List where each patient has:

- Patient ID

Implement the following operations:

1. Insert a new patient at the end (new check-in)
2. Insert a patient at the beginning (emergency patient)
3. Display all patients

Code:

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

struct Node
{
int PatientID;

Node* next;
};

Node* head = NULL;

void InsertStart(int id)
{
     Node* newNode = new Node();
     newNode->PatientID = id;
     newNode->next = head;

     head = newNode;
     cout<<"Emergency Patient Added"<<endl;
}

void InsertEnd(int id)
{
     Node* newNode = new Node();
     newNode->PatientID = id;
     newNode->next = NULL;


     if(head == NULL)
     {
          head = newNode;
          cout<<"Patient added at the end"<<endl;
          return;
     }

     Node* temp = head;

     while(temp->next != NULL)
     {
          temp = temp->next;
     }

     temp->next = newNode;
     cout<<"Patient added at the end"<<endl;
}

void Display()
{
     if(head == NULL)
     {
          cout<<"No Patient In the List"<<endl;
          return;
     }

     Node* temp = head;

     while(temp != NULL)
     {
          cout<<"***********"<<endl;
          cout<<"ID: "<<temp->PatientID<<endl;
          cout<<" "<<endl;
          temp = temp->next;
     }
}

void Search(int id)
```

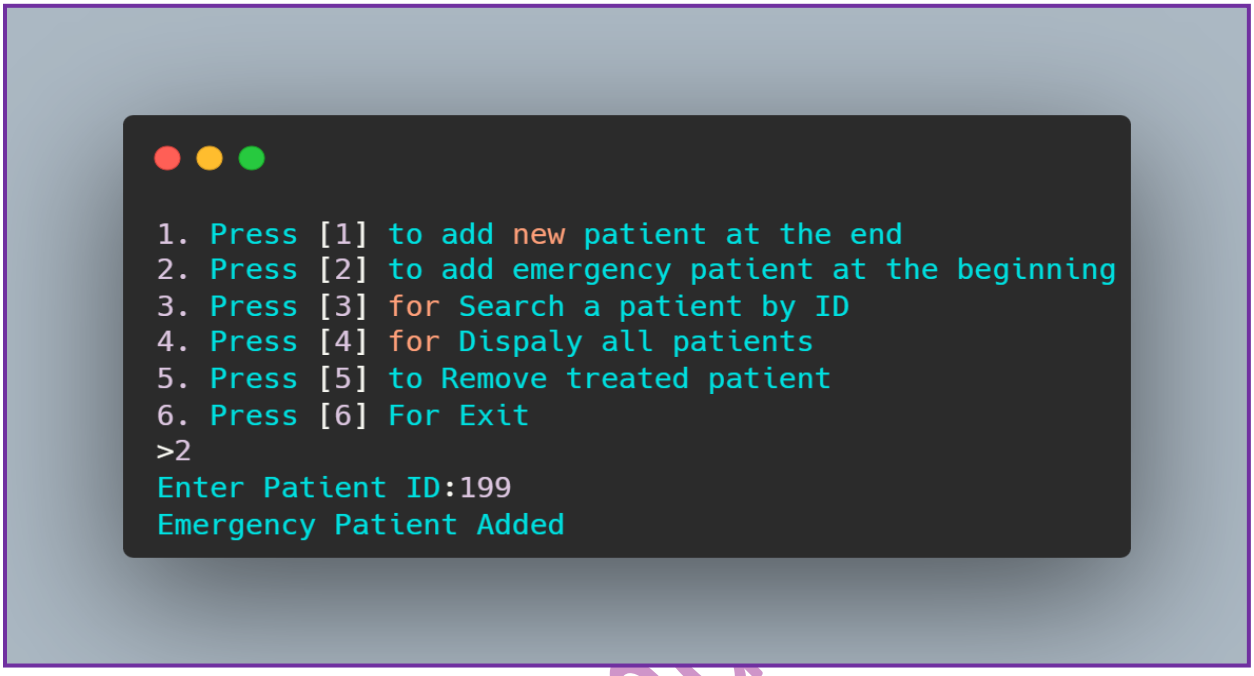
```
void Search(int id)
{
    Node* temp = head;

    while(temp != NULL)
    {
        if(temp->PatientID == id)
        {
            cout<<"Patient Found:"<<endl;
            cout<<"ID: "<<temp->PatientID<<endl;
            return;
        }
        temp = temp->next;
    }

    cout<<"Patient Not Found"<<endl;
}

void RemovePatient(int id)
{
    if(head == NULL)
    {
        cout<<"No patient for remove"<<endl;
        return;
    }


    if(head->PatientID == id)
    {
        Node* temp = head;
        head = head->next;
        delete temp;

        cout<<"Patient Removed"<<endl;
        return;
    }

    Node* curr = head;

    while(curr->next != NULL && curr->next->PatientID != id)
    {
        curr = curr->next;
    }

    if(curr->next == NULL)
    {
        cout<<"No Patient for remove"<<endl;
        return;
    }

    Node* temp = curr->next;
    curr->next = curr->next->next;
    delete temp;
    cout<<"Patient Removed"<<endl;
}
int main()
```

```cpp
int main()
{

int choice;

while(true)
{
 cout<<"1. Press [1] to add new patient at the end"<<endl;
 cout<<"2. Press [2] to add emergency patient at the beginning"<<endl;
 cout<<"3. Press [3] for Search a patient by ID"<<endl;
 cout<<"4. Press [4] for Dispaly all patients"<<endl;
 cout<<"5. Press [5] to Remove treated patient"<<endl;
 cout<<"6. Press [6] For Exit"<<endl;
 cout<<">";
 cin>>choice;

 if(choice == 1)
 {
    int ID;
    cout<<"Enter Patient ID:";
    cin>>ID;
    InsertEnd(ID);
 }
 else if(choice == 2)
 {
    int ID;
    cout<<"Enter Patient ID:";
    cin>>ID;

    InsertStart(ID);
 }
 else if(choice == 3)
 {
    int ID;
    cout<<"Enter Patient ID:";
    cin>>ID;
    Search(ID);
 }
 else if(choice == 4)
 {
    Display();
 }
 else if(choice == 5)
 {
    int ID;
    cout<<"Enter Patient ID:";
    cin>>ID;
    RemovePatient(ID);
 }
 else if(choice == 6)
 {
    return 0;
 }
 else
 {
    cout<<"Invalid Choice"<<endl;
 }


 }

return 0;
}
```

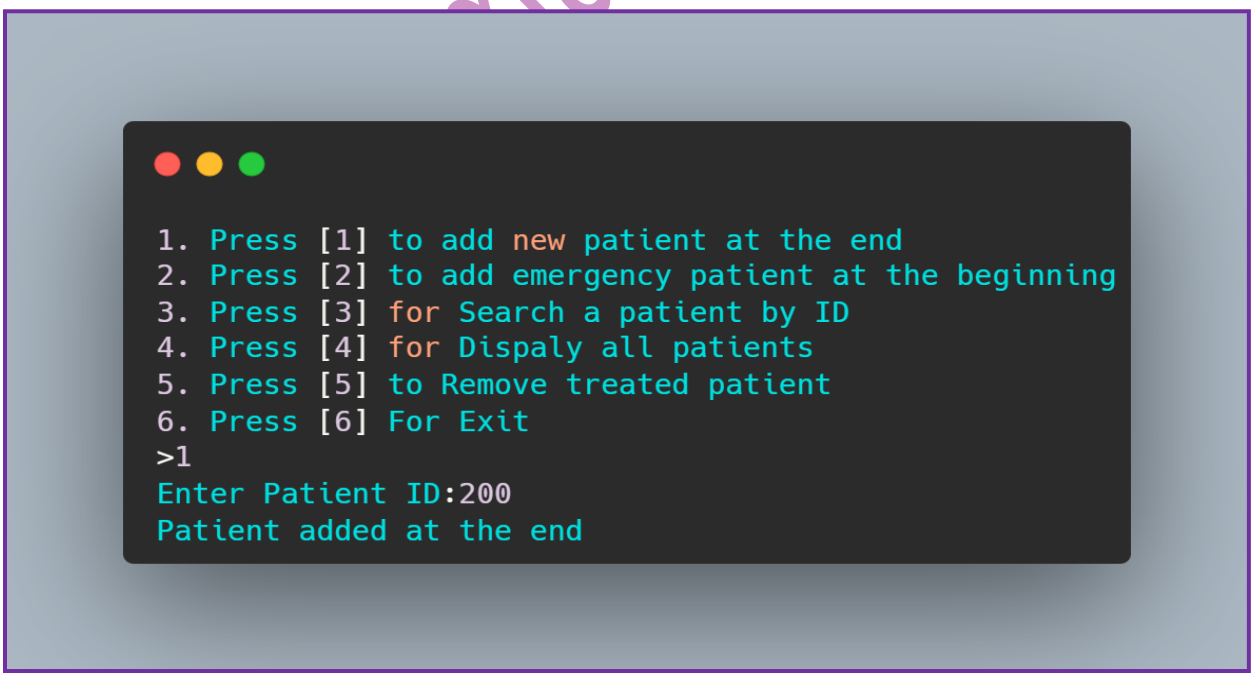
Table Of Output:

	Condition	Output
1	**Add emergency patient (choice 2)**	Emergency Patient Added
2	**Add patient at end (choice 1, list empty or not)**	Patient added at the end
3	**Display when list is empty**	No Patient In the List
4	**Display when list has patients**	Patient IDs printed
5	**Search when patient is found**	Patient Found + ID
6	**Search when patient not found**	Patient Not Found
7	**Remove when list is empty**	No patient for remove
8	**Remove when patient ID is found**	Patient Removed
9	**Remove when patient ID not found**	No Patient for remove
10	**Invalid menu option**	Invalid Choice
Total 10 Possible Outputs		

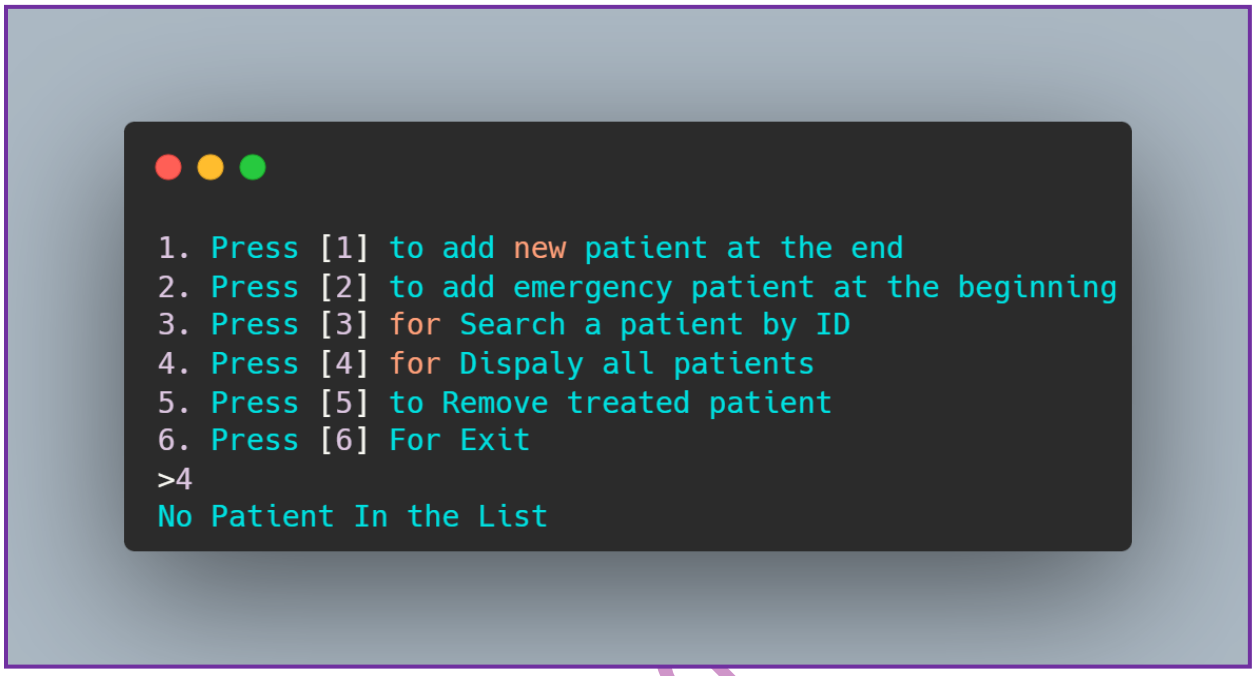
When Add emergency patient (choice 2):



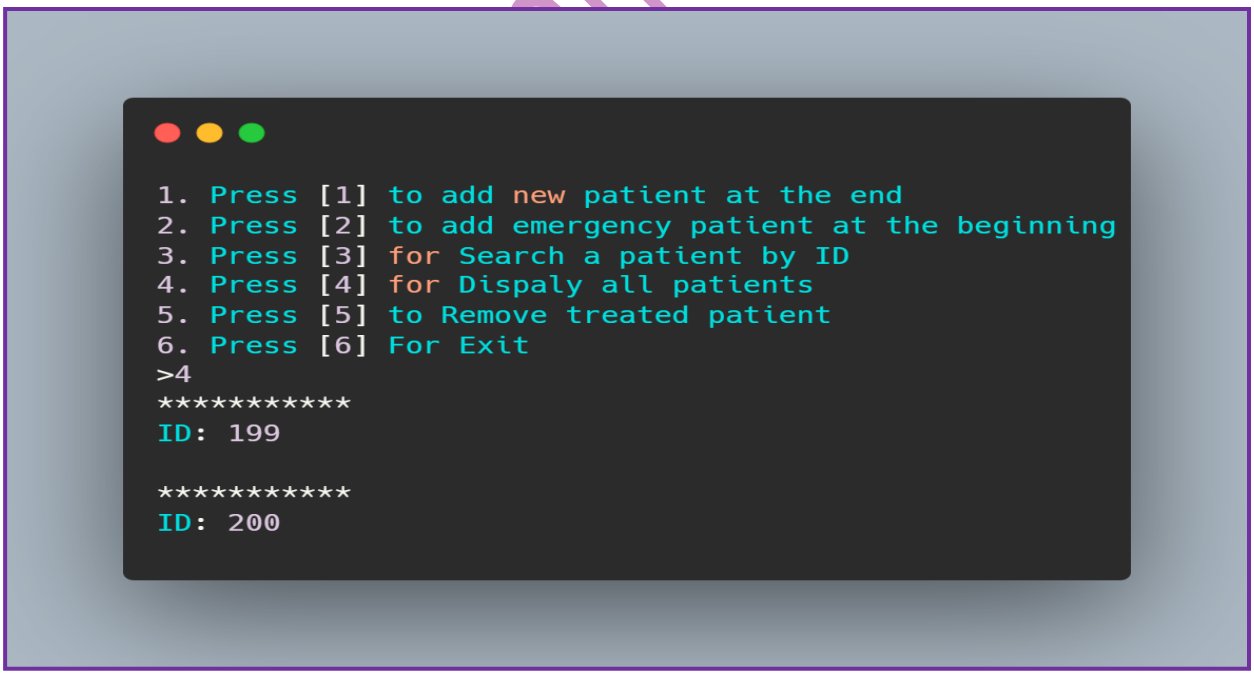
When Add patient at end (choice 1, list empty or not):



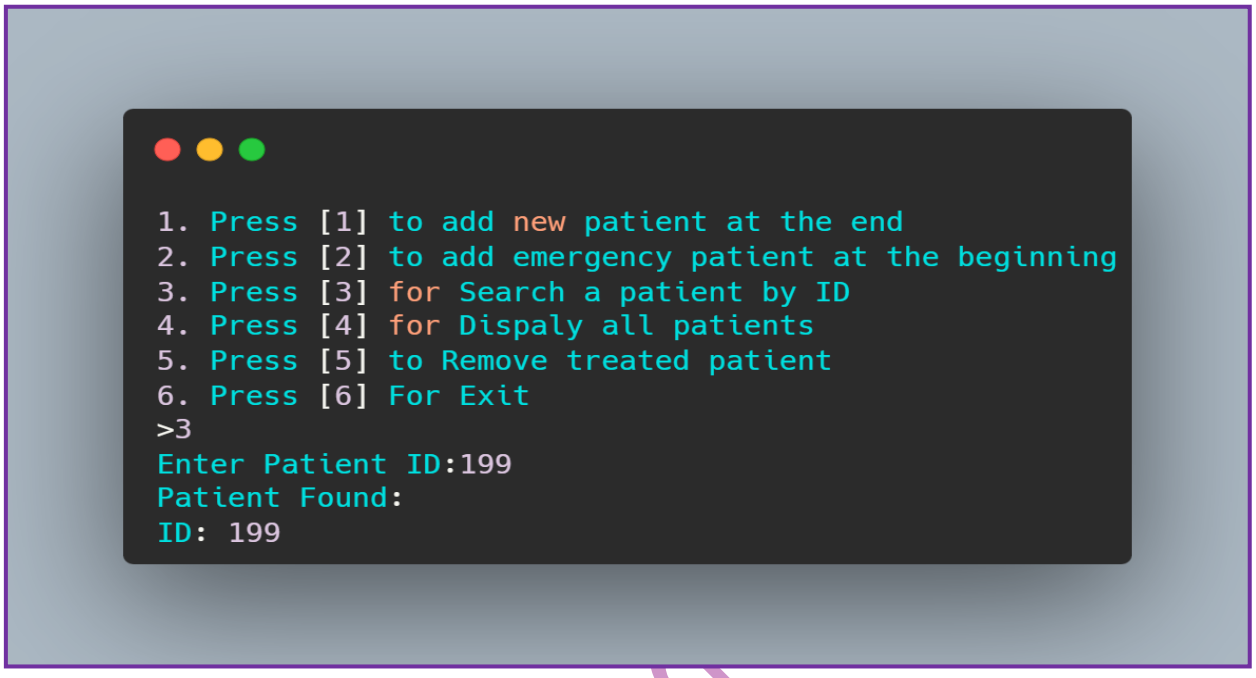
Display When list is empty:



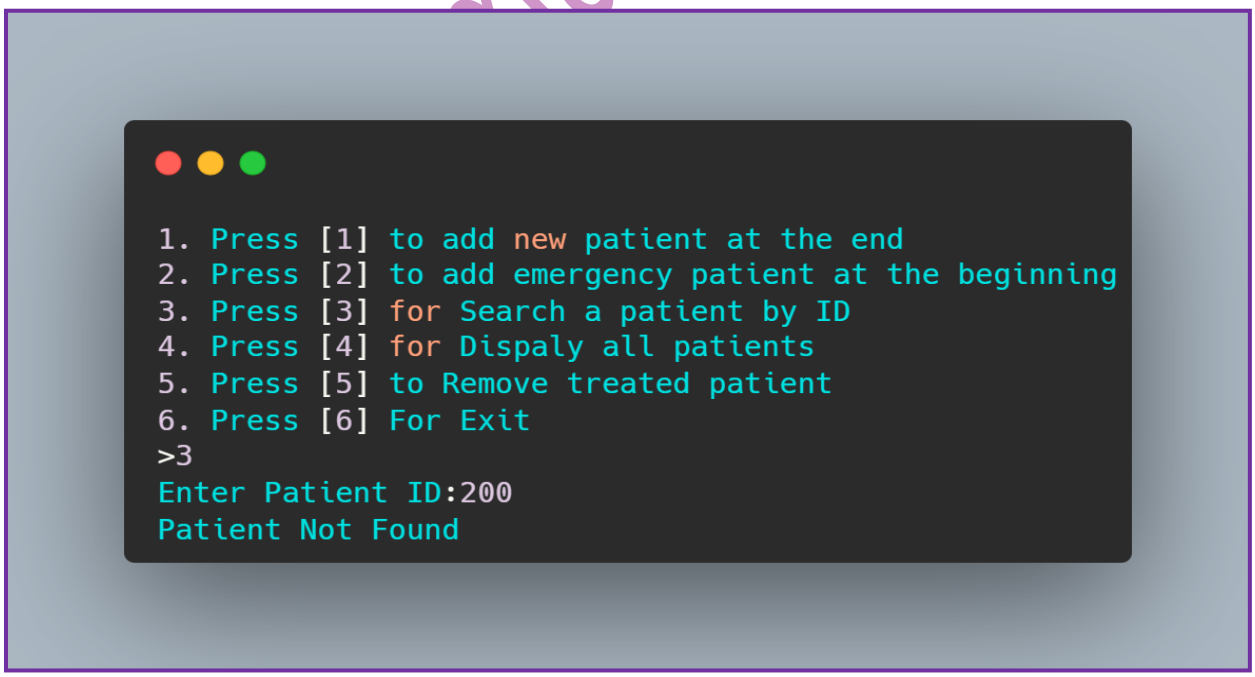
Display When list has patients:



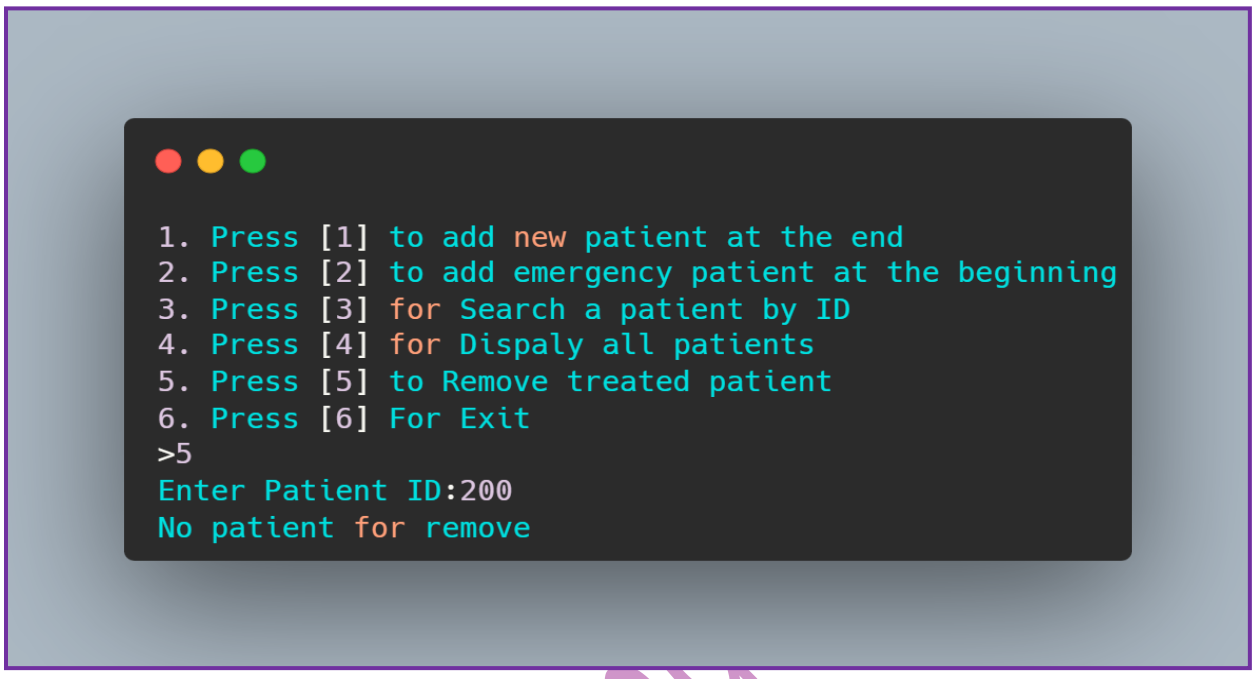
Search When patient is found



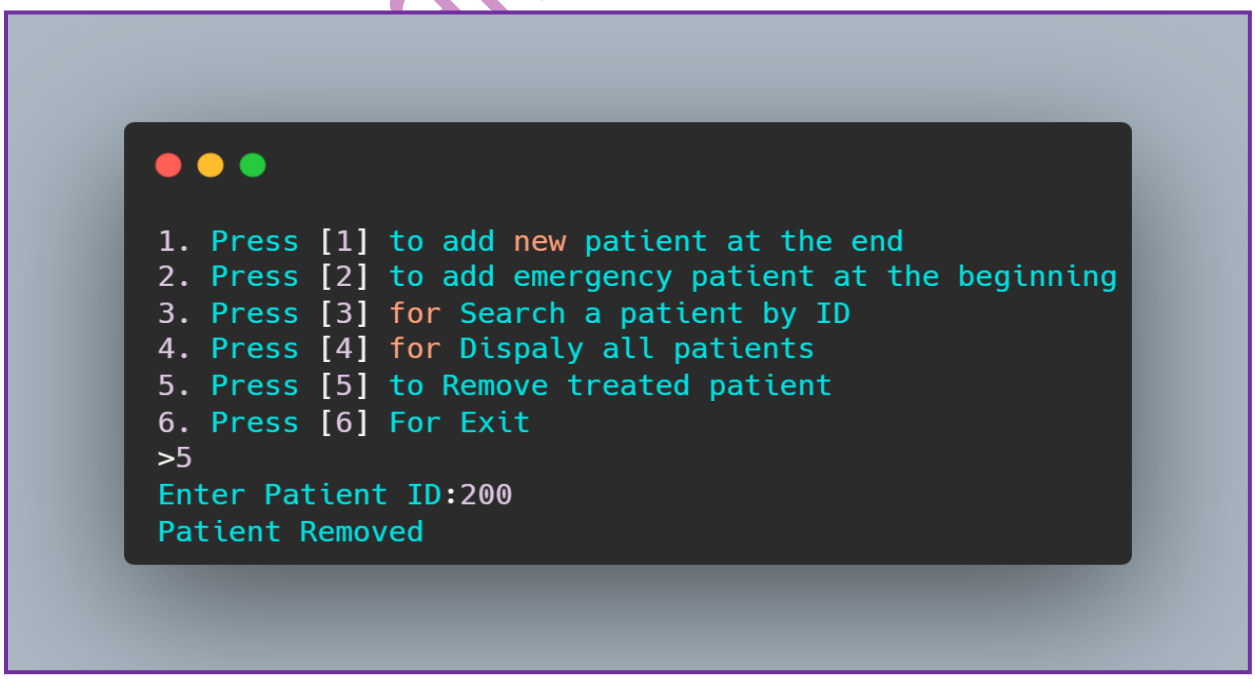
Search When patient not found:



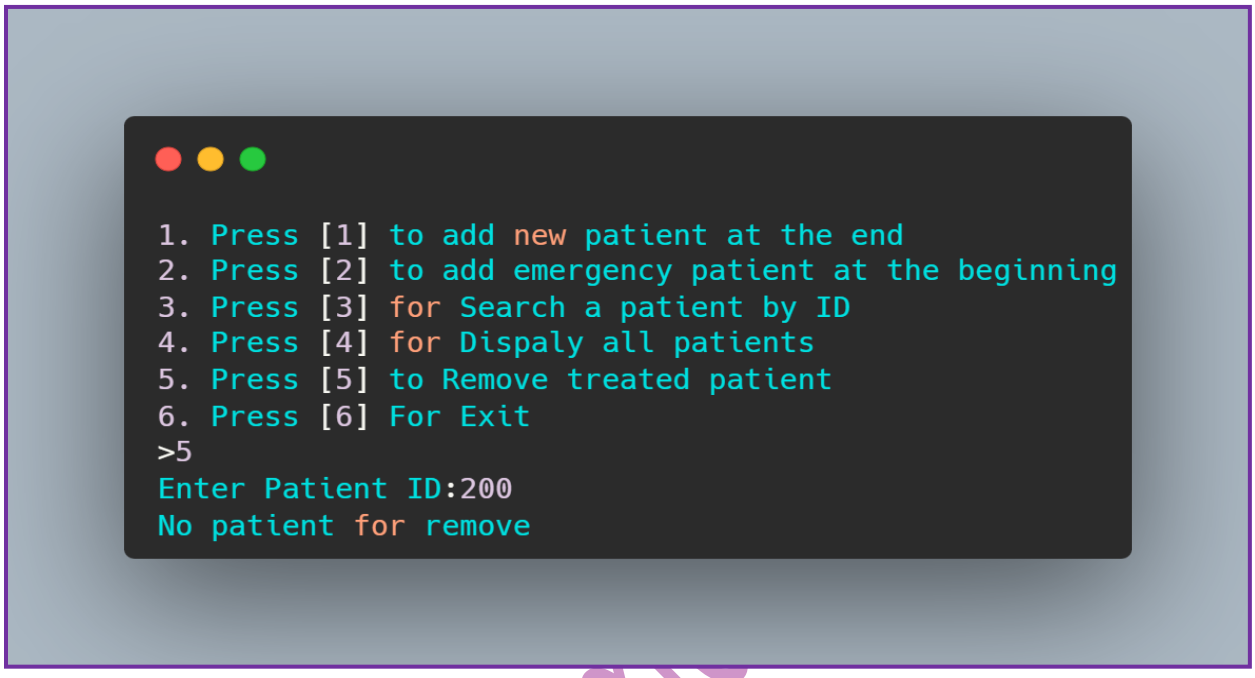
Remove When list is empty



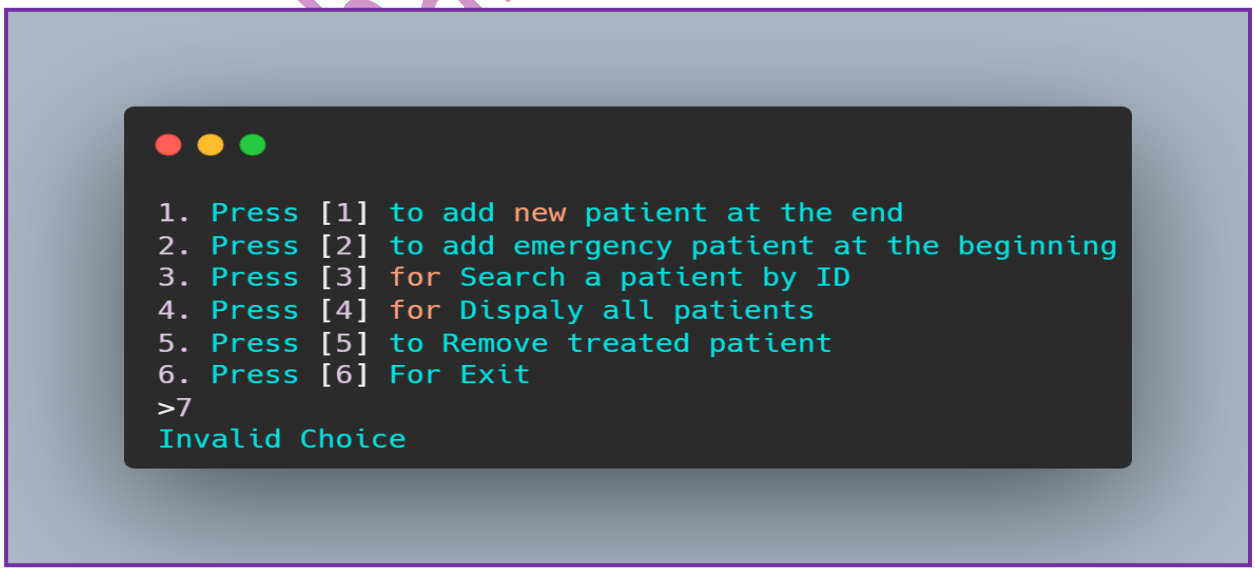
Remove When patient ID is found



Remove When patient ID not found



When Invalid menu option



Click here to Get this code on GitHub

Click here to Test this Code by Yourself.