singlyLL.c
~/DSC_Lab/LabCycle1

```c
#include <stdio.h>
#include <stdlib.h>

struct node
{
    int data;
    struct node *next;
};
typedef struct node Node;

Node* newNode(int data)
{
    Node *n=(Node*)malloc(sizeof(Node));
    n->data=data;
    n->next=NULL;
    return n;
}

struct list
{
    Node *head;
    Node *tail;
    int min;
    int max;
};
typedef struct list List;

List* newList()
{
    List *l=(List*)malloc(sizeof(List));
    l->head=NULL;
    l->tail=NULL;
    return l;
}

void insertTail(List *l,int data)
{
    Node *new_node=newNode(data);
    if(l->head==NULL)
        l->head=l->tail=new_node;
    else
    {
        l->tail->next=new_node;
        l->tail=new_node;
    }
}

void insertAtPos(List *l,int data,int pos)
```

C Tab Width: 4 Ln 34, Col 2 INS

singlyLL.c
~/DSC_Lab/LabCycle1

```c
void insertAtPos(List *l,int data,int pos)
{
    Node *n=l->head,*prev=NULL;
    int i=0;
    if(pos<=0)
    {
        printf("Position does not exist\n");
        return;
    }
    if(pos==1)
    {
        Node *new_node=newNode(data);//put this here,coz if pos doesnot exist then this node will not be created
        new_node->next=l->head;
        l->head=new_node;
        if(l->tail==NULL)
            l->tail=l->head;
        return;
    }
    while(n!=NULL && i<pos-1)
    {
        prev=n;
        n=n->next;
        i++;
    }
    if(n==NULL)
    {
        printf("Position does not exist\n");
        return;
    }
    Node *new_node=newNode(data);
    prev->next=new_node;
    new_node->next=n;//tail will remain as it is, don't need to alter
}

void DeleteHead(List *l)
{
    Node *temp=l->head;
    if(l->head==NULL)
        return;
    else if(l->head==l->tail)
        l->head=l->tail=NULL;
    else
        l->head=l->head->next;
    free(temp);
}
```

C Tab Width: 4 Ln 93, Col 1 INS

singlyLL.c
~/DSC_Lab/LabCycle1

```c
void DeleteValue(List *l,int data)
{
    Node *n=l->head,*prev=NULL;
    if(l->head->data==data)
    {
        DeleteHead(l);
        return;
    }
    while(n!=NULL && n->data!=data)
    {
        prev=n;
        n=n->next;
    }
    if(n==NULL)
    {
        printf("Element with value %d does not exist in the list\n",data);
        return;
    }
    Node *temp=n;
    prev->next=n->next;
    if(l->tail==temp)
        l->tail=prev;
    free(temp);
}

void findMin(List *l)
{
    Node *n=l->head;
    if(l->head==NULL)
    {
        printf("There are no elements in the list\n");
        return;
    }
    int min=n->data;
    n=n->next;
    while(n!=NULL)
    {
        if(n->data<min)
            min=n->data;
        n=n->next;
    }
    l->min=min;
}

void findMax(List *l)
{
    Node *n=l->head;
```

C Tab Width: 4 Ln 140, Col 1 INS

Open | singlyLL.c ~/DSC_Lab/LabCycle1 | Save

```c
void findMax(List *l)
{
    Node *n=l->head;
    if(l->head==NULL)
    {
        printf("There are no elements in the list\n");
        return;
    }
    int max=n->data;
    n=n->next;
    while(n!=NULL)
    {
        if(n->data>max)
            max=n->data;
        n=n->next;
    }
    l->max=max;
}

void PrintList(List *l)
{
    printf("The List is : ");
    Node *n=l->head;
    if(n==NULL)
        printf("The list is empty");
    while(n!=NULL)
    {
        printf("%d ",n->data);
        n=n->next;
    }
    printf("\n");
}

void freeList(List *l)
{
    Node *n=l->head,*temp;
    while(n!=NULL)
    {
        temp=n;
        n=n->next;
        free(temp);
    }
}

int main()
{
    List *l=newList();
```

C | Tab Width: 4 | Ln 184, Col 1 | INS

Open | singlyLL.c ~/DSC_Lab/LabCycle1 | Save

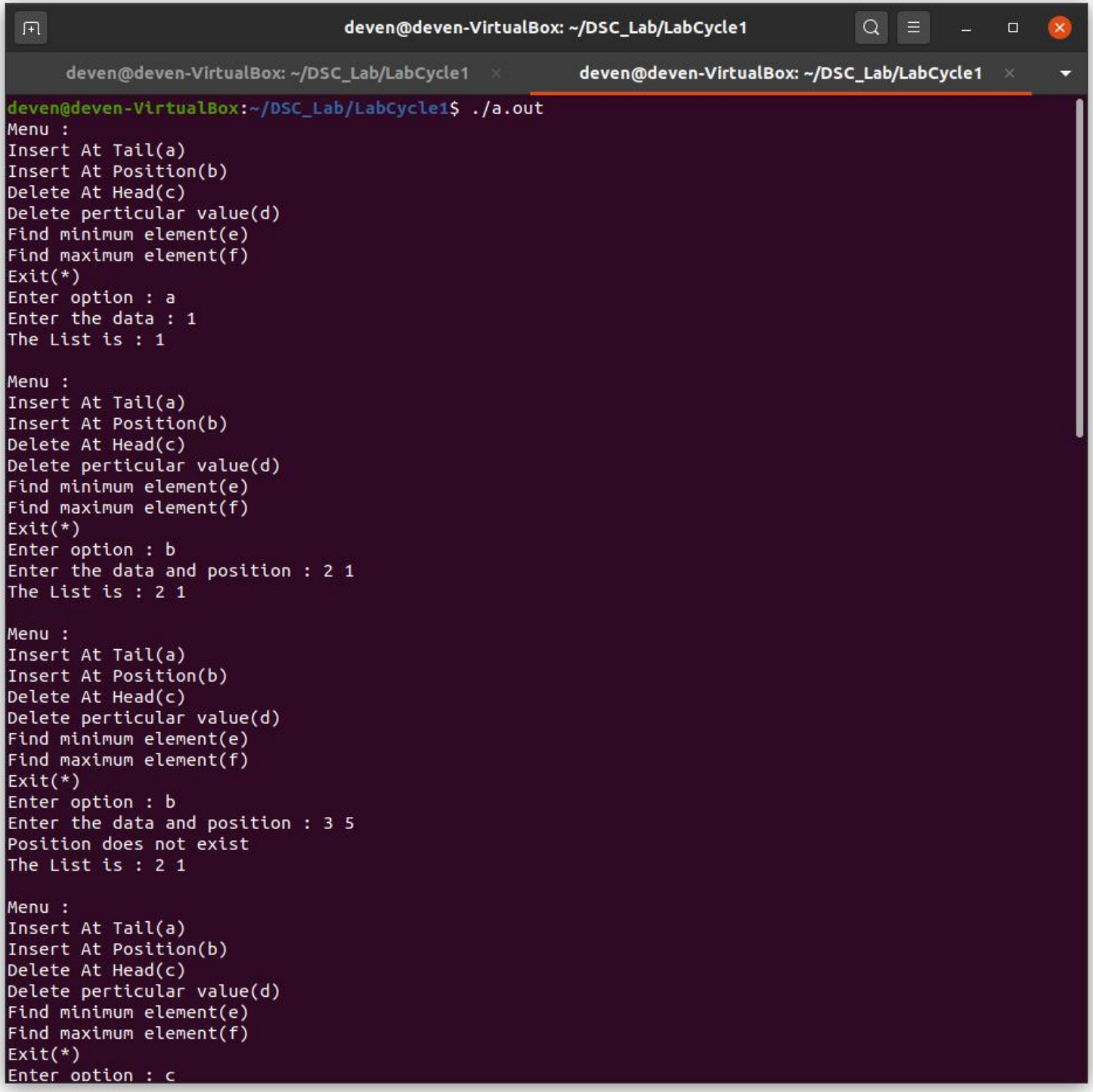
```c
int main()
{
    List *l=newList();
    char ch;
    printf("Menu :\n");
    printf("Insert At Tail(a)\n");
    printf("Insert At Position(b)\n");
    printf("Delete At Head(c)\n");
    printf("Delete perticular value(d)\n");
    printf("Find minimum element(e)\n");
    printf("Find maximum element(f)\n");
    printf("Exit(*)\n");
    printf("Enter option : ");
    scanf("%c",&ch);
    while(ch!='*')
    {
        if(ch=='a')
        {
            int data;
            printf("Enter the data : ");
            scanf("%d",&data);
            insertTail(l,data);
        }
        else if(ch=='b')
        {
            int data,pos;
            printf("Enter the data and position : ");
            scanf("%d%d",&data,&pos);
            insertAtPos(l,data,pos);
        }
        else if(ch=='c')
            DeleteHead(l);
        else if(ch=='d')
        {
            int data;
            printf("Enter the data to be deleted : ");
            scanf("%d",&data);
            DeleteValue(l,data);
        }
        else if(ch=='e')
        {
            findMin(l);
            printf("The minimum element is %d\n",l->min);
        }
        else if(ch=='f')
        {
            findMax(l);
```

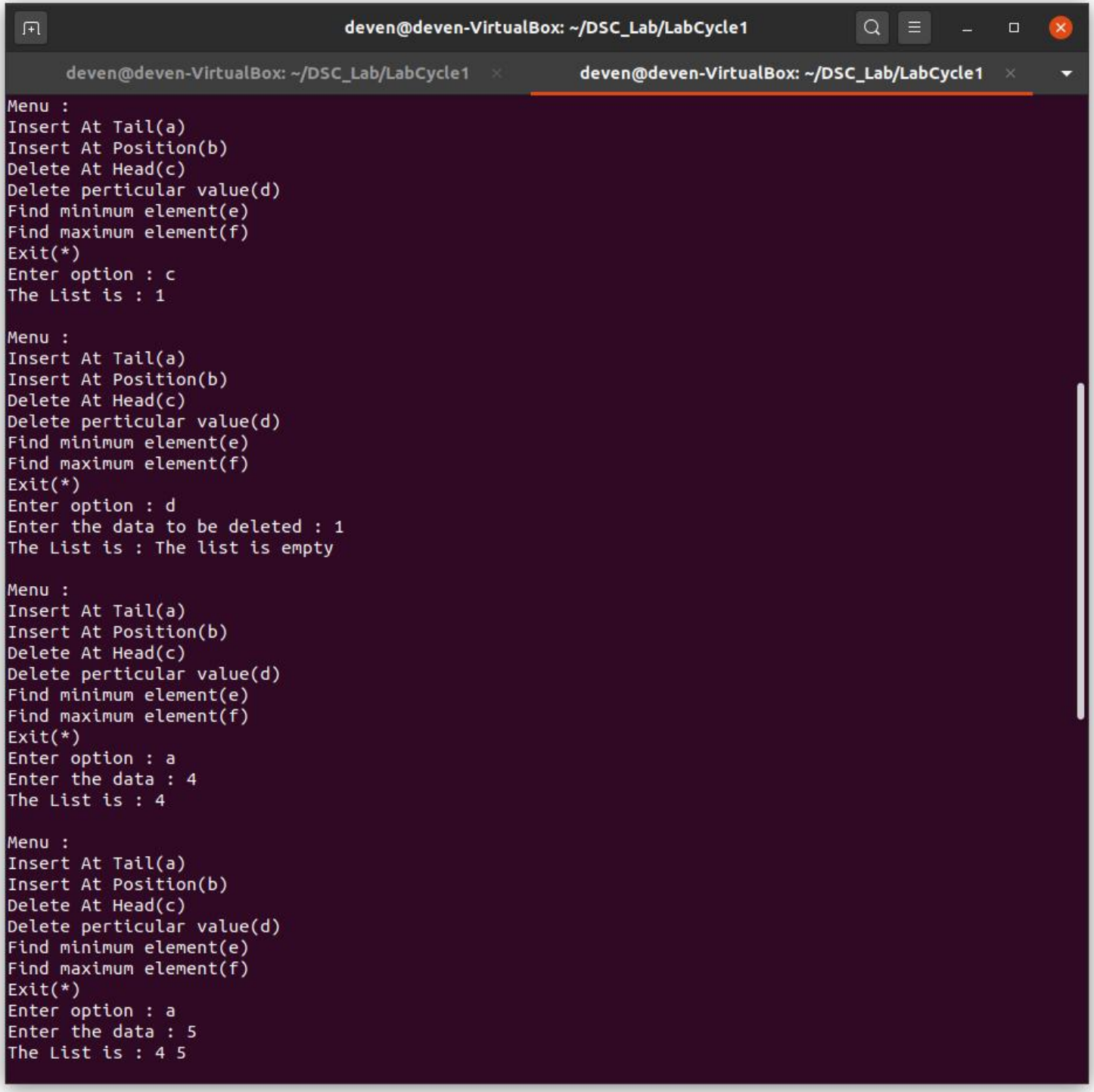
C | Tab Width: 4 | Ln 228, Col 1 | INS

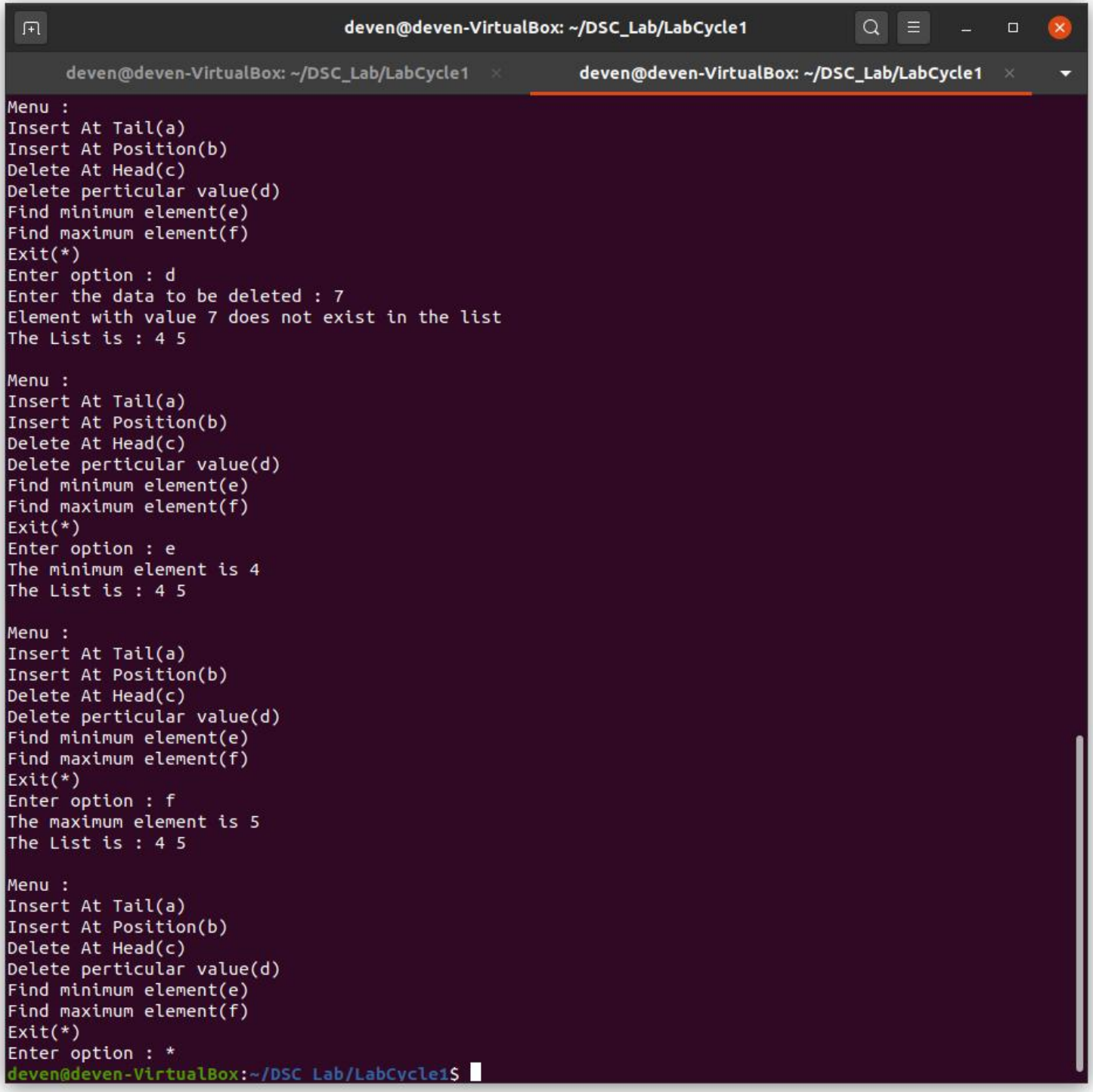
Open | singlyLL.c ~/DSC_Lab/LabCycle1 | Save

```
            int data;
            printf("Enter the data : ");
            scanf("%d",&data);
            insertTail(l,data);
        }
        else if(ch=='b')
        {
            int data,pos;
            printf("Enter the data and position : ");
            scanf("%d%d",&data,&pos);
            insertAtPos(l,data,pos);
        }
        else if(ch=='c')
            DeleteHead(l);
        else if(ch=='d')
        {
            int data;
            printf("Enter the data to be deleted : ");
            scanf("%d",&data);
            DeleteValue(l,data);
        }
        else if(ch=='e')
        {
            findMin(l);
            printf("The minimum element is %d\n",l->min);
        }
        else if(ch=='f')
        {
            findMax(l);
            printf("The maximum element is %d\n",l->max);
        }
        PrintList(l);
        printf("\n");
        printf("Menu :\n");
        printf("Insert At Tail(a)\n");
        printf("Insert At Position(b)\n");
        printf("Delete At Head(c)\n");
        printf("Delete perticular value(d)\n");
        printf("Find minimum element(e)\n");
        printf("Find maximum element(f)\n");
        printf("Exit(*)\n");
        printf("Enter option : ");
        scanf(" %c",&ch);
    }
    freeList(l);
    free(l);
    return 0;
}
```

Bracket match found on line: 183 | C | Tab Width: 4 | Ln 247, Col 2 | INS


deven@deven-VirtualBox: ~/DSC_Lab/LabCycle1
deven@deven-VirtualBox: ~/DSC_Lab/LabCycle1
deven@deven-VirtualBox: ~/DSC_Lab/LabCycle1
deven@deven-VirtualBox:~/DSC_Lab/LabCycle1$ ./a.out
Menu :
Insert At Tail(a)
Insert At Position(b)
Delete At Head(c)
Delete perticular value(d)
Find minimum element(e)
Find maximum element(f)
Exit(*)
Enter option : a
Enter the data : 1
The List is : 1
Menu :
Insert At Tail(a)
Insert At Position(b)
Delete At Head(c)
Delete perticular value(d)
Find minimum element(e)
Find maximum element(f)
Exit(*)
Enter option : b
Enter the data and position : 2 1
The List is : 2 1
Menu :
Insert At Tail(a)
Insert At Position(b)
Delete At Head(c)
Delete perticular value(d)
Find minimum element(e)
Find maximum element(f)
Exit(*)
Enter option : b
Enter the data and position : 3 5
Position does not exist
The List is : 2 1
Menu :
Insert At Tail(a)
Insert At Position(b)
Delete At Head(c)
Delete perticular value(d)
Find minimum element(e)
Find maximum element(f)
Exit(*)
Enter option : c


deven@deven-VirtualBox: ~/DSC_Lab/LabCycle1
deven@deven-VirtualBox: ~/DSC_Lab/LabCycle1
deven@deven-VirtualBox: ~/DSC_Lab/LabCycle1
Menu :
Insert At Tail(a)
Insert At Position(b)
Delete At Head(c)
Delete perticular value(d)
Find minimum element(e)
Find maximum element(f)
Exit(*)
Enter option : c
The List is : 1
Menu :
Insert At Tail(a)
Insert At Position(b)
Delete At Head(c)
Delete perticular value(d)
Find minimum element(e)
Find maximum element(f)
Exit(*)
Enter option : d
Enter the data to be deleted : 1
The List is : The list is empty
Menu :
Insert At Tail(a)
Insert At Position(b)
Delete At Head(c)
Delete perticular value(d)
Find minimum element(e)
Find maximum element(f)
Exit(*)
Enter option : a
Enter the data : 4
The List is : 4
Menu :
Insert At Tail(a)
Insert At Position(b)
Delete At Head(c)
Delete perticular value(d)
Find minimum element(e)
Find maximum element(f)
Exit(*)
Enter option : a
Enter the data : 5
The List is : 4 5


deven@deven-VirtualBox: ~/DSC_Lab/LabCycle1
deven@deven-VirtualBox: ~/DSC_Lab/LabCycle1
deven@deven-VirtualBox: ~/DSC_Lab/LabCycle1
Menu :
Insert At Tail(a)
Insert At Position(b)
Delete At Head(c)
Delete perticular value(d)
Find minimum element(e)
Find maximum element(f)
Exit(*)
Enter option : d
Enter the data to be deleted : 7
Element with value 7 does not exist in the list
The List is : 4 5
Menu :
Insert At Tail(a)
Insert At Position(b)
Delete At Head(c)
Delete perticular value(d)
Find minimum element(e)
Find maximum element(f)
Exit(*)
Enter option : e
The minimum element is 4
The List is : 4 5
Menu :
Insert At Tail(a)
Insert At Position(b)
Delete At Head(c)
Delete perticular value(d)
Find minimum element(e)
Find maximum element(f)
Exit(*)
Enter option : f
The maximum element is 5
The List is : 4 5
Menu :
Insert At Tail(a)
Insert At Position(b)
Delete At Head(c)
Delete perticular value(d)
Find minimum element(e)
Find maximum element(f)
Exit(*)
Enter option : *
deven@deven-VirtualBox:~/DSC Lab/LabCycle1$