Assignment 9

Q)

Code:

#include <stdio.h>

#include <stdlib.h>

struct node

{

int data;

struct node \*next;

struct node \*prev;

};

struct node \*head=NULL;

struct node \*tail=NULL;

int length();

void Takeinput();

void Insert();

void Delete();

void Display();

int main()

{

int choice;

while(1)

{

printf("\nMain Menu");

printf("\n1.Creation of the List");

printf("\n2.Insert");

printf("\n3.Delete");

printf("\n4.Display");

printf("\n5.Exit");

printf("\nEnter your choice");

scanf("%d",&choice);

switch (choice)

{

case 1:Takeinput();

Display();

break;

case 2:Insert();

Display();

break;

case 3:Delete();

Display();

break;

case 4:Display();

break;

case 5:exit(0);

break;

default:printf("\n Invalid Entry");

}

}

}

int length()

{

int l=0;

struct node \*temp=head;

while(temp!=NULL)

{

l++;

temp=temp->next;

}

return l;

}

void Takeinput()

{

int data;

struct node \*n=(struct node \*)malloc(sizeof(struct node));

printf("\nEnter the data");

scanf("%d",&data);

n->data=data;

n->next=NULL;

n->prev=NULL;

if(head==NULL)

{

head=tail=n;

}

else

{

tail->next=n;

n->prev=tail;

tail=n;

}

}

void Insert()

{

int l=length();

int data;

struct node \*n=(struct node \*)malloc(sizeof(struct node));

int pos;

printf("\nEnter the position at which the node is to be inserted");

scanf("%d",&pos);

if(pos>l)

{

printf("\nInvalid Location");

return;

}

printf("\nEnter the data");

scanf("%d",&data);

n->data=data;

n->next=NULL;

n->prev=NULL;

struct node \*temp=head;

if(pos==0)

{

n->next=head;

head->prev=n;

head=n;

}

else if(pos==l)

{

while(temp->next!=NULL)

{

temp=temp->next;

}

temp->next=n;

n->prev=temp;

}

else

{

int i=0;

while(i<pos-1 && temp!=NULL)

{

temp=temp->next;

i++;

}

if(temp!=NULL)

{

n->next=temp->next;

temp->next->prev=n;

n->prev=temp;

temp->next=n;

}

}

}

void Delete()

{

int l=length();

struct node \*temp=head;

int pos;

if(head==NULL)

{

return;

}

else if(head->next==NULL)

{

head=NULL;

free(head);

}

else

{

printf("\nEnter the position at which the node is to be deleted");

scanf("%d",&pos);

if(pos>=l)

{

printf("\nInvalid position");

}

else if(pos==0)

{

temp=head;

head=head->next;

head->prev=NULL;

free(temp);

}

else if(pos==l-1)

{

while(temp->next!=NULL)

{

temp=temp->next;

}

temp->prev->next=NULL;

free(temp);

}

else

{

int i=0;

while(i<pos && temp->next!=NULL)

{

temp=temp->next;

i++;

}

if(temp->next!=NULL)

{

temp->prev->next=temp->next;

temp->next->prev=temp->prev;

free(temp);

}

}

}

}

void Display()

{

struct node \*temp=head;

if(head==NULL)

{

printf("\nThe list is empty");

}

printf("\n");

while(temp!=NULL)

{

printf("%d ",temp->data);

temp=temp->next;

}

}

Output:

Main Menu

1.Creation of the List

2.Insert

3.Delete

4.Display

5.Exit

Enter your choice1

Enter the data34

34

Main Menu

1.Creation of the List

2.Insert

3.Delete

4.Display

5.Exit

Enter your choice1

Enter the data475

34 475

Main Menu

1.Creation of the List

2.Insert

3.Delete

4.Display

5.Exit

Enter your choice2

Enter the position at which the node is to be inserted0

Enter the data23

23 34 475

Main Menu

1.Creation of the List

2.Insert

3.Delete

4.Display

5.Exit

Enter your choice2

Enter the position at which the node is to be inserted9

Invalid Location

23 34 475

Main Menu

1.Creation of the List

2.Insert

3.Delete

4.Display

5.Exit

Enter your choice2

Enter the position at which the node is to be inserted2

Enter the data190

23 34 190 475

Main Menu

1.Creation of the List

2.Insert

3.Delete

4.Display

5.Exit

Enter your choice2

Enter the position at which the node is to be inserted4

Enter the data64

23 34 190 475 64

Main Menu

1.Creation of the List

2.Insert

3.Delete

4.Display

5.Exit

Enter your choice3

Enter the position at which the node is to be deleted0

34 190 475 64

Main Menu

1.Creation of the List

2.Insert

3.Delete

4.Display

5.Exit

Enter your choice2

Enter the position at which the node is to be inserted0

Enter the data78

78 34 190 475 64

Main Menu

1.Creation of the List

2.Insert

3.Delete

4.Display

5.Exit

Enter your choice3

Enter the position at which the node is to be deleted6

Invalid position

78 34 190 475 64

Main Menu

1.Creation of the List

2.Insert

3.Delete

4.Display

5.Exit

Enter your choice3

Enter the position at which the node is to be deleted2

78 34 475 64

Main Menu

1.Creation of the List

2.Insert

3.Delete

4.Display

5.Exit

Enter your choice2

Enter the position at which the node is to be inserted3

Enter the data56

78 34 475 56 64

Main Menu

1.Creation of the List

2.Insert

3.Delete

4.Display

5.Exit

Enter your choice2

Enter the position at which the node is to be inserted0

Enter the data1

1 78 34 475 56 64

Main Menu

1.Creation of the List

2.Insert

3.Delete

4.Display

5.Exit

Enter your choice3

Enter the position at which the node is to be deleted5

1 78 34 475 56

Main Menu

1.Creation of the List

2.Insert

3.Delete

4.Display

5.Exit

Enter your choice4

1 78 34 475 56

Main Menu

1.Creation of the List

2.Insert

3.Delete

4.Display

5.Exit

Enter your choice5

**...Program finished with exit code 0**

**Press ENTER to exit console.**

Q)

Code:

#include <stdio.h>

#include <stdlib.h>

struct node

{

int data;

struct node \*next;

struct node \*prev;

};

struct node \*head=NULL;

struct node \*tail=NULL;

void insert();

void display();

struct node \*reverse();

int main()

{

int n;

printf("\n Enter the no of elements that you wan to insert in the linked list: ");

scanf("%d",&n);

int i;

for(i=0;i<n;i++)

{

insert();

}

printf("\n The elements present in the linked list are: ");

display();

printf("\n The reversed linked list is: ");

head=reverse();

display();

return 0;

}

void insert()

{

int data;

struct node \*n=(struct node \*)malloc(sizeof(struct node));

printf("\nEnter the data");

scanf("%d",&data);

n->data=data;

n->next=NULL;

n->prev=NULL;

if(head==NULL)

{

head=tail=n;

}

else

{

tail->next=n;

n->prev=tail;

tail=n;

}

}

void display()

{

struct node \*temp=head;

printf("\n");

while(temp!=NULL)

{

printf("%d ",temp->data);

temp=temp->next;

}

}

struct node \*reverse()

{

struct node \*curr=head;

struct node \*p=NULL;

struct node \*n=curr->next;

while(curr!=NULL)

{

curr->next=p;

p=curr;

curr=n;

if(curr)

n=curr->next;

}

return p;

}

Output:

Enter the no of elements that you wan to insert in the linked list: 20

Enter the data1

Enter the data2

Enter the data3

Enter the data4

Enter the data5

Enter the data7

Enter the data23

Enter the data23

Enter the data45

Enter the data67

Enter the data123

Enter the data94

Enter the data345

Enter the data10

Enter the data37

Enter the data2

Enter the data34

Enter the data56

Enter the data23

Enter the data45

 The elements present in the linked list are:

1 2 3 4 5 7 23 23 45 67 123 94 345 10 37 2 34 56 23 45

 The reversed linked list is:

45 23 56 34 2 37 10 345 94 123 67 45 23 23 7 5 4 3 2 1

**...Program finished with exit code 0**

**Press ENTER to exit console.**

Q)

Code:

#include<stdio.h>

#include<stdlib.h>

struct node

{

int data;

struct node \*next;

};

struct node \*head;

void beginsert()

{

struct node \*ptr,\*temp;

int item;

ptr = (struct node \*)malloc(sizeof(struct node));

if(ptr == NULL)

{

printf("\nOverflow");

}

else

{

printf("\nEnter the node data: ");

scanf("%d",&item);

ptr -> data = item;

if(head == NULL)

{

head = ptr;

ptr -> next = head;

}

else

{

temp = head;

while(temp->next != head)

temp = temp->next;

ptr->next = head;

temp -> next = ptr;

head = ptr;

}

printf("\nNode inserted in beginning\n");

}

}

void lastinsert()

{

struct node \*ptr,\*temp;

int item;

ptr = (struct node \*)malloc(sizeof(struct node));

if(ptr == NULL)

{

printf("\nOVERFLOW\n");

}

else

{

printf("\nEnter Data: ");

scanf("%d",&item);

ptr->data = item;

if(head == NULL)

{

head = ptr;

ptr -> next = head;

}

else

{

temp = head;

while(temp -> next != head)

{

temp = temp -> next;

}

temp -> next = ptr;

ptr -> next = head;

}

printf("\nNode inserted at last\n");

}

}

void begin\_delete()

{

struct node \*ptr;

if(head == NULL)

{

printf("\nUNDERFLOW");

}

else if(head->next == head)

{

head = NULL;

free(head);

printf("\nNode deleted\n");

}

else

{ ptr = head;

while(ptr -> next != head)

ptr = ptr -> next;

ptr->next = head->next;

free(head);

head = ptr->next;

printf("\nNode deleted\n");

}

}

void last\_delete()

{

struct node \*ptr, \*preptr;

if(head==NULL)

{

printf("\nUNDERFLOW");

}

else if (head ->next == head)

{

head = NULL;

free(head);

printf("\nNode deleted\n");

}

else

{

ptr = head;

while(ptr ->next != head)

{

preptr=ptr;

ptr = ptr->next;

}

preptr->next = ptr -> next;

free(ptr);

printf("\nNode deleted\n");

}

}

void display()

{

struct node \*ptr;

ptr=head;

if(head == NULL)

{

printf("\nEmpty");

}

else

{

printf("\nElements in linked list are: \n");

while(ptr -> next != head)

{

printf("%d\n", ptr -> data);

ptr = ptr -> next;

}

printf("%d\n", ptr -> data);

}

}

int main ()

{

int choice =0;

while(choice != 6)

{

printf("\n1.Insert in beginning\n2.Insert at last\n3.Delete from Beginning\n4.Delete from last\n5.Display\n6.Exit\n");

printf("\nEnter your choice: ");

scanf("\n%d",&choice);

switch(choice)

{

case 1: beginsert();

break;

case 2: lastinsert();

break;

case 3: begin\_delete();

break;

case 4: last\_delete();

break;

case 5: display();

break;

case 6: exit(0);

break;

default: printf("Wrong choice");

}

}

}

Output:

1.Insert in beginning

2.Insert at last

3.Delete from Beginning

4.Delete from last

5.Display

6.Exit

Enter your choice: 1

Enter the node data: 1

Node inserted in beginning

1.Insert in beginning

2.Insert at last

3.Delete from Beginning

4.Delete from last

5.Display

6.Exit

Enter your choice: 2

Enter Data: 3

Node inserted at last

1.Insert in beginning

2.Insert at last

3.Delete from Beginning

4.Delete from last

5.Display

6.Exit

Enter your choice: 1

Enter the node data: 3

Node inserted in beginning

1.Insert in beginning

2.Insert at last

3.Delete from Beginning

4.Delete from last

5.Display

6.Exit

Enter your choice: 1

Enter the node data: 4

Node inserted in beginning

1.Insert in beginning

2.Insert at last

3.Delete from Beginning

4.Delete from last

5.Display

6.Exit

Enter your choice: 5

Elements in linked list are:

4

3

1

3

1.Insert in beginning

2.Insert at last

3.Delete from Beginning

4.Delete from last

5.Display

6.Exit

Enter your choice: 3

Node deleted

1.Insert in beginning

2.Insert at last

3.Delete from Beginning

4.Delete from last

5.Display

6.Exit

Enter your choice: 3

Node deleted

1.Insert in beginning

2.Insert at last

3.Delete from Beginning

4.Delete from last

5.Display

6.Exit

Enter your choice: 5

Elements in linked list are:

1

3

1.Insert in beginning

2.Insert at last

3.Delete from Beginning

4.Delete from last

5.Display

6.Exit

Enter your choice: 6

**...Program finished with exit code 0**

**Press ENTER to exit console.**