#include<stdio.h>

#include<stdlib.h>

typedef struct node

{

int data;

struct node\* next;

}N;

N \*getnode();

N \*create();

int count(N \*first);

N \*search(N \*first,int val);

N \*insert\_beg(N \*first);

N \*insert\_betw(N \*first);

N \*insert\_end(N \*first);

N \*delete(N \*first);

N \*reverse(N \*first);

void printReverse(N \*first);

void display(N \*first);

void main()

{

int ch,ls,val,p;

N \*first=NULL;

while(1)

{

printf("Enter your choice \n 1: Create a list \n 2: Search \n 3: Count \n 4: Insert a node \n 5: Delete a node \n 6: Display a list \n 7: Reverse the list \n 8: Display in Reverse \n 9: Exit \n");

scanf("%d",&ch);

switch(ch)

{

case 1:

first=create();

break;

case 2:

if(first!=NULL)

{

printf("Enter the value to be searched \n");

scanf("%d",&val);

first=search(first,val);

if(first!=NULL)

printf("The data is %d \n",first->data);

else

printf("Data not found \n");

}

else

printf("The list is empty \n");

break;

case 3:

ls=count(first);

printf("The no. of nodes in the list are %d \n",ls);

break;

case 4:

printf("Enter your choice \n 1: Insert at beginning \n 2: Insert in between \n 3: Insert in end \n");

scanf("%d",&p);

switch(p)

{

case 1:

first=insert\_beg(first);

printf("The list is \n");

display(first);

printf("\n");

break;

case 2:

first=insert\_betw(first);

printf("The list is \n");

display(first);

printf("\n");

break;

case 3:

first=insert\_end(first);

printf("The list is \n");

display(first);

printf("\n");

break;

}

break;

case 5:

first=delete(first);

break;

case 6:

if(first==NULL)

printf("The list is empty");

else

{

printf("The list is \n");

display(first);

printf("\n");

}

break;

case 7:

first=reverse(first);

break;

case 8:

printReverse(first);

printf("\n");

break;

case 9:

exit(0);

}

}

}

N \*getnode()

{

N \*temp;

temp=(N\*)malloc(sizeof(N));

printf("Enter the data to be inserted:");

scanf("%d",&temp->data);

temp->next=NULL;

return temp;

}

N \*create()

{

int i,n;

N \*first=NULL;

N \*last=NULL;

N \*temp=NULL;

printf("Enter no. of nodes \n");

scanf("%d",&n);

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

{

temp=getnode();

if(first==NULL)

first=last=temp;

else

last->next=temp;

last=temp;

}

return first;

}

void display(N \*first)

{

N \*temp;

temp=first;

while(temp!=NULL)

{

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

temp=temp->next;

}

}

int count(N \*first)

{

int i;

N \*temp;

temp=first;

while(temp!=NULL)

{

i++;

temp=temp->next;

}

return i;

}

N \*search(N \*first,int val)

{

N \*temp;

temp=first;

while(temp!=NULL)

{

if(temp->data==val)

return temp;

temp=temp->next;

}

return NULL;

}

N \*insert\_beg(N \*first)

{

N \*new;

new=getnode();

new->next=first;

return new;

}

N \*insert\_betw(N \*first)

{

N \*new,\*temp;

int p;

new=getnode();

printf("Enter the data of the link after which you want to insert \n");

scanf("%d",&p);

temp=search(first,p);

new->next=temp->next;

temp->next=new;

return first;

}

N \*insert\_end(N \*first)

{

N \*new,\*temp;

new=getnode();

temp=first;

while(temp->next!=NULL)

temp=temp->next;

temp->next=new;

return first;

}

N \*delete(N \*first)

{

N \*curr,\*prev;

int val;

curr=first;

printf("Enter the node value \n");

scanf("%d",&val);

while(curr!=NULL)

{

if(curr->data==val)

break;

prev=curr;

curr=curr->next;

}

if(curr!=NULL)

{

if(curr==first)

first=curr->next;

else

prev->next=curr->next;

printf("Data %d deleted \n",curr->data);

free(curr);

}

else

printf("Data not found \n");

return first;

}

N \*reverse(N \*first)

{

N \*a,\*b,\*c;

c=NULL;

a=first;

b=NULL;

while(a!=NULL)

{

b=a->next;

a->next=c;

c=a;

a=b;

}

first=c;

display(first);

printf("\n");

return first;

}

void printReverse(N \*first)

{

N \*head;

head=first;

if (head == NULL)

return;

printReverse(head->next);

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

}

Output:







