**Program: Stack using Linked List**

#include<stdio.h>

#include<stdlib.h>

#include<math.h>

#include<malloc.h>

struct node

{

int data;

struct node \*next;

};

typedef struct node node;

node \*start = NULL;

node \*insertend(node \*);

node \*deleteend(node \*);

void checkend(node \*);

void checkempty(node \*);

void checkfull(node \*);

void checknodes(node \*);

void display(node\*);

int main()

{

int choice,ans;

printf("\n 1: Push a element");

printf("\n 2: Pop the elem");

printf("\n 3: Peek topmost");

printf("\n 4: Check if stack is empty");

printf("\n 5: Check if stack is full");

printf("\n 6: Check the number of elements");

printf("\n 7: Display elements");

do

{

printf(" \n Enter your choice");

scanf("%d",&choice);

switch(choice)

{

case 1:

start=insertend(start);

break;

case 2:

start=deleteend(start);

break;

case 3:

checkend(start);

break;

case 4:

checkempty(start);

break;

case 5:

checkfull(start);

break;

case 6:

checknodes(start);

break;;

case 7:

display(start);

break;

default:

printf("Wrong choice");

}

printf("\n Press 1 to choose another option");

scanf("%d",&ans);

}while(ans ==1);

return 0;

}

node \*insertend(node \*start)

{

node \*ptr, \*newnode;

int num,ans;

do

{

printf("\n Enter the data : ");

scanf("%d", &num);

newnode = ( node \*)malloc(sizeof(node));

newnode -> data = num;

if(start == NULL)

{

start = newnode;

newnode -> next = NULL;

}

else

{

ptr=start;

while(ptr->next!=NULL)

ptr=ptr->next;

ptr->next = newnode;

newnode->next=NULL;

}

printf("\n Press 1 to continue adding elements");

scanf("%d",&ans);

}while(ans==1);

return start;

}

node \*deleteend(node \*start)

{

node \*ptr, \*preptr;

ptr = start;

while(ptr -> next != NULL && start!=NULL)

{

preptr = ptr;

ptr = ptr -> next;

}

preptr -> next = NULL;

free(ptr);

return start;

}

void checkend(node \*start)

{

node \*ptr;

ptr=start;

while(ptr->next != NULL && start!=NULL)

ptr=ptr->next;

printf("\n The topmost element is %d",ptr->data);

}

void checkempty(node \*start)

{

if(start==NULL)

printf("\n Stack Underflow");

else

printf("\n Stack is not empty");

}

void checkfull(node \*start)

{

node \*newnode;

newnode = ( node \*)malloc(sizeof(node));

if(newnode ==NULL)

printf("\n Stack Overflow");

else

printf("\n Stack is not full");

}

void checknodes(node \*start)

{

int count=0;

node \*ptr;

ptr=start;

while(ptr!=NULL )

{

count=count+1;

ptr=ptr->next;

}

printf("\n There are %d elements",count);

}

void display(node \*start)

{

int i;

if (start==NULL)

{

printf("\n Stack is empty\n");

}

else

{

printf("\n The elements of the stack are \n");

node \*ptr;

ptr=start;

while(ptr!=NULL )

{

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

ptr=ptr->next;

}

}

printf("\n");

}

**Output:**

1: Push a element

2: Pop the elem

3: Peek topmost

4: Check if stack is empty

5: Check if stack is full

6: Check the number of elements

7: Display elements

Enter your choice1

Enter the data : 12

Press 1 to continue adding elements1

Enter the data : 13

Press 1 to continue adding elements1

Enter the data : 14

Press 1 to continue adding elements1

Enter the data : 15

Press 1 to continue adding elements0

Press 1 to choose another option1

Enter your choice7

The elements of the stack are

12 13 14 15

Press 1 to choose another option1

Enter your choice6

There are 4 elements

Press 1 to choose another option1

Enter your choice3

The topmost element is 15

Press 1 to choose another option1

Enter your choice2

Press 1 to choose another option1

Enter your choice7

The elements of the stack are

12 13 14

Press 1 to choose another option1

Enter your choice4

Stack is not empty

Press 1 to choose another option1

Enter your choice5

Stack is not full

Press 1 to choose another option0