**Implement Stack using List**

# include <stdio.h>

# include <conio.h>

# include <stdlib.h>

struct node

{

int data;

struct node \*next;

};

typedef struct node NODE;

NODE \*start = NULL;

int menu()

{

 int ch;

 system("cls");

printf("\n 1.Create a list ");

 printf("\n--------------------------");

 printf("\n 2. PUSH ");

 printf("\n 3. POP");

 printf("\n 4. Displaying the list");

printf("\n 5. Quit");

 printf("\n--------------------------");

 printf("\n\n Enter your choice: ");

 scanf("%d",&ch);

 return ch;

}

NODE\* getnode()

{

 NODE \* newnode;

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

 printf("\n Enter data: ");

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

 newnode -> next = NULL;

 return newnode;

}

void createlist(int n)

{

 int i;

 NODE \*newnode, \*temp;

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

 {

 newnode = getnode();

 if(start == NULL)

 {

 start = newnode;

 }

 else

 {

 temp = start;

 while(temp -> next != NULL)

 temp = temp -> next;

 temp -> next = newnode;

 }

 }

}

void display()

{

 NODE \*temp;

 temp = start;

 printf("\n The contents of List (Left to Right): \n");

 if(start == NULL)

 {

 printf("\n Empty List");

 return;

 }

 else

 {

 while(temp != NULL)

 {

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

 temp = temp -> next;

 }

 }

}

void push()

{

 NODE \*newnode;

 newnode = getnode();

 if(start == NULL)

 {

 start = newnode;

 }

 else

 {

 newnode -> next = start;

 start = newnode;

 }

}

void pop()

{

 NODE \*temp;

 if(start == NULL)

 {

 printf("\n No nodes are exist..");

 return ;

 }

 else

 {

 temp = start;

 start = temp -> next;

 printf("\n Node deleted %d", temp->data);

 free(temp);

 }

}

void main(void)

{

 int ch, n;

 while(1)

 {

 ch = menu();

 switch(ch)

 {

 case 1:

 if(start == NULL)

 {

 printf("\n Number of nodes you want to create:");

 scanf("%d", &n);

 createlist(n);

 printf("\n List created..");

 }

 else

 printf("\n List is already created..");

 break;

 case 2:

 push();

 break;

 case 3:

 pop();

 break;

 case 4:

 display();

 break;

 case 5:

 exit(0);

 }

 getch();

 }

}