**Write a C program to implement a stack using Array and linked List implementation and execute the following operation on stack.**

1. **Push an element into a stack**
2. **Pop an element from a stack**
3. **Return the Top most element from a stack**
4. **Display the elements in a stack**

**Algorithm:**

**Code:**

**Array Implementation:**

#include

#define MAX 5

int Stack[MAX], top = -1;

int IsFull();

int IsEmpty();

void Push(int ele);

void Pop();

void Top();

void Display();

int main()

{ int ch, e;

do {

printf("1.PUSH 2.POP 3.TOP 4.DISPLAY 5.EXIT");

printf("\nEnter your choice : ");

scanf("%d", &ch);

switch(ch) {

case 1: printf("Enter the element : ");

scanf("%d", &e);

Push(e);

break;

case 2:

Pop();

break;

case 3:

Top();

break;

case 4:

Display();

break; }

} while(ch <= 4);

return 0; }

int IsFull() {

if(top == MAX - 1)

return 1;

else

return 0; }

int IsEmpty()

{

if(top == -1)

return 1;

else

return 0;

}

void Push(int ele)

{ if(IsFull())

printf("Stack Overflow...!\n");

else { top = top + 1; Stack[top] = ele; }

}

void Pop() {

if(IsEmpty())

printf("Stack Underflow...!\n");

else {

printf("%d\n", Stack[top]); top = top - 1; }

}

void Top()

{

if(IsEmpty())

printf("Stack Underflow...!\n");

else

printf("%d\n", Stack[top]); }

void Display() {

int i;

if(IsEmpty())

printf("Stack Underflow...!\n");

else {

for(i = top; i >= 0; i--)

printf("%d\t", Stack[i]);

printf("\n");

}

}

**Linked List Implementation:**

#include <stdio.h>

#include <stdlib.h>

struct Node {

int data;

struct Node\* next;

};

struct LinkedListStack {

struct Node\* top;

};

struct LinkedListStack\* createLinkedListStack() {

struct LinkedListStack\* stack = (struct LinkedListStack\*)malloc(sizeof(struct LinkedListStack));

stack->top = NULL;

return stack;

}

int isLinkedListStackEmpty(struct LinkedListStack\* stack) {

return (stack->top == NULL);

}

void pushLinkedList(struct LinkedListStack\* stack, int data) {

struct Node\* newNode = (struct Node\*)malloc(sizeof(struct Node));

newNode->data = data;

newNode->next = stack->top;

stack->top = newNode;

}

int popLinkedList(struct LinkedListStack\* stack) {

if (isLinkedListStackEmpty(stack)) {

printf("Stack Underflow\n");

return -1;

}

struct Node\* temp = stack->top;

int data = temp->data;

stack->top = temp->next;

free(temp);

return data;

}

int main() {

struct LinkedListStack\* linkedListStack = createLinkedListStack();

// Pushing elements onto the stack

pushLinkedList(linkedListStack, 40);

pushLinkedList(linkedListStack, 50);

pushLinkedList(linkedListStack, 60);

// Popping elements from the stack

printf("Popped from linkedListStack: %d\n", popLinkedList(linkedListStack));

printf("Popped from linkedListStack: %d\n", popLinkedList(linkedListStack));

printf("Popped from linkedListStack: %d\n", popLinkedList(linkedListStack));

return 0;

}

**OUTPUT :**

**Stack Menu**

**1. Push**

**2. Pop**

**3. Display**

**4. Exit**

**Enter your choice: 1**

**Enter data to push: 10**

**10 pushed to stack**

**Enter your choice: 1**

**Enter data to push: 20**

**20 pushed to stack**

**Enter your choice: 1**

**Enter data to push: 30**

**30 pushed to stack**

**Enter your choice: 3**

**Elements in stack: 30 20 10**

**Enter your choice: 2**

**Popped element: 30**

**Enter your choice: 3**

**Elements in stack: 20 10**

**Enter your choice: 2**

**Popped element: 20**

**Enter your choice: 2**

**Popped element: 10**

**Enter your choice: 2**

**Stack Underflow**

**Enter your choice: 3**

**Stack is empty**

**Enter your choice: 4**

**Exiting program**