IMPLEMENTATION OF STACK

AIM:

The aim of the program is to execute stack using array and linked list.

ALGORITHM:

1.Start

2.Initialize: Create an empty stack and set its maximum size if applicable.

3.Push: To add an element onto the stack, increment the stack pointer and place the new element into the location pointed to by the stack pointer.

4.Pop: To remove an element from the stack, return the element at the current stack pointer location and then decrement the stack pointer.

5.Peek: To view the top element of the stack without removing it, return the element at the current stack pointer location.

6.isEmpty: Check if the stack is empty by examining if the stack pointer is pointing to the base of the stack.

7.isFull (if there's a maximum size): Check if the stack is full by comparing the stack pointer to the maximum size.

8.End

PROGRAM USING ARRAY:

```
#include <stdio.h>

#define MAX_SIZE 10

typedef struct {
    int data[MAX_SIZE];
    int top;
} Stack;

void initStack(Stack* stack) {
    stack->top = -1;
}

int isEmpty(Stack* stack) {
```

```
    return stack->top == -1;
}

int isFull(Stack* stack) {
    return stack->top == MAX_SIZE - 1;
}

void push(Stack* stack, int element) {
    if (isFull(stack)) {
        printf("Stack overflow!\n");
        return;
    }
    stack->data[++stack->top] = element;
}

int pop(Stack* stack) {
    if (isEmpty(stack)) {
        printf("Stack underflow!\n");
        return -1;
    }
    return stack->data[stack->top--];
}

int top(Stack* stack) {
    if (isEmpty(stack)) {
        printf("Stack is empty!\n");
        return -1;
    }
    return stack->data[stack->top];
}

void displayStack(Stack* stack) {
    int i;
    for (i = 0; i <= stack->top; i++) {
        printf("%d ", stack->data[i]);
    }
    printf("\n");
}

int main() {
    Stack stack;
    initStack(&stack);

    int choice, element;
```

```
    while (1) {
        printf("1. Push\n");
        printf("2. Pop\n");
        printf("3. Top\n");
        printf("4. Display\n");
        printf("5. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                printf("Enter element to push: ");
                scanf("%d", &element);
                push(&stack, element);
                break;
            case 2:
                element = pop(&stack);
                if (element != -1) {
                    printf("Popped element: %d\n", element);
                }
                break;
            case 3:
                element = top(&stack);
                if (element != -1) {
                    printf("Top element: %d\n", element);
                }
                break;
            case 4:
                displayStack(&stack);
                break;
            case 5:
                return 0;
            default:
                printf("Invalid choice!\n");
        }
    }

    return 0;
}
```

OUTPUT:

1. Push
2. Pop

```
3. Top
4. Display
5. Exit
Enter your choice: 1
Enter element to push: 10
1. Push
2. Pop
3. Top
4. Display
5. Exit
Enter your choice: 5
```

PROGRAM USING LINKED LIST:

```
#include <stdio.h>

#define MAX_SIZE 10

typedef struct {
    int data[MAX_SIZE];
    int top;
} Stack;

void initStack(Stack* stack) {
    stack->top = -1;
}

int isEmpty(Stack* stack) {
    return stack->top == -1;
}

int isFull(Stack* stack) {
    return stack->top == MAX_SIZE - 1;
}

void push(Stack* stack, int element) {
    if (isFull(stack)) {
        printf("Stack overflow!\n");
        return;
    }
    stack->data[++stack->top] = element;
}
```

```
int pop(Stack* stack) {
    if (isEmpty(stack)) {
        printf("Stack underflow!\n");
        return -1;
    }
    return stack->data[stack->top--];
}

int top(Stack* stack) {
    if (isEmpty(stack)) {
        printf("Stack is empty!\n");
        return -1;
    }
    return stack->data[stack->top];
}

void displayStack(Stack* stack) {
    int i;
    for (i = 0; i <= stack->top; i++) {
        printf("%d ", stack->data[i]);
    }
    printf("\n");
}

int main() {
    Stack stack;
    initStack(&stack);

    int choice, element;

    while (1) {
        printf("1. Push\n");
        printf("2. Pop\n");
        printf("3. Top\n");
        printf("4. Display\n");
        printf("5. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                printf("Enter element to push: ");
                scanf("%d", &element);
                push(&stack, element);
```

```
            break;
        case 2:
            element = pop(&stack);
            if (element!= -1) {
                printf("Popped element: %d\n", element);
            }
            break;
        case 3:
            element = top(&stack);
            if (element!= -1) {
                printf("Top element: %d\n", element);
            }
            break;
        case 4:
            displayStack(&stack);
            break;
        case 5:
            return 0;
        default:
            printf("Invalid choice!\n");
    }
  }

  return 0;
}
```

OUTPUT:

```
1. Push
2. Pop
3. Top
4. Display
5. Exit
Enter your choice: 1
Enter element to push: 10
1. Push
2. Pop
3. Top
4. Display
5. Exit
Enter your choice: 5
```

RESULT:

The output is verified successfully for the above program.