

**Practical no. 1**

**Name - Ankit yadav**

**Roll no - 48**

**Section - B**

**Aim:** To implement a menu-driven program in C to perform basic stack operations including push, pop, peek, and display using an array.res.

**TOOLS/SOFTWARE REQUIRED:**

- 1.C Compiler (for example: GCC, Turbo C)
- 2.Text Editor (for example: VS Code, Notepad++)

**THEORY:**

A stack is a linear data structure that follows the Last-In-First-Out (LIFO) principle, where the most recently added element is accessed first. Stack operations include:

- Push: Add an element to the top.
- Pop: Remove the topmost element.
- Peek: View the top element without removing it.
- Display: Show all stack elements.

**CODES:**

```
#include <stdio.h>

#define MAX 100

int stack[MAX], top = -1;
void push(int val) {
    if(top == MAX - 1)
        printf("Stack Overflow\n");
    else
        stack[++top] = val;
}

void pop() {
    if(top == -1)
        printf("Stack Underflow\n");
    else
        printf("Popped element: %d\n", stack[top--]);
}
```

```
}  
void peek() {  
    if(top == -1)  
        printf("Stack is empty\n");  
    else  
        printf("Top element: %d\n", stack[top]);  
}  
void display() {  
    if(top == -1)  
        printf("Stack is empty\n");  
    else {  
        printf("Stack elements:\n");  
        for(int i = top; i >= 0; i--)  
            printf("%d\n", stack[i]);  
    }  
}  
int main() {  
    int choice, val;  
    while(1) {  
        printf("\n1.Push\n2.Pop\n3.Peek\n4.Display\n5.Exit\n");  
        printf("Enter your choice: ");  
        scanf("%d", &choice);  
        switch(choice) {  
            case 1:  
                printf("Enter value to push: ");  
                scanf("%d", &val);  
                push(val);  
                break;  
            case 2:
```

```
        pop();
        break;
    case 3:
        peek();
        break;
    case 4:
        display();
        break;
    case 5:
        return 0;
    default:
        printf("Invalid choice\n");
    }
}
return 0;
}
```

## RESULT:

```
C:\Windows\System32\cmd.e  X  +  v

Microsoft Windows [Version 10.0.26100.6725]
(c) Microsoft Corporation. All rights reserved.

C:\Users\ASUS\OneDrive\Desktop\MCA\DSA\Practical\Practical1>gcc practical1.c

C:\Users\ASUS\OneDrive\Desktop\MCA\DSA\Practical\Practical1>.\a.exe

1.Push
2.Pop
3.Peek
4.Display
5.Exit
Enter your choice: 1
Enter value to push: 5

1.Push
2.Pop
3.Peek
4.Display
5.Exit
Enter your choice: 4
Stack elements:
5

1.Push
2.Pop
3.Peek
4.Display
5.Exit
Enter your choice:
```

## CONCLUSION:

A stack was implemented using an array with all basic operations. The menu-driven approach makes it user-friendly and demonstrates real-time stack usage.

## VIVA QUESTIONS:

1. What is a stack and how does it work?  
A stack is a data structure that operates on the principle of LIFO—last element inserted is the first one to be removed.

2. What is the difference between stack and queue?  
 Stack uses LIFO, while queue uses FIFO (First-In-First-Out); in queue, the first element inserted is removed first.
3. What causes stack overflow and underflow?  
 Overflow occurs when pushing onto a full stack; underflow occurs when popping from an empty stack.
4. How is a stack implemented in memory?  
 It can be implemented using arrays (fixed size) or linked lists (dynamic size).
5. What are some real-life applications of stacks?  
 Function call management, expression evaluation, undo operations in editors, and parsing expressions.

**For Faculty use only**

Correction parameter	Formative Assessment (40%)	Timely completion of Practical (40%)	Attendance Learning attitude (20%)	Total
Marks Obtained				