

1a. Write a program to simulate the working of stack using an array with the following:

i) PUSH

ii) POP

iii) Display

Program should print appropriate messages for stack overflow and stack underflow.

```
#include <stdio.h>
#define MAX 5
int stack[MAX];
int top = -1;
void push(int value) {
    if (top == MAX - 1) {
        printf("Stack Overflow! Cannot push %d\n", value);
    } else {
        stack[++top] = value;
        printf("Pushed %d onto the stack.\n", value);
    }
}
void pop() {
    if (top == -1) {
        printf("Stack Underflow! Cannot pop.\n");
    } else {
        printf("Popped %d from the stack.\n", stack[top--]);
    }
}
void peek() {
    if (top == -1) {
        printf("Stack is empty. Nothing to peek.\n");
    } else {
        printf("Top element: %d\n", stack[top]);
    }
}
void display() {
    if (top == -1) {
        printf("Stack is empty.\n");
    } else {
        printf("Stack elements: ");
        for (int i = top; i >= 0; i--) {
            printf("%d ", stack[i]);
        }
        printf("\n");
    }
}
int main() {
    int choice, value;
    while (1) {
        printf("\n1. PUSH\n2. POP\n3. PEEK\n4. DISPLAY\n5. EXIT\n");
        printf("Choose an operation: ");
        scanf("%d", &choice);
        switch (choice) {
            case 1:
                printf("Enter value to PUSH: ");
                scanf("%d", &value);
                push(value);
                break;
```

```

case 2:
pop();
break;
case 3:
peek();
break;
case 4:
display();
break;
case 5:
printf("Exiting...\n");
return 0;
default:
printf("Invalid choice.\n");
}
}
return 0;

```

output:

```

D:\DS LAB\DS LAB PROGRAM
1. PUSH
2. POP
3. PEEK
4. DISPLAY
5. EXIT
Choose an operation: 1
Enter value to PUSH: 20
Pushed 20 onto the stack.

1. PUSH
2. POP
3. PEEK
4. DISPLAY
5. EXIT
Choose an operation: 1
Enter value to PUSH: 30
Pushed 30 onto the stack.

1. PUSH
2. POP
3. PEEK
4. DISPLAY
5. EXIT
Choose an operation: 1
Enter value to PUSH: 45
Pushed 45 onto the stack.

1. PUSH
2. POP
3. PEEK
4. DISPLAY
5. EXIT
Choose an operation: 2
Popped 45 from the stack.
}

```

1. PUSH
2. POP
3. PEEK
4. DISPLAY
5. EXIT

Choose an operation: 2

Popped 45 from the stack.

1. PUSH
2. POP
3. PEEK
4. DISPLAY
5. EXIT

Choose an operation: 4

Stack elements: 30 20

1. PUSH
2. POP
3. PEEK
4. DISPLAY
5. EXIT

Choose an operation: 3

Top element: 30