K. J. Somaiya College of Engineering, Mumbai-77
(A Constituent College of Somaiya Vidyavihar University)
Department of Electronics and Computer Engineering

Course Name:	**Data Structures Laboratory**	**Semester:**	**III**
Date of Performance:	**28-07-2025**	**Batch No:**	**A1**
Faculty Name:	**Prof. Sushma Kadge**	**Roll No:**	**20**
Faculty Sign & Date:		**Grade/Marks:**	**/25**

Experiment No: 3
Title: Application of Stack

Aim and Objective of the Experiment:

To understand stack operation.
Write a program for stack using arrays. Given A[] = {21,34,45,21,60}, perform Push, Pop operations and Display Stack contents.

COs to be achieved:

CO1 : Understand and implement the different data structures used in problem solving

CO2: Apply linear and non-linear data structure in application development

Books/Journals/Websites referred:

1. PPT

Tools required:

DEV C/C++ compiler/ Code blocks C compiler/VS Code Python Compiler

Theory:

A stack is a linear data structure in which insertion and deletion of elements are done at only one end, which is known as the top of the stack. Stack is called a last-in, first-out (LIFO) structure because the last element which is added to the stack is the first element which is deleted from the stack.

Implementation details:

1. Enlist all the Steps followed and various options explored

Ans: The program demonstrates basic stack operations (push, pop, display) using an array and a menu-driven interface.

2. Explain your program logic and methods used.

Ans: It follows the LIFO principle with checks for overflow and underflow, using functions for modularity.

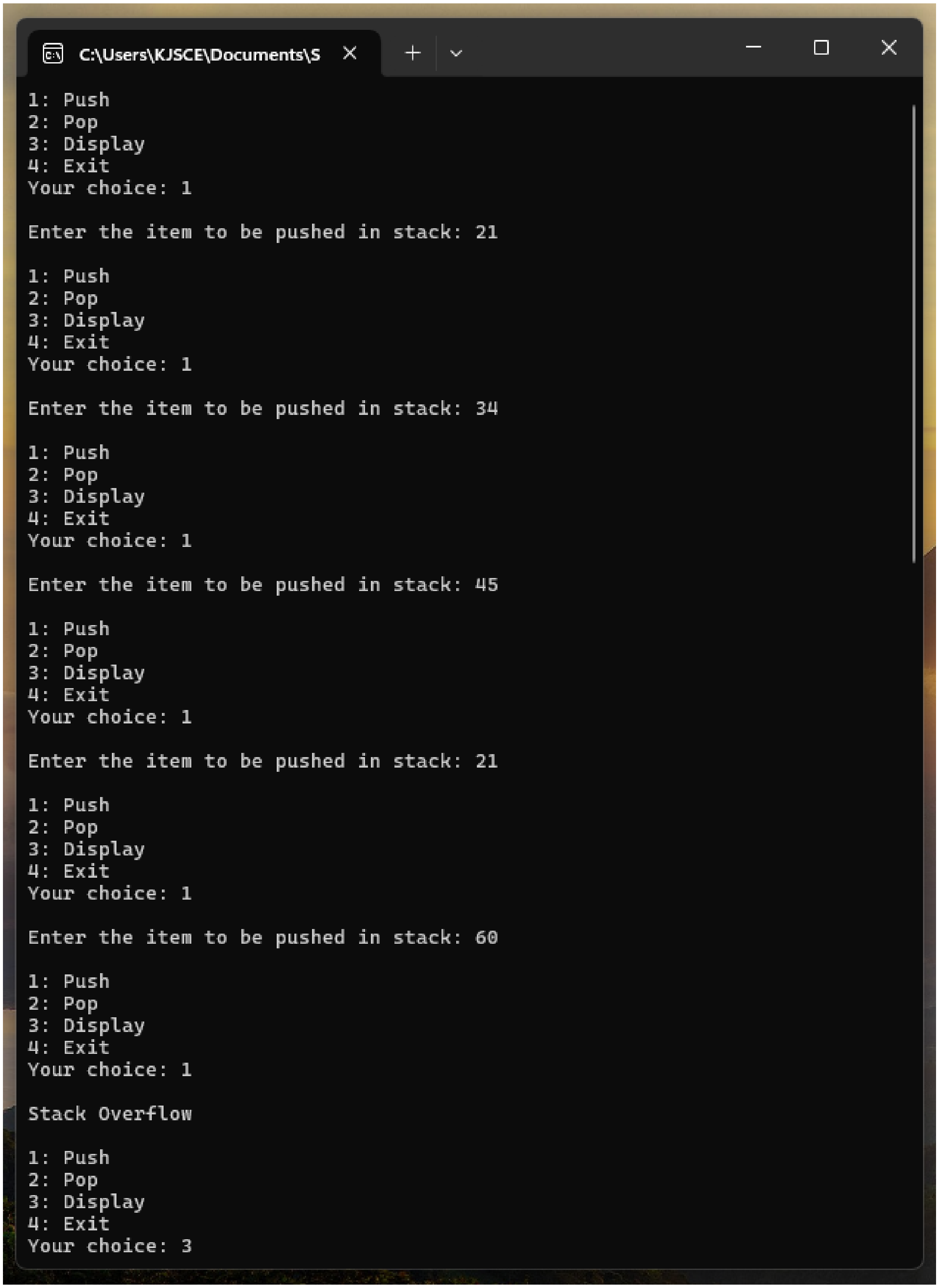
3. Explain the Importance of the approach followed by you

Ans: This structured approach helps in understanding stack behavior and its practical applications like undo features or expression evaluation.

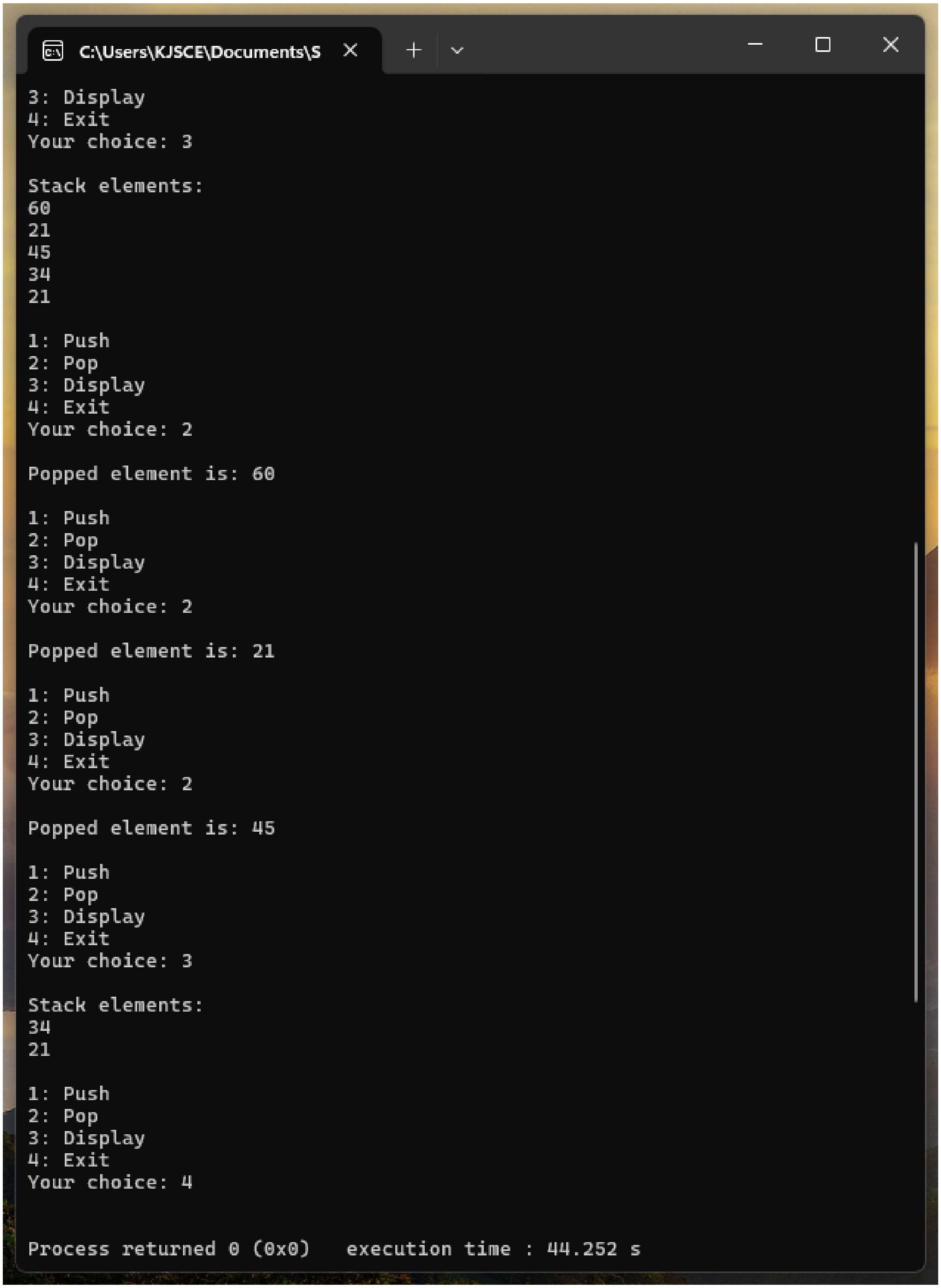
C/C++ Code implemented:

```
#include <stdio.h>

int stack_arr[50], top = -1, MAX = 5;

int main()
{
    int op;
    do
    {
        printf("1: Push\n2: Pop\n3: Display\n4: Exit\n");
        printf("Your choice: ");
        scanf("%d", &op);
        printf("\n");

        switch(op)
        {
            case 1:
                push();
                break;
            case 2:
                pop();
                break;
            case 3:
                display();
                break;
        }
    } while(op != 4);

    return 0;
}

void push()
{
    int pushed_item;
    if(top == (MAX-1))
        printf("Stack Overflow\n");
    else
    {
        printf("Enter the item to be pushed in stack: ");
        scanf("%d", &pushed_item);
        top++;
        stack_arr[top] = pushed_item;
    }
    printf("\n");
}

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

void display()
{
    int i;
    if(top == -1)
        printf("Stack is empty\n");
    else
    {
        printf("Stack elements:\n");
        for(i = top; i >= 0; i--)
        {
            printf("%d\n", stack_arr[i]);
        }
    }
    printf("\n");
}
```

Output/ program results after execution:



```
1: Push
2: Pop
3: Display
4: Exit
Your choice: 1

Enter the item to be pushed in stack: 21

1: Push
2: Pop
3: Display
4: Exit
Your choice: 1

Enter the item to be pushed in stack: 34

1: Push
2: Pop
3: Display
4: Exit
Your choice: 1

Enter the item to be pushed in stack: 45

1: Push
2: Pop
3: Display
4: Exit
Your choice: 1

Enter the item to be pushed in stack: 21

1: Push
2: Pop
3: Display
4: Exit
Your choice: 1

Enter the item to be pushed in stack: 60

1: Push
2: Pop
3: Display
4: Exit
Your choice: 1

Stack Overflow

1: Push
2: Pop
3: Display
4: Exit
Your choice: 3
```



```
3: Display
4: Exit
Your choice: 3

Stack elements:
60
21
45
34
21

1: Push
2: Pop
3: Display
4: Exit
Your choice: 2

Popped element is: 60

1: Push
2: Pop
3: Display
4: Exit
Your choice: 2

Popped element is: 21

1: Push
2: Pop
3: Display
4: Exit
Your choice: 2

Popped element is: 45

1: Push
2: Pop
3: Display
4: Exit
Your choice: 3

Stack elements:
34
21

1: Push
2: Pop
3: Display
4: Exit
Your choice: 4


Process returned 0 (0x0)   execution time : 44.252 s
```

Post Lab Subjective/Objective type Questions:
Write a program to evaluate postfix expressions using stack in C/C++. Ans:

Conclusion:
In this experiment, I learned how stack works using arrays. I implemented push, pop, and display operations, and understood the LIFO principle, highlighted overflow and underflow conditions, and emphasized modular programming through function-based design. It helped me see how stacks are used in real-life.

Signature of faculty in-charge with Date: