Student Information

- **Name:- Aditya Kumar**
- **Sap Id :- 590015145**
- **Branch :- M.C.A**
- **Batch :- B1**
- **Instructor :-Dr. Sourbh Kumar**

Lab Assignment 1: Stack Implementation Using Arrays

```
#include <stdio.h>
#include <stdlib.h>
#define MAX_SIZE 100

int stack[MAX_SIZE];
int top = -1;
int is_empty()
{
   return top == -1;
}

int is_full()
{
   return top == MAX_SIZE - 1;
}
void push(int item)
{
   if (is_full())
 {
       printf("Stack Overflow\n");
       return;
}
   top++;
   stack[top] = item;
   printf("%d pushed to stack\n", item);
}
int pop()
{
   if (is_empty())
{
       printf("Stack Underflow\n");
       return -1;
}
   int item = stack[top];
   top--;
 return item;

}
int peek()
```

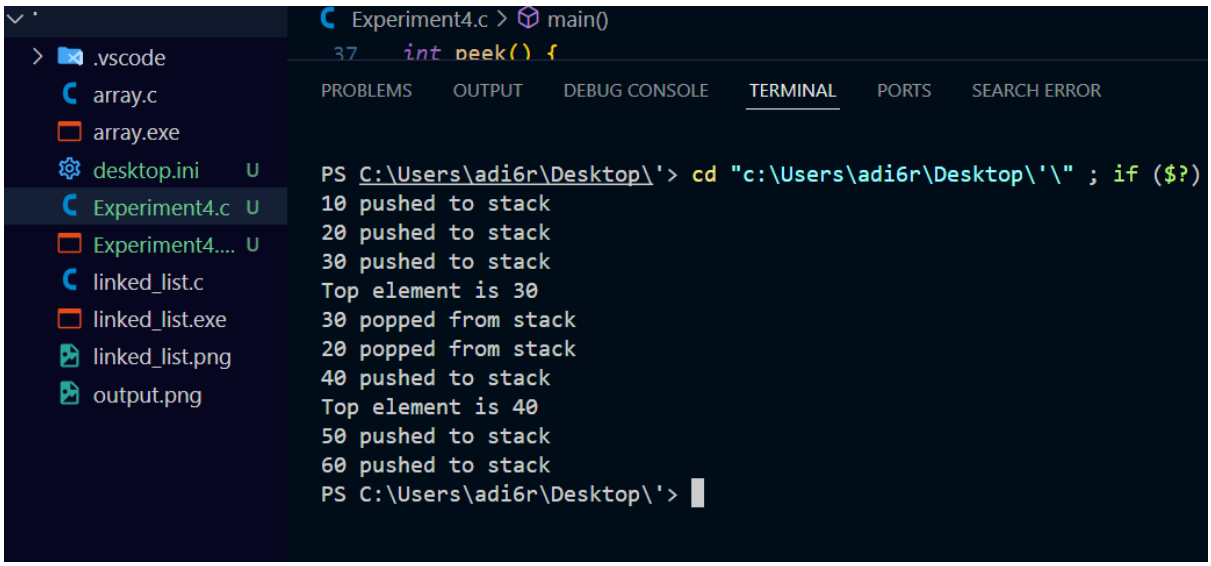
```
{
    if (is_empty())
{
        printf("Stack is empty\n");
        return -1;
    }
    return stack[top];
}
int main()
{
    push(10);
    push(20);
    push(30);

    printf("Top element is %d\n", peek());
    printf("%d popped from stack\n", pop());
    printf("%d popped from stack\n", pop());

    push(40);
    printf("Top element is %d\n", peek());

    // Trying to push more elements than the stack capacity
    push(50);
    push(60);

    return 0;
}
```



Lab Assignment 2: Stack Implementation Using Linked Lists

```
#include <stdio.h>
 #include <stdlib.h>
// Node structure
struct Node
{
    int data;
    struct Node* next;
};
// Stack structure
struct Stack
{
    struct Node* top;
};
// Function to create a new node
struct Node* newNode(int data)
 {
    struct Node* node = (struct Node*)malloc(sizeof(struct Node));
    node->data = data;
    node->next = NULL;
    return node;
}
// Function to check if the stack is empty
int isEmpty(struct Stack* stack)
{
    return stack->top == NULL;
}

// Function to push an item onto the stack
void push(struct Stack* stack, int data)
{
    struct Node* node = newNode(data);
    node->next = stack->top;
    stack->top = node;
    printf("%d pushed to stack\n", data);
}

// Function to pop an item from the stack
int pop(struct Stack* stack)
 {
    if (isEmpty(stack)) {
        printf("Stack Underflow\n");
        return -1;
    }
    struct Node* temp = stack->top;
    int popped = temp->data;
```

```
    stack->top = temp->next;
    free(temp);
    return popped;
}

// Function to peek the top element of the stack
int peek(struct Stack* stack)
{
    if (isEmpty(stack))
 {
        printf("Stack is Empty\n");
        return -1;
    }
    return stack->top->data;
}

int main()
{
    struct Stack* stack = (struct Stack*)malloc(sizeof(struct Stack));
    stack->top = NULL;

    push(stack, 10);
    push(stack, 20);
    push(stack, 30);

    printf("Top element is %d\n", peek(stack));

    printf("%d popped from stack\n", pop(stack));
    printf("%d popped from stack\n", pop(stack));

    push(stack, 40);
    printf("Top element is %d\n", peek(stack));

    return 0;
}
```