Student Information

- **Name**: [ATUL KUMAR]
- **Student ID**: [590011190]
- **Branch**: [MCA]
- **Batch**: [B1]
- **Instructor**: [Dr. Sourbh Kumar]
- **Date**: [28/10/2024]

Data Structure….

Lab Assignment 1: Stack Implementation Using Arrays

Problem Statement: Implement a stack data structure using an array. Your program should support

The following stack operations:

1. Push: Add an element to the top of the stack.
2. Pop: Remove an element from the top of the stack.
3. Peek: Display the top element without removing it.
4. IsEmpty: Check if the stack is empty.
5. IsFull: Check if the stack is full (assume a fixed size).

Assignment Tasks:

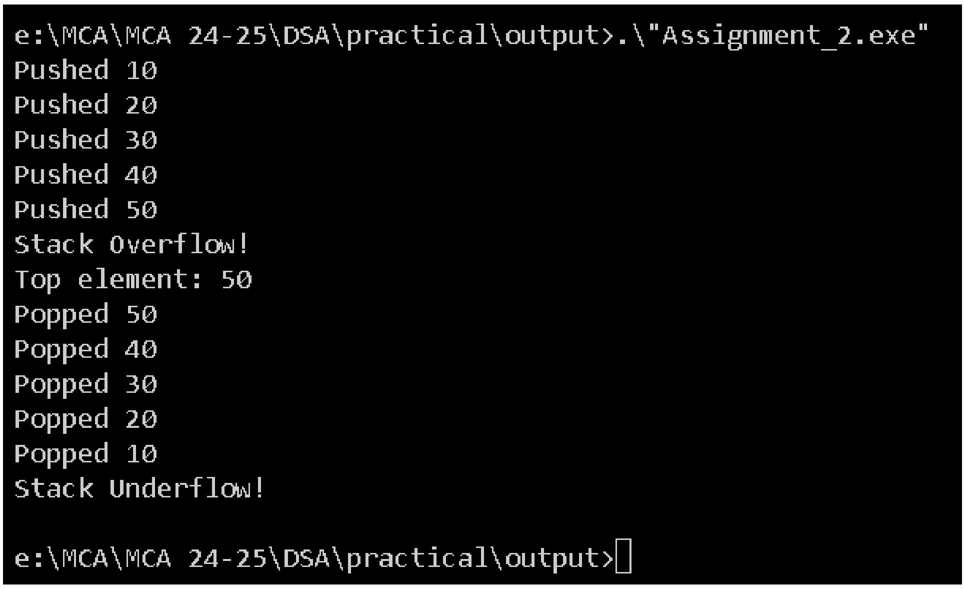
- Write a C program that defines a stack using arrays.
- Implement the stack operations mentioned above.
- Demonstrate stack overflow and underflow conditions.
- Write a main program to test all stack operations.

Solution:-

```
#include <stdio.h>
#define MAX 5 // Maximum size of the stack
// Stack structure definition
struct Stack {
int items[MAX];
int top;
};
// Initialize the stack
void initializeStack(struct Stack *stack) {
stack->top = -1;
}
// Check if the stack is full
int isFull(struct Stack *stack) {
return stack->top == MAX - 1;
}
// Check if the stack is empty
int isEmpty(struct Stack *stack) {
return stack->top == -1;
}
// Push an element onto the stack
void push(struct Stack *stack, int value) {
if (isFull(stack)) {
printf("Stack Overflow!\n");
} else {
stack->items[++stack->top] = value;
printf("Pushed %d\n", value);
}
}
// Pop an element from the stack
int pop(struct Stack *stack) {
if (isEmpty(stack)) {
printf("Stack Underflow!\n");
return -1;
} else {
return stack->items[stack->top--];
}
}

// Peek at the top element of the stack
int peek(struct Stack *stack) {
if (isEmpty(stack)) {
printf("Stack is empty!\n");
return -1;
} else {
return stack->items[stack->top];
}
}
int main() {
struct Stack stack;
initializeStack(&stack);
// Demonstrate stack operations
push(&stack, 10);
push(&stack, 20);
push(&stack, 30);
push(&stack, 40);
push(&stack, 50);
// Attempt to push when stack is full
push(&stack, 60);
// Peek at the top element
printf("Top element: %d\n", peek(&stack));
// Pop elements from the stack
printf("Popped %d\n", pop(&stack));
printf("Popped %d\n", pop(&stack));
printf("Popped %d\n", pop(&stack));
// Attempt to pop when stack is empty
while (!isEmpty(&stack)) {
printf("Popped %d\n", pop(&stack));
}
pop(&stack);  // Underflow demonstration
return 0;
}
```

Output:-



```
e:\MCA\MCA 24-25\DSA\practical\output>.\"Assignment_2.exe"
Pushed 10
Pushed 20
Pushed 30
Pushed 40
Pushed 50
Stack Overflow!
Top element: 50
Popped 50
Popped 40
Popped 30
Popped 20
Popped 10
Stack Underflow!

e:\MCA\MCA 24-25\DSA\practical\output>
```

Lab Assignment 2: Stack Implementation Using Linked Lists

Problem Statement: Implement a stack data structure using a linked list. The program should support the following operations:

1. **Push: Add an element to the top of the stack.**
2. **Pop: Remove an element from the top of the stack.**
3. **Peek: Display the top element without removing it.**
4. **IsEmpty: Check if the stack is empty.**

Assignment Tasks:

- **Write a C program that defines a stack using a singly linked list.**
- **Implement the stack operations mentioned above.**
- **Demonstrate stack operations using linked lists.**
- **Write a main program to test all stack operations.**

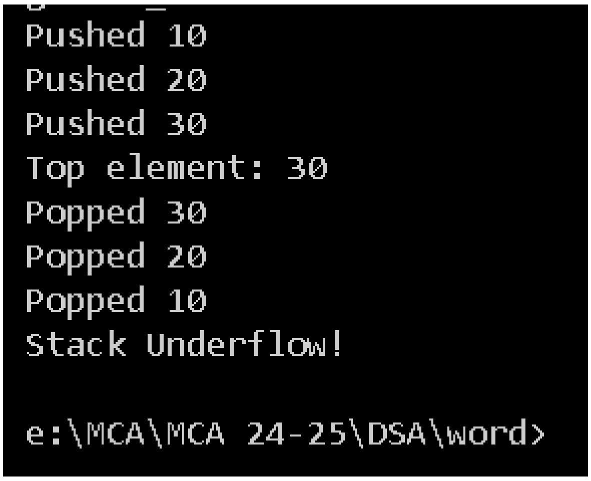
Solution:-

```
#include <stdio.h>
#include <stdlib.h>
// Define a node structure
struct Node {
int data;
struct Node* next;
};
// Check if the stack is empty
int isEmpty(struct Node* top) {
return top == NULL;
}
// Push an element onto the stack
void push(struct Node** top, int value) {
```

```
struct Node* newNode = (struct
Node*)malloc(sizeof(struct Node));
newNode->data = value;
newNode->next = *top;
*top = newNode;
printf("Pushed %d\n", value);
}
// Pop an element from the stack
int pop(struct Node** top) {
if (isEmpty(*top)) {
printf("Stack Underflow!\n");
return -1;
}
struct Node* temp = *top;
int poppedValue = temp->data;
*top = (*top)->next;
free(temp);
return poppedValue;
}
// Peek at the top element of the stack
int peek(struct Node* top) {
if (isEmpty(top)) {
printf("Stack is empty!\n");
return -1;
}
return top->data;
}
int main() {
struct Node* stack = NULL; // Initialize stack
// Perform stack operations
push(&stack, 10);
push(&stack, 20);
push(&stack, 30);
// Peek at the top element
printf("Top element: %d\n", peek(stack));
// Pop elements from the stack
printf("Popped %d\n", pop(&stack));
printf("Popped %d\n", pop(&stack));
printf("Popped %d\n", pop(&stack));
// Attempt to pop from an empty stack
pop(&stack);
```

```
return 0;
}
```

Output:-



```
Pushed 10
Pushed 20
Pushed 30
Top element: 30
Popped 30
Popped 20
Popped 10
Stack Underflow!

e:\MCA\MCA 24-25\DSA\word>
```