}

LINKED LIST IN AN ARRAY

```
#include <stdio.h>

#include <stdlib.h>

#define MAX 100

typedef struct {

    int arr[MAX];

    int top;

} Stack;

void initialize(Stack* stack) {

    stack->top = -1;

}

int isFull(Stack* stack) {

    return stack->top == MAX - 1;

}

int isEmpty(Stack* stack) {

    return stack->top == -1;

}

void push(Stack* stack, int value) {

    if (isFull(stack)) {

        printf("Stack overflow\n");

        return;

    }

    stack->arr[++stack->top] = value;

}
```

```
int pop(Stack* stack) {
    if (isEmpty(stack)) {
        printf("Stack underflow\n");
        return -1;
    }
    return stack->arr[stack->top--];
}
int peek(Stack* stack) {
    if (isEmpty(stack)) {
        printf("Stack is empty\n");
        return -1;
    }
    return stack->arr[stack->top];
}
int main() {
    Stack stack;
    initialize(&stack);

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

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

    printf("Popped element is %d\n", pop(&stack));
    printf("Popped element is %d\n", pop(&stack));

    printf("Popped element is %d\n", pop(&stack));

    printf("Popped element is %d\n", pop(&stack));

    return 0;

}
```

LINKED LIST STACK USING AN ARRAY

```
#include <stdio.h>

#include <stdlib.h>

typedef struct Node {

    int data;

    struct Node* next;

} Node;

typedef struct {

    Node* top;

} Stack;

void initialize(Stack* stack) {

    stack->top = NULL;

}

int isEmpty(Stack* stack) {

    return stack->top == NULL;

}

void push(Stack* stack, int value) {

    Node* newNode = (Node*)malloc(sizeof(Node));

    if (newNode == NULL) {
        printf("Memory allocation failed\n");
        return;
    }
    newNode->data = value;
    newNode->next = stack->top;
    stack->top = newNode;
}
int pop(Stack* stack) {
    if (isEmpty(stack)) {
        printf("Stack underflow\n");
        return -1;
    }
    Node* temp = stack->top;
    int value = temp->data;
    stack->top = temp->next;
    free(temp);
    return value;
}
int peek(Stack* stack) {
    if (isEmpty(stack)) {
        printf("Stack is empty\n");
        return -1;
    }
    return stack->top->data;
}
```

```
void freeStack(Stack* stack) {
    while (!isEmpty(stack)) {
        pop(stack);
    }
}
int main() {
    Stack stack;
    initialize(&stack);

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

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

    printf("Popped element is %d\n", pop(&stack));
    printf("Popped element is %d\n", pop(&stack));
    printf("Popped element is %d\n", pop(&stack));
    printf("Popped element is %d\n", pop(&stack)); // This will indicate underflow
freeStack(&stack);
return 0;
}
```