Experiment 1

CODE:

```c
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>

#define MAX_SIZE 100

struct Stack {
    int items[MAX_SIZE];
    int top;
};

void initialize(struct Stack *s) {
    s->top = -1;
}

void push(struct Stack *s, int value) {
    if (s->top == MAX_SIZE - 1) {
        printf("Stack Overflow\n");
        return;
    }
    s->items[++s->top] = value;
}

int pop(struct Stack *s) {
    if (s->top == -1) {
        printf("Stack Underflow\n");
        exit(1);
    }
    return s->items[s->top--];
}

int peek(struct Stack *s) {
    if (s->top == -1) {
        printf("Stack is empty\n");
        exit(1);
    }
    return s->items[s->top];
}

bool isEmpty(struct Stack *s) {
    return s->top == -1;
}

int main() {
    struct Stack stack;
    initialize(&stack);
    int choice, value;

```

```c
    do {
        printf("\nStack Operations:\n");
        printf("1. Push\n");
        printf("2. Pop\n");
        printf("3. Peek\n");
        printf("4. Check if Empty\n");
        printf("5. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                printf("Enter value to push: ");
                scanf("%d", &value);
                push(&stack, value);
                break;
            case 2:
                printf("Popped element: %d\n", pop(&stack));
                break;
            case 3:
                printf("Top element: %d\n", peek(&stack));
                break;
            case 4:
                if (isEmpty(&stack)) {
                    printf("Stack is empty\n");
                } else {
                    printf("Stack is not empty\n");
                }
                break;
            case 5:
                printf("Exiting...\n");
                break;
            default:
                printf("Invalid choice. Please try again.\n");
                break;
        }
    } while (choice != 5);

    return 0;
}
```

OUTPUT:

```
Stack Operations:
1. Push
2. Pop
3. Peek
4. Check if Empty
5. Exit
Enter your choice: 1
Enter value to push: 85

Stack Operations:
1. Push
2. Pop
3. Peek
4. Check if Empty
5. Exit
Enter your choice: 1
Enter value to push: 86

Stack Operations:
1. Push
2. Pop
3. Peek
4. Check if Empty
5. Exit
Enter your choice: 1
Enter value to push: 87

Stack Operations:
1. Push
2. Pop
3. Peek
4. Check if Empty
5. Exit
Enter your choice: 2
Popped element: 87

Stack Operations:
1. Push
2. Pop
3. Peek
4. Check if Empty
5. Exit
Enter your choice: 3
Top element: 86

Stack Operations:
1. Push
2. Pop
3. Peek
4. Check if Empty
5. Exit
Enter your choice: 4
Stack is not empty

Stack Operations:
1. Push
2. Pop
3. Peek
4. Check if Empty
5. Exit
Enter your choice: 5
Exiting...

=== Code Execution Successful ===
```