

## SEM 3\Exp5\Stack\_ARR.c

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  #define MAX 100 // Maximum size of the stack
5
6  // Stack structure using arrays
7  struct StackArray
8  {
9      int top;
10     int arr[MAX];
11 };
12
13 // Function to create a stack
14 struct StackArray *createStack()
15 {
16     struct StackArray *stack = (struct StackArray *)malloc(sizeof(struct
StackArray));
17     stack->top = -1; // Initialize the top index
18     return stack;
19 }
20
21 // Check if the stack is full
22 int isFull(struct StackArray *stack)
23 {
24     return stack->top == MAX - 1;
25 }
26
27 // Check if the stack is empty
28 int isEmpty(struct StackArray *stack)
29 {
30     return stack->top == -1;
31 }
32
33 // Push an element onto the stack
34 void push(struct StackArray *stack, int value)
35 {
36     if (isFull(stack))
37     {
38         printf("Stack overflow!\n");
39         return;
40     }
41     stack->arr[++stack->top] = value;
42     printf("%d pushed onto stack\n", value);
43 }
44
45 // Pop an element from the stack
46 int pop(struct StackArray *stack)
47 {
48     if (isEmpty(stack))
49     {
50         printf("Stack underflow!\n");
51         return -1;
```

```
52     }
53     return stack->arr[stack->top--];
54 }
55
56 // Peek at the top element of the stack
57 int peek(struct StackArray *stack)
58 {
59     if (isEmpty(stack))
60     {
61         printf("Stack is empty!\n");
62         return -1;
63     }
64     return stack->arr[stack->top];
65 }
66
67 // Display the stack
68 void display(struct StackArray *stack)
69 {
70     if (isEmpty(stack))
71     {
72         printf("Stack is empty!\n");
73         return;
74     }
75     printf("Stack elements: ");
76     for (int i = stack->top; i >= 0; i--)
77     {
78         printf("%d ", stack->arr[i]);
79     }
80     printf("\n");
81 }
82
83 int main()
84 {
85     struct StackArray *stack = createStack();
86     int choice, value;
87
88     printf("\nStack Operations (Array Implementation):\n");
89     printf("1. Push\n");
90     printf("2. Pop\n");
91     printf("3. Peek\n");
92     printf("4. Display\n");
93     printf("5. Exit\n");
94
95     while (1)
96     {
97
98         printf("Enter your choice: ");
99         scanf("%d", &choice);
100
101         switch (choice)
102         {
103             case 1:
104                 printf("Enter the value to push: ");
105                 scanf("%d", &value);
```

```
106         push(stack, value);
107         printf("\n");
108         break;
109     case 2:
110         value = pop(stack);
111         if (value != -1)
112             printf("Popped value: %d\n", value);
113
114         printf("\n");
115         break;
116     case 3:
117         value = peek(stack);
118         if (value != -1)
119             printf("Top value: %d\n", value);
120
121         printf("\n");
122         break;
123     case 4:
124         display(stack);
125         printf("\n");
126         break;
127     case 5:
128         free(stack);
129         exit(0);
130     default:
131         printf("Invalid choice!\n");
132     }
133 }
134
135 return 0;
136 }
137
```