S.No: 3	Exp. Name: ***Stack using Arrays***	Date:2025-08-13

ID: 24CS079

2024-2028-CSE-B

KPR Institute of Engineering and Technology

Aim:

Write a C program to implement stack operations- push, pop, peek, display, isEmpty using arrays.

Source Code:

StackUsingArray.c

```
#include <stdio.h>
#include <stdlib.h>
#define STACK_MAX_SIZE 10
#include "StackOperations.c"

int main() {
   int op, x;
   while(1) {
      printf("1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit\n");
      printf("Option: ");
      scanf("%d", &op);
      switch(op) {
         case 1:
            printf("element: ");
            scanf("%d", &x);
            push(x);
            break;
         case 2:
            pop();
            break;
         case 3:
            display();
            break;
         case 4:
            isEmpty();
            break;
         case 5:
            peek();
            break;
         case 6:
            exit(0);
      }
   }
}
```

StackOperations.c

```
// declare the size of the array
int stack[STACK_MAX_SIZE];
int top = -1;
int i;
// define the top to -1

void push(int element) {
	if(top==STACK_MAX_SIZE - 1){
		printf("Stack is overflow\n");
	}else{
		stack[++top] = element;
		printf("Successfully pushed\n");
	}

}
void display() {
	if(top==-1){
		printf("Stack is empty\n");
	}else{
		printf("Elements: ");
		for(i = top; i>=0;i--){
			printf("%d ", stack[i]);
		}
		printf("\n");
	}

}
void pop() {
	if(top==-1){
		printf("Stack is underflow\n");
	}else{
		printf("Popped value: %d\n", stack[top--]);
	}

}
void peek(){
	if(top==-1){
		printf("Stack is underflow\n");
	}else{
		printf("Peek value: %d\n", stack[top]);
	}

}
void isEmpty() {
	if(top==-1){
		printf("Stack is empty\n");
	}
	else{
		printf("Stack is not empty\n");
	}

}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
`1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit` 4
`Option:  4`
`Stack is empty` 2
`1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit` 2

```
Option:  2
Stack is underflow 3
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 3
Option:  3
Stack is empty 5
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 5
Option:  5
Stack is underflow 1
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 1
Option:  1
element:  25
Successfully pushed 1
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 1
Option:  1
element:  26
Successfully pushed 3
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 3
Option:  3
Elements: 26 25  2
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 2
Option:  2
Popped value: 26 4
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 4
Option:  4
Stack is not empty 5
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 5
Option:  5
Peek value: 25 6
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 6
Option:  6
```

Test Case - 2

User Output

```
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 1
Option:  1
element:  1
Successfully pushed 1
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 1
Option:  1
element:  2
Successfully pushed 1
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 1
Option:  1
element:  3
Successfully pushed 1
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 1
Option:  1
element:  4
Successfully pushed 1
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 1
```

```
Option:  1
element:  5
Successfully pushed 1
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 1
Option:  1
element:  6
Successfully pushed 1
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 1
Option:  1
element:  7
Successfully pushed 1
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 1
Option:  1
element:  8
Successfully pushed 1
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 1
Option:  1
element:  9
Successfully pushed 1
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 1
Option:  1
element:  10
Successfully pushed 1
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 1
Option:  1
element:  11
Stack is overflow 6
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit 6
Option:  6
```