Aim:

To implement a stack using an array and perform basic operations:

- PUSH → Insert an element
- POP → Delete top element
- PEEK → Display top element

Algorithm:

1. Start
2. Define an array stack[MAX] and variable top = -1.
3. PUSH:
    - If top == MAX-1 → Overflow
    - Else → stack[++top] = value
4. POP:
    - If top == -1 → Underflow
    - Else → top--
5. PEEK:
    - If top == -1 → Empty
    - Else → Print stack[top]

CODE:

```
#include <stdio.h>
#define MAX 10

int stack[MAX], top = -1;
```

```
void push(int val) {
   if (top == MAX - 1)
      printf("Stack Overflow!\n");
   else
      stack[++top] = val;
}

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

void peek() {
   if (top == -1)
      printf("Stack is empty!\n");
   else
      printf("Top element: %d\n", stack[top]);
}

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

int main() {
   int choice, val;
   while (1) {
      printf("\n1.PUSH  2.POP  3.PEEK  4.DISPLAY  5.EXIT\n");
      printf("Enter choice: ");
      scanf("%d", &choice);

      if (choice == 1) {
         printf("Enter value: ");
         scanf("%d", &val);
         push(val);
      }
```

```
        else if (choice == 2)
            pop();
        else if (choice == 3)
            peek();
        else if (choice == 4)
            display();
        else if (choice == 5)
            break;
        else
            printf("Invalid choice!\n");
    }
    return 0;
}
```

Output

```
1.PUSH  2.POP  3.PEEK  4.DISPLAY  5.EXIT
Enter choice: 1
Enter value: 10

1.PUSH  2.POP  3.PEEK  4.DISPLAY  5.EXIT
Enter choice: 2
Popped: 10

1.PUSH  2.POP  3.PEEK  4.DISPLAY  5.EXIT
Enter choice: 3
Stack is empty!
```

RESULT:

The program successfully executed and displayed the operations of stack like push(),pop(),peek().