Experiment 1.

Aim: Implement stack ADT using array.

Theory:

Stack ADT: A list with the restriction that insertion and deletion can be performed only from one end. ~~called~~ according to the Last in First Out (LIFO) principle.

1. PUSH: The push operation is used to insert an element into the stack at the topmost position of stack.

```
step1: If (TOP=MAX-1) {
           print "Overflow"     → {If this is the case, then stack
           Goto step 4 }           is full & no more insertions
step2: Set TOP=TOP+1.              can be done}
step3: Set STACK[TOP]=VALUE
step4: END.
```

2. POP: The pop operation is used to delete the topmost element from the stack.

```
step1: If TOP=NULL {             → {If this is the case, it means
           Print "UNDERFLOW"        the stack is empty & no
           Goto step 4 }            more deletions can be done}
step2: Set VAL = STACK[TOP]
step3: Set TOP = TOP-1.
step4: END.
```

3. PEEK : The peek operation returns the value of the topmost element of the stack without deleting it from the stack

```
Step 1 : If TOP = NULL                    {checks if the stack
             print "STACK IS EMPTY"        is empty}
             Goto step 3
Step 2 : Return STACK[TOP]
Step 3 : END.
```

4. isEmpty(): The isEmpty() operation is to test for whether or not stack is Empty.

```
Step 1 : If TOP < 1
             return true.
         else
             return false
Step 2 : END.
```

5. isFull(): The isFull() operation is used to check is the stack is full or not.

```
Step 1 : If TOP = MAXSIZE.
             return true
         else
             return false.
Step 2 : END.
```

6. Display : The Display operation is used to display any element from the stack

```
1. Display (TOP, i, a[i])
2. If TOP=0 then
      Print "STACK EMPTY"
      exit
3. ELSE.
        For i=TOP to 0
        print a[i]
4. END.
```

→ If this the case, it means the stack is empty.

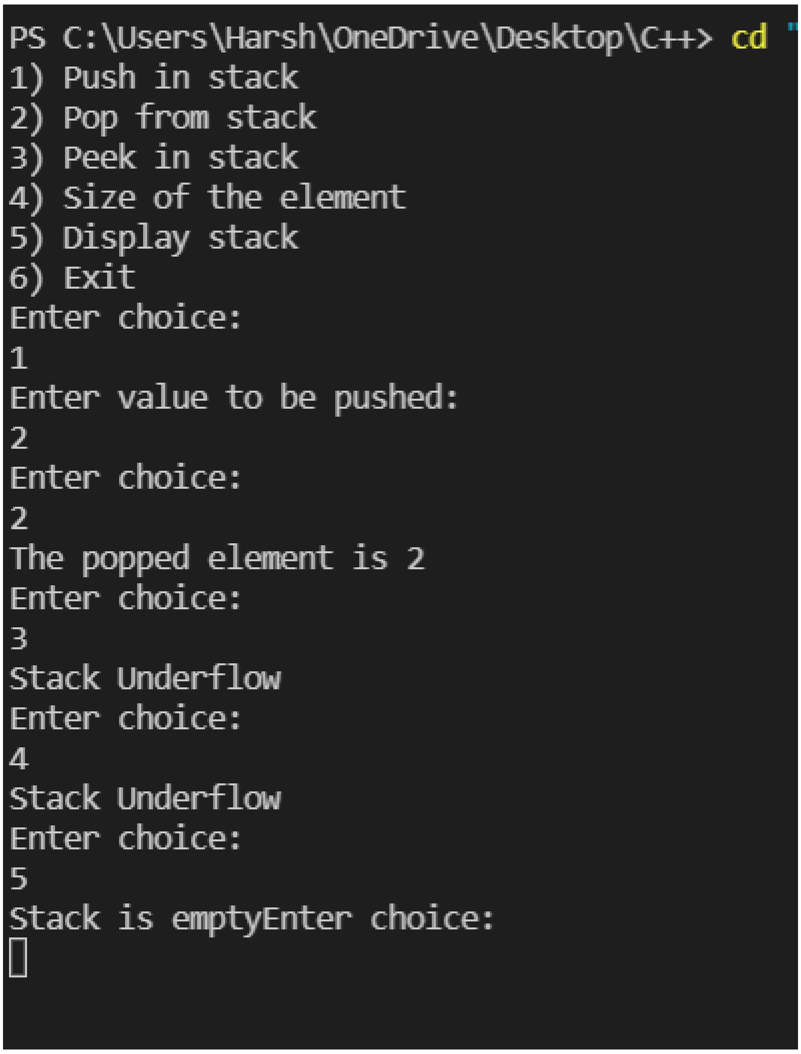
Conclusion : Stack ADT allows all data operations at one end only. At any time, we can only access the top element of a stack.

Program:

```
#include <iostream>
using namespace std;
int stack[10], n=10, top=-1;
void push(int val) {
   if(top>=n-1){
       cout<<"Stack Overflow"<<endl;
   }
   else {
      top++;
      stack[top]=val;
   }}
void pop() {
   if(top<=-1){
       cout<<"Stack Underflow"<<endl;
   }
   else {
      cout<<"The popped element is "<< stack[top] <<endl;
      top--;
   }}
void peek(){
    if(top<=-1){
        cout<<"Stack Underflow"<<endl;
    }
    else {
        cout<<"The top most element is : "<<stack[top]<<endl;
    }}
void size(){
    int count = 0;
     if(top<=-1){
        cout<<"Stack Underflow"<<endl;
    }
    else {
        for (int i= top; i>=0; i--){
            count++;
        }
        cout<<"Size of the stack is:"<<count<<endl;
    }}

void display() {
   if(top>=0) {
      cout<<"Stack elements are:";
      for(int i=top; i>=0; i--)
      cout<<stack[i]<<" ";
      cout<<endl;
   } else
   cout<<"Stack is empty";
}
int main() {
   int ch, val;
   cout<<"1) Push in stack"<<endl;
   cout<<"2) Pop from stack"<<endl;
   cout<<"3) Peek in stack"<<endl;
   cout<<"4) Size of the element"<<endl;
   cout<<"5) Display stack"<<endl;
   cout<<"6) Exit"<<endl;
   do {
      cout<<"Enter choice: "<<endl;
      cin>>ch;
      switch(ch) {
          case 1: {
            cout<<"Enter value to be pushed:"<<endl;
            cin>>val;
            push(val);
            break;}
          case 2: {
            pop();
            break; }
          case 3: {
             peek();
             break;}
          case 4: {
             size();
             break;}
          case 5: {
             display();
             break;}
          case 6: {
            cout<<"Exit"<<endl;
            break; }
          default: {
            cout<<"Invalid Choice"<<endl;
          }
      }}
 while(ch!=6);
   return 0;
}
```

Output:



```
PS C:\Users\Harsh\OneDrive\Desktop\C++> cd "
1) Push in stack
2) Pop from stack
3) Peek in stack
4) Size of the element
5) Display stack
6) Exit
Enter choice:
1
Enter value to be pushed:
2
Enter choice:
2
The popped element is 2
Enter choice:
3
Stack Underflow
Enter choice:
4
Stack Underflow
Enter choice:
5
Stack is emptyEnter choice:
```