|  |  |
| --- | --- |
| yss | SVKM’s NMIMS  School of Technology Management & Engineering Navi Mumbai Campus |
| Department of Computer Engineering |

|  |  |
| --- | --- |
| Name: Yash Ravindra Patil | SAP ID: 70472200140 |
| Semester: III | Year: II |
| Subject: Data Structure and Algorithm | Roll No.: A176 |
| Practical: 2 | Date: 4/7/2023 |
| Batch: 1 |  |

**Aim:–**

Implementation of various array operations like traversal, insertion and deletion using any real-life application.

**Theory:–**

A stack is a linear data structure in which the insertion of a new element and removal of an existing element takes place at the same end represented as the top of the stack.

Basic Operations on Stack

In order to make manipulations in a stack, there are certain operations provided to us.

**push() to** insert an element into the stack

**pop()** to remove an element from the stack

**top()** Returns the top element of the stack.

**isEmpty()** returns true if stack is empty else false.

**size()** returns the size of stack.

The process in which a function calls itself directly or indirectly is called recursion and the corresponding function is called a recursive function. Using a recursive algorithm, certain problems can be solved quite easily.

A recursive function solves a particular problem by calling a copy of itself and solving smaller subproblems of the original problems. Many more recursive calls can be generated as and when required. It is essential to know that we should provide a certain case in order to terminate this recursion process. So we can say that every time the function calls itself with a simpler version of the original problem.

**Code/Implementation –**

// Online C++ compiler to run C++ program online

#include <iostream>

using namespace std;

int stack[100], n=100, 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 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) Display stack"<<endl;

cout<<"4) 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: {

display();

break;

}

case 4: {

cout<<"Exit"<<endl;

break;

}

default: {

cout<<"Invalid Choice"<<endl;

}

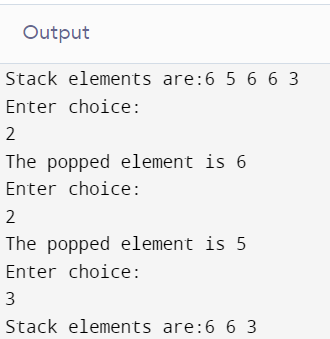
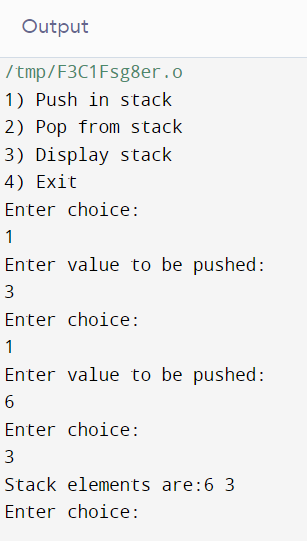
}

}while(ch!=4);

return 0;

}

**Output:-**

****

**Conclusion:-**

From the given experiment we learned about the implementation of concept push pop and display function of Stack data structure and use in recursion.

**Outcome: -**

Identified and applied appropriate linear data structure for the given problem