Experiment 6
Implementation Of Stack Using Array

Date: 21-09-2020

Aim: To implement Stack data structure using array

Data Structure Used: Arrays, Stack

Operation Used: Comparisons

Algorithm:

Algorithm for isEmpty()

Input: Stack A and pointer to the top most element, Top
Output: True if stack is empty, false if stack is not empty

Step 1 : Start
Step 2 : If top<0
    Step 1 : return true
Step 3: else
    Step 1: return false
Step 4 : Stop

Description of the Algorithm:

Returns a true value it the stack is empty false if otherwise

Algorithm for isFull()

Input: Stack A and pointer to the top most element, Top
Output: True if stack is full, false if stack is not full

Step 1 : Start
Step 2 : If top>=size //size is the predefined size of the array A
    Step 1 : return false
Step 3: else
    Step 1: return true
Step 4 : Stop

Description of the Algorithm:

Returns a true value it the stack is full false if otherwise

Algorithm for push function:

Input: Stack A and pointer to the top most element, Top and element to be inserted e
Output: Stack with the element e added on the top

Step 1 : Start
Step 2 : If isFull()
    Step 1 : Print “Overflow, The stack is Full”
Step 3: else
    Step 1: A[++top] = e
Step 4 : Stop

Description of the Algorithm:

Takes an input e and adds it to the top of the stack if it is not full

Algorithm for pop function:

Input: Stack A and pointer to the top most element, Top
Output: Stack with the top element removed

Step 1 : Start
Step 2 : If isEmpty()
    Step 1 : Print "Underflow, There is no element in the stack"
Step 3: else
    Step 1: top--
Step 4 : Stop

Description of the Algorithm:

Removes the top element of the array by decrementing it

Algorithm for seek function:

Input: Stack A and pointer to the top most element, Top and the index of the element from the top
Output: Element e at position index from the top

Step 1 : Start
Step 2 : i = top-index+1
Step 3 : If i<0
    Step 1 : Print "There is no element at position index from top"
    Step 2: return 0
Step 4: else
    Step 1: return A[i]
Step 5 : Stop

Description of the Algorithm:

Returns the element a position index from the top. That is if index is 1 then it will return top. If the value of top is less than index-1 then error is shown.

Algorithm for seekTop function:

Input: Stack A and pointer to the top most element, Top.
Output: Element at the top of the stack

Step 1 : Start
Step 2 : If isEmpty()
    Step 1 : Print "Underflow There is no element in the array"
    Step 2: return 0
Step 3: else
    Step 1: return A[top]
Step 4 : Stop

Description of the Algorithm:

Element at the top of the array is returned

Result: the Program is successfully compiled and the desired output is obtained.

Program/ Source Code:

```
#include<stdio.h>
#include<stdlib.h>

#define SIZE 50


int A[SIZE];
int top = -1;

int  isEmpty(){
    if(top<0){
        return 1;
    }
    else{
        return 0;
    }
}

int isFull(){
    if(top<SIZE)
        return 0;
    else
        return 1;
}

int peek(int index){
    int i = top-index +1;
    if(i<0){
        printf("Underflow there is no element in the array \n");
        return 0;
    }
    else{
        return A[i];
    }
}

int stackTop(){
    if(isEmpty()){
        printf("The Stack is empty no element in stack\n");
        return 0;
    }
    else{
        return A[top];
    }
}

void push(int a){
    if(isFull()){
        printf("Stack is Full: Overflow");
    }
    else{
        top = top+1;
        A[top] = a;
```

```
    }
}

int pop(){
    int a;
    if(isEmpty()){
        printf("Underflow Stack is empty no element to pop");
    }
    else{
        a =  A[top];
        top--;
    }
    return a; //garbage or error is returned if underflow occurs
}

void main(){
    int c;
    int i;
    int e;
    int RUN = 1;
    while(RUN){
        printf("\n");
        printf("=========================\n");
        printf("Menu\n");
        printf("1.push\n2.pop\n3.Check if empty\n4.Check if full\n5.Element at top\
n6.peek\n7.Exit\n");
        printf("=========================\n");
        printf("\nEnter Choice ---> ");
        scanf("%d%*c",&c);
        switch(c){
            case 1: printf("\nEnter an element to push into the array --> ");
                    scanf("%d%*c",&e);
                    push(e);
                    break;
            case 2: e =pop() ;
                    printf("\nElement poped is %d\n",e);
                    break;
            case 3: if(isEmpty()){
                        printf("Stack is empty\n");
                    }
                    else{
                        printf("Stack is not empty\n");
                    }
                    break;
            case 4: if(isFull()){
                        printf("Stack is full\n");
                    }
                    else{
                        printf("Stack is not full\n");
                    }
                    break;
            case 5: e = stackTop();
                    printf("The Element at top is %d\n", e);
                    break;
            case 6: printf("Enter the value of the index--> ");
                    scanf("%d%*c",&i);
```

```
                    e = peek(i);
                    printf("\nThe %dth element in the stack is %d\n",i,e);
                    break;
            case 7: RUN = 0;
                    printf("\nExiting!!!!!!!!!!\n");
                    break;
            default: printf("Enter a proper value!!!!!!!!!!!!!! \n");
        }
    }
}
```

Sample Input/Output

Sample input:

```
1
32
1
-41
1
12
2
5
6
2
2
2
7
```

Sample Output:

```
=========================
Menu
1.push
2.pop
3.Check if empty
4.Check if full
5.Element at top
6.peek
7.Exit
=========================

Enter Choice ---> 1

Enter an element to push into the array --> 32

=========================
Menu
1.push
2.pop
3.Check if empty
4.Check if full
5.Element at top
6.peek
7.Exit
=========================
```

```
Enter Choice ---> 1

Enter an element to push into the array --> -41

=========================
Menu
1.push
2.pop
3.Check if empty
4.Check if full
5.Element at top
6.peek
7.Exit
=========================

Enter Choice ---> 1

Enter an element to push into the array --> 12

=========================
Menu
1.push
2.pop
3.Check if empty
4.Check if full
5.Element at top
6.peek
7.Exit
=========================

Enter Choice ---> 2

Element poped is 12

=========================
Menu
1.push
2.pop
3.Check if empty
4.Check if full
5.Element at top
6.peek
7.Exit
=========================

Enter Choice ---> 5
The Element at top is -41

=========================
Menu
1.push
2.pop
3.Check if empty
4.Check if full
5.Element at top
6.peek
```

```
7.Exit
=========================

Enter Choice ---> 6
Enter the value of the index--> 2

The 2th element in the stack is 32

=========================
Menu
1.push
2.pop
3.Check if empty
4.Check if full
5.Element at top
6.peek
7.Exit
=========================

Enter Choice ---> 2

Element poped is -41

=========================
Menu
1.push
2.pop
3.Check if empty
4.Check if full
5.Element at top
6.peek
7.Exit
=========================

Enter Choice ---> 2

Element poped is 32

=========================
Menu
1.push
2.pop
3.Check if empty
4.Check if full
5.Element at top
6.peek
7.Exit
=========================

Enter Choice ---> 7

Exiting!!!!!!!!!!
```