PROGRAM 3

Design, Develop and Implement a menu driven Program in C for the following operations on STACK of Integers (Array Implementation of Stackwith maximum size MAX)

a.Push an Element ontoStackb
b.Pop an Element fromStack
c.Demonstrate how Stack can be used to check Palindrome
d.Demonstrate Overflow and Underflow situations on Stack
e. Display the status of Stack
f. Exit
Support the program with appropriate functions for each of the above operations.

<u>Program objective:</u>

- Understand the concept of palindrome.
- Understand the stack data structures.
- Understand the different functions onstacks i.e., push, pop and implement the same.
- Understand stack overflow and underflow.

Algorithm:

```
PUSH (item)
Step 1: Read an element to be pushed on to stack item
Step 2: check overflow condition of stack before inserting element into
stack Top=max-1
Step 3: update the top pointer and insert an element into stack
Top=top+1
S[top] <-item

POP (item)
Step1: check underflow condition of stack before deleting element from stack
 top=-1
Step2: Display deleted element pointed by top
Deleted element<- s[top]
Step3: Decrement top pointer by 1
top<-top-1
```

Palindrome
Step 1: Two pointers are required , one is pointed to top of stack
another is bottom of stack
Step 2: compare top and bottom elements of stack if it is equal update top and
bottom pointer by1
Step 3: if all elements are equal, then stack content is palindrome

THEORY

It is called as last in, first out. The element inserted first is the last one to be deleted. It is used for various applications like infix to postfix expression, postfix evaluation and for maintaining stack frames for function calling

A real-world stack allows operations at one end only. For example, we can place or remove a card or plate from top of the stack only. Likewise, Stack ADT allows all data operations at one end only.

At any given time, we can only access the top element of a stack. This feature makes it LIFO data structure. LIFO stands for Last-in-first-out. Here, the element which is placed (inserted or added) last is accessed first. In stack terminology, insertion operation is called **PUSH** operation and removal operation is called **POP** operation.

Below given diagram tries to depict a stack and its operations −

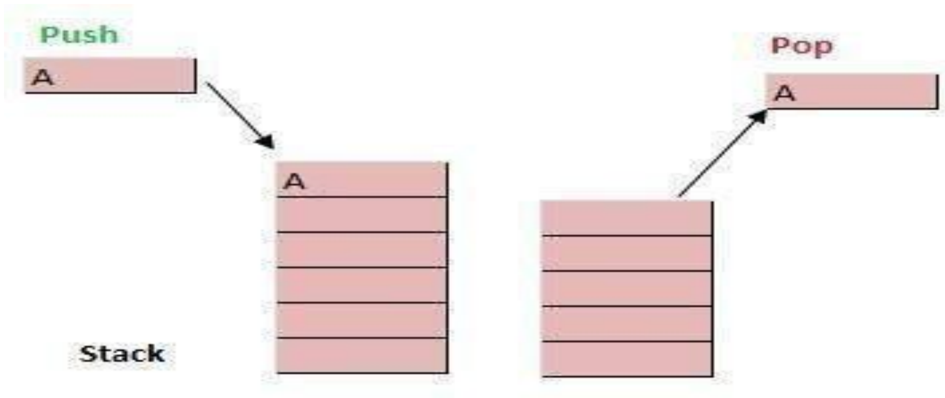

Fig4-Example of Stack

A stack can be implemented by means of Array, Structure, Pointer and Linked-List. Stack can either be a fixed size one or it may have a sense of dynamic resizing.

Here, we are going to implement stack using arrays which makes it a fixed size stack implementation.

Basic Operations performed on stack:

- **push()** - pushing (storing) an element on the stack.
- **pop()** - removing (accessing) an element from the stack.

To use a stack efficiently we need to check status of stack as well. For the same purpose, the following functionality is added to stacks;

- **peek()** − get the top data element of the stack, without removing it.
- **isFull()** − check if stack is full.
- **isEmpty()** − check if stack is empty.

PROGRAM:

```
#include<stdio.h>
#include<stdlib.h>
#define MAX 4
int stack[MAX],top=-1,item;
void push();
void pop();
void palindrome();
void display();
void main()
{
int choice;
while(1)
{
Printf(“------- STACK OPERATIONS -------\n“);
printf("1.push\n 2.pop\n 3.palindrome\n 4.display\n 5.exit\n");
printf("enter choice");
scanf("%d",&choice);
switch(choice)
{
case 1:push();
       break;
case 2:pop();
       break;
case 3:palindrome();
       break;
case 4:display();
       break;
case 5:exit(0);
       break;
default:printf("invalid choice\n");
        break;
}
}
}
```

```
void push()
{
if(top==MAX-1)
printf("stack overflow");
else
{
printf("enter the item to be pushed\n");
scanf("%d",&item);
top=top+1;
stack[top]=item;
}
}
void pop()
{
if(top==-1)
printf("stack underflow");
else
{
item=stack[top];
top=top-1;
printf("deleted item is %d",item);
}
}
void display()
{
int i;
if(top==-1)
printf("stack is empty");
else
{
for(i=top;i>=0;i--)
printf("%d\t",stack[i]);
}
}
```

```
void palindrome()
{
int num[10],i=0,k,flag=1;
k=top;
while(k!=-1)
num[i++]=stack[k--];
for(i=0;i<=top;i++)
{
if(num[i]==stack[i])
continue;
else
flag=0;
}
if(top==-1)
printf("stack is empty");
else
{
if(flag)
printf("palindrome");
else
printf("not a palindrome");
}
}
```

Output

```
------- STACK OPERATIONS ------
1.  Push
2.  Pop
3.  Palindrome
4.  Display
5.  Exit
Enter your choice 1
enter element to be inserted
10
------- STACK OPERATIONS------
1.  Push
2.  Pop
3.  Palindrome
4.  Display
5.  Exit
Enter your choice 1
enter element to be inserted
20
------- STACK OPERATIONS------
1.  Push
2.  Pop
3.  Palindrome
4.  Display
5.  Exit
```

```
Enter your choice 1
enter element to be inserted
30
------- STACK OPERATIONS------
1. Push
2. Pop
3. Palindrome
4. Display
5. Exit
Enter your choice 1
enter element to be inserted
40
------- STACK OPERATIONS------
1. Push
2. Pop
3. Palindrome
4. Display
5. Exit
Enter your choice 1
enter element to be inserted
50
------- STACK OPERATIONS------
1. Push
2. Pop
3. Palindrome
4. Display
5. Exit
Enter your choice 1
Stack Overflow:
------- STACK OPERATIONS------
1. Push
2. Pop
3. Palindrome
4. Display
5. Exit
Enter your choice 4
```

```
stack elements are:
50      40      30      20      10

------- STACK OPERATIONS------
1.Push
2.Pop
3.  Palindrome
4.  Display
5.  Exit
Enter your choice 2
The poped element: 50
------- STACK OPERATIONS------
1.Push
2.Pop
3.  Palindrome
4.  Display
5.  Exit
Enter your choice 2
The poped element: 40
------- STACK OPERATIONS------
1.Push
2.Pop
3.  Palindrome
4.  Display
5.  Exit
Enter the choice
2
The poped element: 30
------- STACK OPERATIONS------
1.  Push
2.  Pop
3.  Palindrome
4.  Display
5.  Exit
Enter your choice 2
The poped element: 20
```

```
------- STACK OPERATIONS------
Push
1. Pop
2. Palindrome
3. Display
4. Exit
Enter the choice
2
The poped element: 10
------- STACK OPERATIONS------
1. Push
2. Pop
3. Palindrome
4. Display
5. Exit
The enter the choice        2
Stack is Empty
```

Program outcome :

- Analyze the stack overflow and underflow conditions.
- Identify different application ofstacks.
- Implement to check palindrome numbers using stacks.
- Familiarized with push and pop operations on stack.

Viva Questions:

- What is Stack and where it can be used?
- What is the difference between PUSH and POP?
- Differentiate STACK from ARRAY.
- What is the difference between a stack and a Queue?