G. H. RAISONI COLLEGE OF ENGG., NAGPUR
(An Autonomous Institute under UGC Act 1956)
Department of Artificial Intelligence

Date: 20/07/2020

Practical Subject: Data Structures and Algorithms
Session: 2020-21

Student Details:

Roll Number	49
Name	Shivam Tawari
Semester	3
Section	A
Branch	Artificial Intelligence

Practical Details: Practical Number- 2

Practical Aim	Design, Develop and Implement a menu driven Program in C for the following operations on STACK of Integers (Array Implementation of Stack with maximum size MAX) a) Push an Element on to Stack b) Pop an Element from Stack 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
Theory	***Theory:*** A stack is a data structure that holds a list of elements. A stack works based on the LIFO principle i.e., Last In, First out, meaning that the most recently added element is the first one to remove. A stack has two main operations that occur only at the top of the stack: push and pop. The push operation places an element at the top of stack whereas the pop operation removes an element from the top of the stack. push () method:

	The push () method allows you to add one or more elements to the end of the array. The push () method returns the value of the length property that specifies the number of elements in the array. pop () method: The pop () method removes the element at the end of the array and returns the element to the caller. If the array is empty, the pop () method returns undefined. display (): Display function used to print the values at any time. Palindrome number: A palindrome number is a number that is same after reverse. Stack overflow: A stack overflow is when you've used up more memory for the stack than your program was supposed to use Stack underflow: An error condition that occurs when an item is called for from the stack, but the stack is empty.
Procedure	1. User selects the operation; it wants to perform on the stack. 2. We have Push, Pop and Display functions and also a program to check palindrome in the menu. 3. If the stack gets full while inserting the integer, it gives a message as 'Stack Overflow!!'. 4. If the stack is empty then it gives a message as 'Stack Underflow!!'. 5. Selecting the Palindrome option in the menu, checks if the given input is palindrome or not and displays the appropriate message. 6. We can then display the integers in the stack by selecting Display. 7. And we can exit the program by selecting the last option.
Algorithm	**Step 1:** START **Step 2:** Initialize stack size MAX and top of stack -1.

	Step 3: Push integer element on to stack and display the contents of the stack. If stack is full give a message as 'Stack Overflow!!'. Pop element from stack along with display the stack contents. If stack is empty give a message as 'Stack Underflow!!'. **Step 4:** Check whether the stack contents are Palindrome or not. **Step 5:** Two pointer is required; one is pointed to top of stack another is bottom of stack. **Step 6:** Compare top and bottom elements of stack if it is equal update top and bottom pointer by 1. **Step 7:** If all elements are equal, then stack content is palindrome. **Step 8:** Display the stack, if required or display exit. **Step 9:** STOP

Program	main.c Run
	1 #include <stdio.h>
	2 #include <string.h>
	3 #include <stdlib.h>
	4 #define MAX 5 // maximum 5 elements can be inserted
	5
	6 void push(int stack[MAX], int *top, int ele) {
	7 // function to insert element into Stack
	8 if (*top >= MAX) {
	9 printf("\n\tStack Overflow!!");
	10 *top = 0; // reset top to 0
	11 }
	12 else {
	13 stack[*top] = ele;
	14 *top += 1; // increment top by 1
	15 }
	16 }
	17 int pop(int stack[MAX], int *top) {
	18 // function to delete element from Stack
	19 if(*top <= 0) {
	20 printf("\nStack Underflow!!");
	21 return -1;
	22 }
	23 else {
	24 *top -= 1; // decrement top
	25 return stack[*top];
	26 }

```
}
void check()    {
    // function to check if given number is palindrome or not
    char s[MAX];
    int stack[MAX];
    int i, flag=0, top=0, temp;
    printf("\nEnter any number: ");
    scanf("%s", &s);
    for (i=0; i<strlen(s); i++)    {
        temp = s[i];
        push(stack, &top, temp);
    }

    for (i=0; i<strlen(s)/2; i++)  {
        if (stack[i] != pop(stack, &top))   {
            flag = 1;
            break;
        }
    }
    if (flag == 1)  {
        printf("\nNumber isn't Palindrome");
    }
    else    {
        printf("\nNumber is Palindrome");
    }

}
void display(int stack[MAX], int *top)  {
```

```
55      // function to show elements of Stack
56      int i;
57      printf("\nStack : ");
58      for(i=0; i<*top; i++)   {
59          printf("%d ", stack[i]);
60      }
61  }
62
63  int main(int argc, char const *argv[]) {
64      int i, top=0, ch, temp;
65      int stack[MAX];
66
67      while(1)    {       // Repeat again and again
68          printf("\n------**MENU**------");          // The driving
                Menu
69          printf("\n1. Push");
70          printf("\n2. Pop");
71          printf("\n3. Palindrome Check");
72          printf("\n4. Display");
73          printf("\n5. Exit");
74          printf("\nEnter your choice: ");           // Ask user for
                operation
75          scanf("%d", &ch);                          // Get in ch
76          switch(ch)  {                              // Driving block
77              case 1: printf("\nEnter element: ");
78                      scanf("%d", &temp);
79                      push(stack, &top, temp);
80                      break;
81              case 2: temp = pop(stack, &top);
82                      printf("\nPopped off element: %d", temp);   //
                            display deleted
83                      break;
84              case 3: check();
85                      break;
86              case 4: display(stack, &top);
87                      break;
88              case 5: return 0;                      // No need to
                    write break
89              default: printf("\nWrong Choice!!! \t Enter between 1 to
                    5");
90                      break;
91          }
92      }
93      return 0;
94  }
```

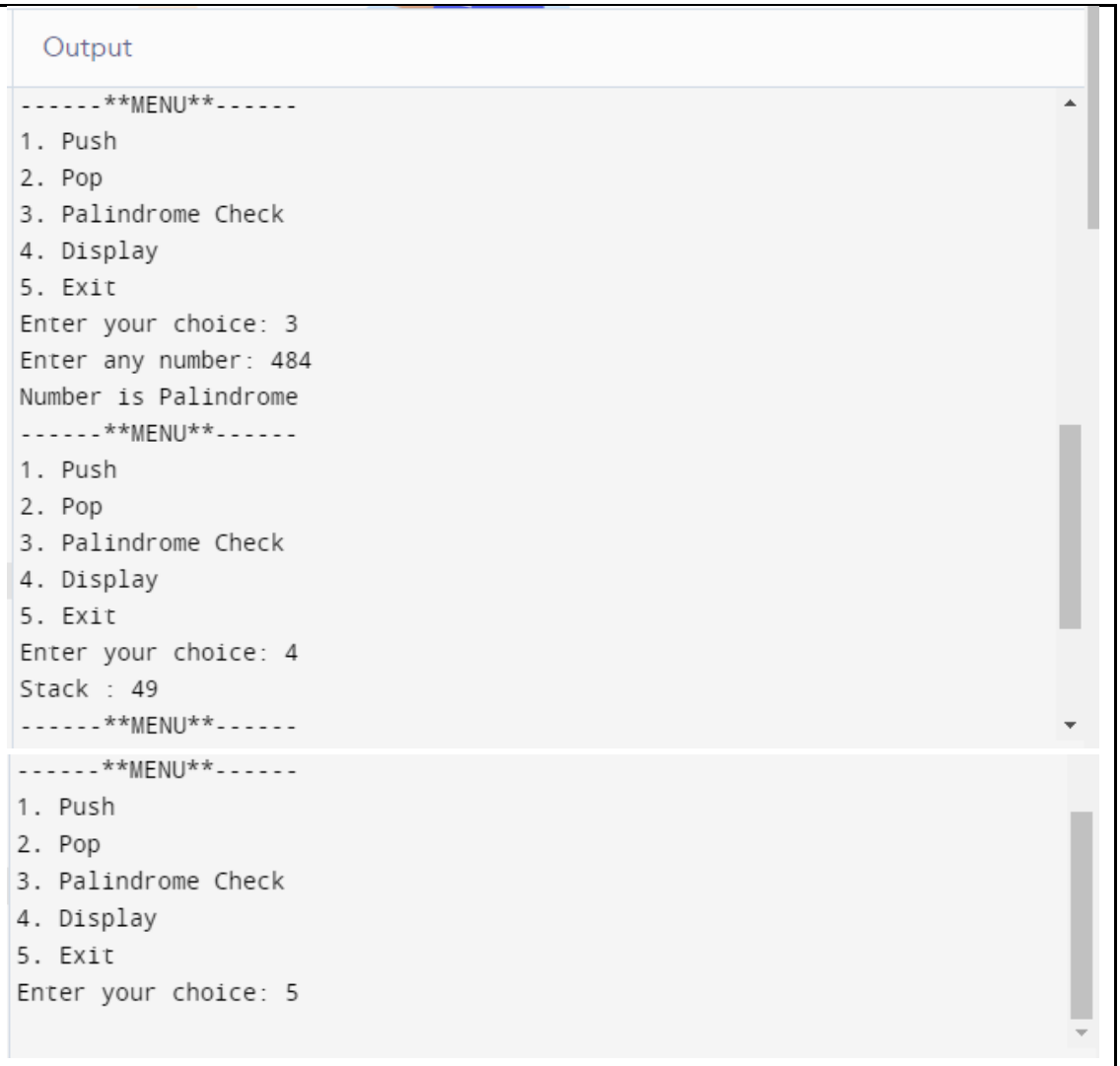
Output

Output

```
------**MENU**------
1. Push
2. Pop
3. Palindrome Check
4. Display
5. Exit
Enter your choice: 2
Stack Underflow!!
Popped off element: -1
------**MENU**------
1. Push
2. Pop
3. Palindrome Check
4. Display
5. Exit
Enter your choice: 1
Enter element: 49
------**MENU**------
```

Output

```
Enter element: 63
------**MENU**------
1. Push
2. Pop
3. Palindrome Check
4. Display
5. Exit
Enter your choice: 1
Enter element: 65
------**MENU**------
1. Push
2. Pop
3. Palindrome Check
4. Display
5. Exit
Enter your choice: 1
Enter element: 23
Stack Overflow!!
```

	
Conclusion	Hence, successfully completed displaying the Push, Pop, Stack overflow, Stack underflow and Palindrome implementation in a C program.