Question
Design and implement an algorithm to evaluate an arithmetic expression which may be any form (postfix, prefix, infix), and demonstrate its working with suitable examples

Aim:
To evaluate given postfix expression

Algorithm:

- Firstly take a pointer that points towards the first character of the string
- Now check traverse through the string and check whether the character is an operand or an operator
- If the character is an operand then push it into the stack
- If its an operator then pop 2 operands from the stack and perform the operation.Push the result into the stack
- Now pop out the last element in the stack to obtain the final result

Program

```
//To design an algorith to evaluate given postfix expression

#include<stdio.h>
#include<ctype.h>
#include<math.h>
#define size 40
int top=1;
int stack[40];
void push(int val){     // push function
    if(top==size-1){
        printf("stack underflow");
        }
    else{
        top++;
        stack[top]=val;
    }
}
int pop(){  //pop function
    if(top==-1){
        printf("stack underflow");
        return 0;
        }
    else{
        int ele=stack[top];
        top--;
```

```
        return ele;
        }
}

int main(){
    char exp[40];
    printf("enter the expression ");
    scanf("%s",exp);
    char *e;
    int t1,t2,res;  //temp variables
    e = exp;
    while(*e!='\0'){    // loop to traverse through the expression
        if(isalnum(*e)){  //to check if given character is a number or a
letter from the english alphabet
            push((*e-48));
        }
        else{
            t1=pop();
            t2=pop();
            switch(*e){  //performs corresponding operations according to
the character
                case'+':
                    res=t2+t1;
                    break;
                case'-':
                    res=t2-t1;
                    break;
                case'*':
                    res=t2*t1;
                    break;
                case'/':
                    res=t2/t1;
                    break;
                case'^':
                    res=pow(t2,t1);
                    break;
                }
            push(res);
        }
        e++;
```

```
    }
    res=pop();
    printf("the result is %d", res); //displays final result
}
```

Output

```
enter the expression 12+
the result is 3
PS E:\code\prefix> cd "e:\code\prefix\" ; if ($?) { gcc postfixeval.c -o postfixeval } ; if ($?) { .\postfixeval }
enter the expression 12+3+
the result is 6
```