Program

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

#define MAX 5

int stack[MAX];
int top = -1;

void push(int item){
    if(top == MAX - 1){
        printf("Stack overflow !!");
    }
    else{
        top++;
        stack[top] = item;
    }
}

int pop(){
    if(top == -1){
        printf("Stack underflow !!");
        return -1;
    }
    else{
        int item = stack[top];
        top--;
        return item;
    }
}

int evaluate(char expression[]){
    int i = 0;
    int num1, num2, result;
    while(expression[i] != '\0'){
        if(expression[i] >= '0' && expression[i] <= '9'){
            int num = expression[i] - '0';
            push(num);
    }
        else if(expression[i] == '+' || expression[i] == '-' || expression[i] == '*' || expression[i] == '
            num2 = pop();
            num1 = pop();

            if(num2 == -1 || num1 == -1){
                printf("Add operands !!\n");
            }

        switch(expression[i]){
            case '+':
                result = num1 + num2;
                break;
            case '-':
                result = num1 - num2;
                break;
            case '*':
                result = num1 * num2;
                break;
            case '/':
                if(num2 == 0){
```

Experiment No: 7
Date:

POSTFIX EXPRESSION EVALUATION

Aim:

To evaluate a postfix expression

Algorithm:

1. Start
2. Define MAX as 5
3. Initialize top = -1 and declare stack[size]
4. Define function push(item)

```
if (top = MAX - 1)
    print "Stack Overflow!!"
else
    top++
    stack[top] = item

end of function
```

5. Define function pop()

```
if (top = -1)
    print "Stack Underflow"
    return -1
else
    item = stack[top]
    top--
    return item

end of function
```

6. Define function evaluate(expression)

```
initialize i = 0
initiate while loop
while (expression[i] ! = '\0')
    if (expression[i] >= '0' and expression[i] <= '9')
        num = expresssion[i]
        push(num)
```

```
                    printf("Can't divide by zero !!");
                    return -1;
                }
                result = num1 / num2;
                break;
            case '^':
                result = pow(num1, num2);
                break;
        }
        push(result);
        }
        i++;
    }
    result = pop();
    return result;
}


int main(){
    char expression[MAX];
    printf("Enter the expression: ");
    scanf("%s", expression);
    int result = evaluate(expression);
    printf("Result : %d",result);
}
```

Output

```
Enter the expression: 562+3*+
Result : 29
```

```
    else if(expression[i] == '+' || expression[i] == '-' || expression[i] == '*' || expression[i]
                num2 = pop()
                num1 = pop()
                if (num2 == -1 || num1 == -1)
                     print "Add operand!!"
                endif
                switch(expression[i])
                    case '+':
                        result = num1 + num2;
                    case '-':
                        result = num1 - num2;
                    case '*':
                        result = num1 * num2;

                    case '/':
                        if(num2 == 0)
                            printf("Can't divide by zero !!")
                        result = num1 / num2;
                    case '^':
                        result = pow(num1, num2);
                endswitch
                push(result)
    endif
    i++
 endwhile
 result = pop()
 return result

 end of function
```

7. Create main function

```
main()
    Read user input expression
    result = evaluate(expression)
    print the result
end of main function
```

8. Stop

Result:

Program has been executed successfully and obtained the output.