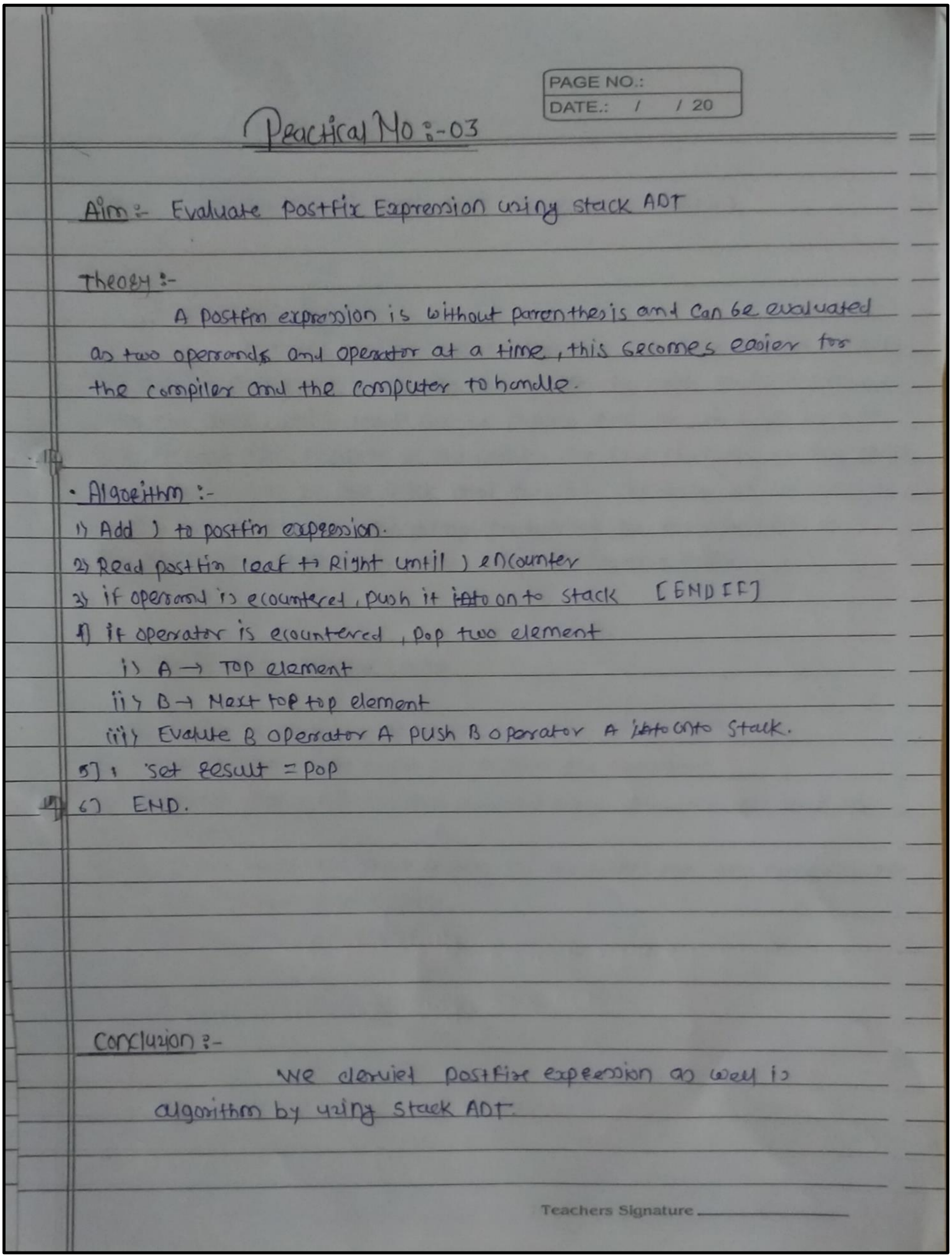
Jawahar Education Societys Annasaheb Chudaman Patil College of Engineering, Kharghar, Navi Mumbai

NAME: PRIYUSH BHIMRAO KHOBRAGADE

PRN NO: 211112018

SUBJECT: DATA STRUCTURES LAB

AIM: Evaluate Postfix Expression using Stack ADT.



PAGE NO.:
DATE.: / / 20

Practical No :-03

Aim :- Evaluate postfix Expression using stack ADT

Theory :-

A postfix expression is without parenthesis and can be evaluated as two operands and operator at a time, this becomes easier for the compiler and the computer to handle.

· Algorithm :-

```
1) Add ) to postfix expression.
2) Read postfix left to Right until ) encounter
3) if operand is ecountered, push it ~~into~~ onto stack   [END IF]
4) if operator is ecountered, pop two element
    i) A → Top element
    ii) B → Next top top element
    iii) Evalute B operator A push B operator A ~~into~~ onto Stack.
5] : Set result = pop
6) END.
```

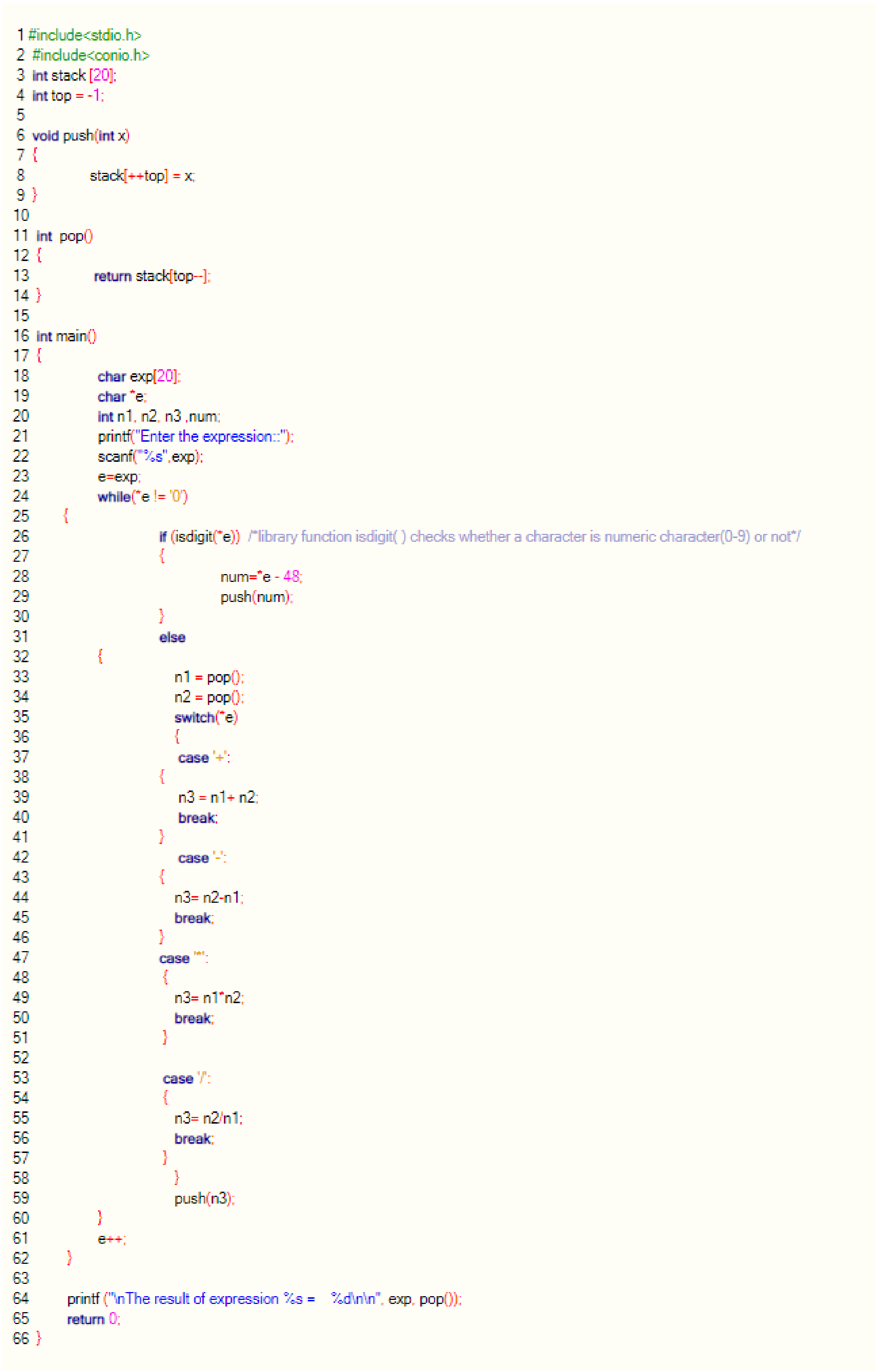
Conclusion :-

We derived postfix expression as well is algorithm by using stack ADT.

Teachers Signature ________

AIM: Evaluate Postfix Expression using Stack ADT.

Input:



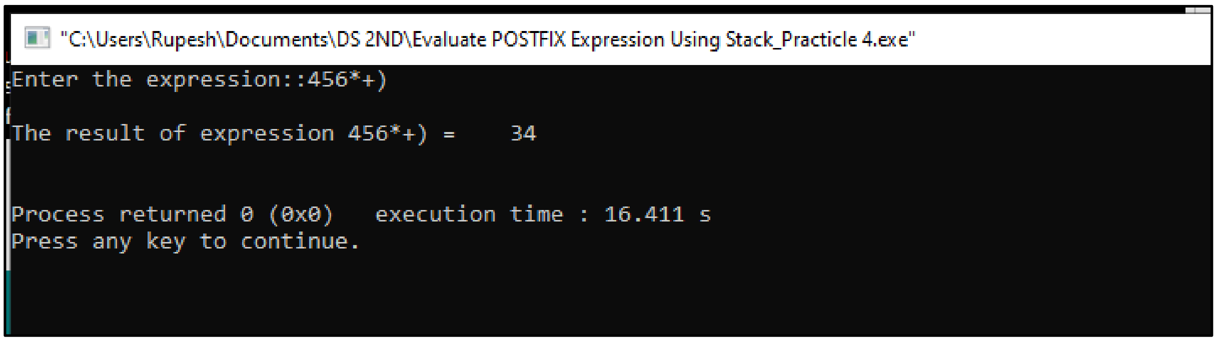
```
#include<stdio.h>
 #include<conio.h>
 int stack [20];
 int top = -1;

 void push(int x)
 {
            stack[++top] = x;
 }

 int  pop()
 {
            return stack[top--];
 }

 int main()
 {
            char exp[20];
            char *e;
            int n1, n2, n3 ,num;
            printf("Enter the expression::");
            scanf("%s",exp);
            e=exp;
            while(*e != '0')
      {
                        if (isdigit(*e))  /*library function isdigit( ) checks whether a character is numeric character(0-9) or not*/
                        {
                                    num=*e - 48;
                                    push(num);
                        }
                        else
            {
                          n1 = pop();
                          n2 = pop();
                          switch(*e)
                          {
                           case '+':
                        {
                           n3 = n1+ n2;
                           break;
                        }
                           case '-':
                        {
                          n3= n2-n1;
                          break;
                        }
                        case '*':
                         {
                          n3= n1*n2;
                          break;
                         }

                         case '/':
                         {
                          n3= n2/n1;
                          break;
                         }
                          }
                          push(n3);
            }
            e++;
      }

      printf ("\nThe result of expression %s =   %d\n\n", exp, pop());
      return 0;
 }
```

Priyush B. Khobragade

AIM: Evaluate Postfix Expression using Stack ADT.

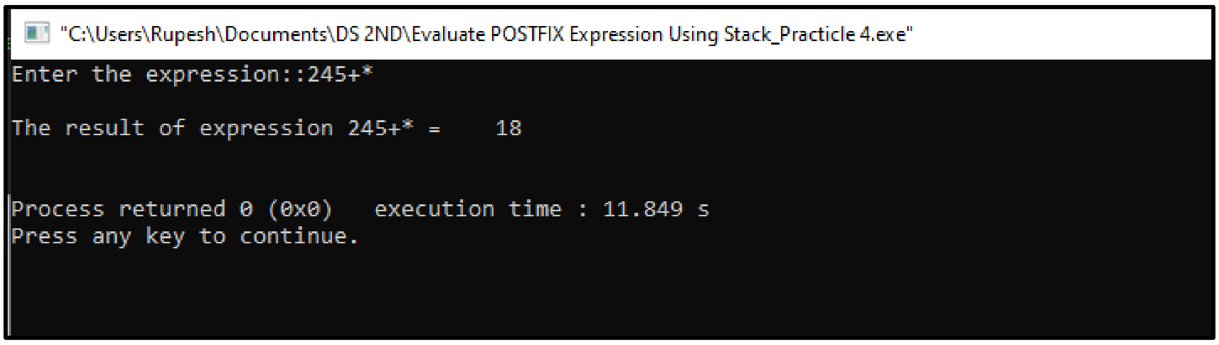
<u>Output:-</u>

Test 01



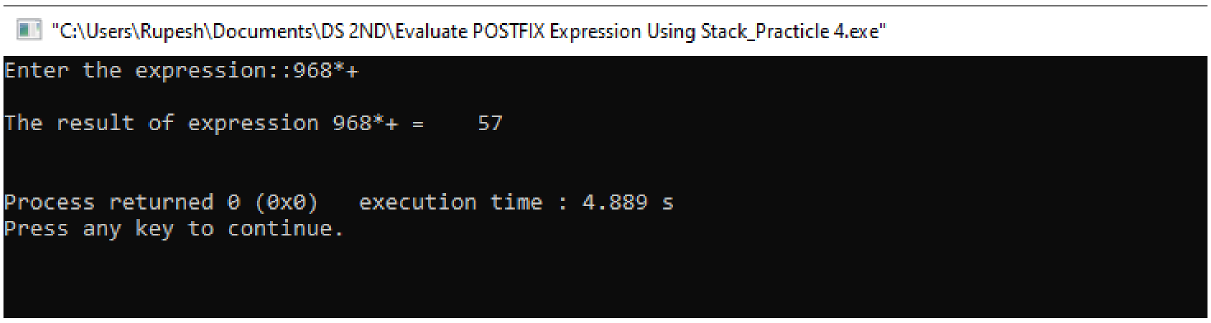
```
"C:\Users\Rupesh\Documents\DS 2ND\Evaluate POSTFIX Expression Using Stack_Practicle 4.exe"
Enter the expression::456*+)

The result of expression 456*+) =    34


Process returned 0 (0x0)   execution time : 16.411 s
Press any key to continue.
```

Test 02



```
"C:\Users\Rupesh\Documents\DS 2ND\Evaluate POSTFIX Expression Using Stack_Practicle 4.exe"
Enter the expression::245+*

The result of expression 245+* =    18


Process returned 0 (0x0)   execution time : 11.849 s
Press any key to continue.
```

Test 03



```
"C:\Users\Rupesh\Documents\DS 2ND\Evaluate POSTFIX Expression Using Stack_Practicle 4.exe"
Enter the expression::968*+

The result of expression 968*+ =    57


Process returned 0 (0x0)   execution time : 4.889 s
Press any key to continue.
```

<u>Conclusion</u>: - We know about implement to <u>postfix</u> expression algorithm well as is al using stack ADT.

Priyush B. Khobragade