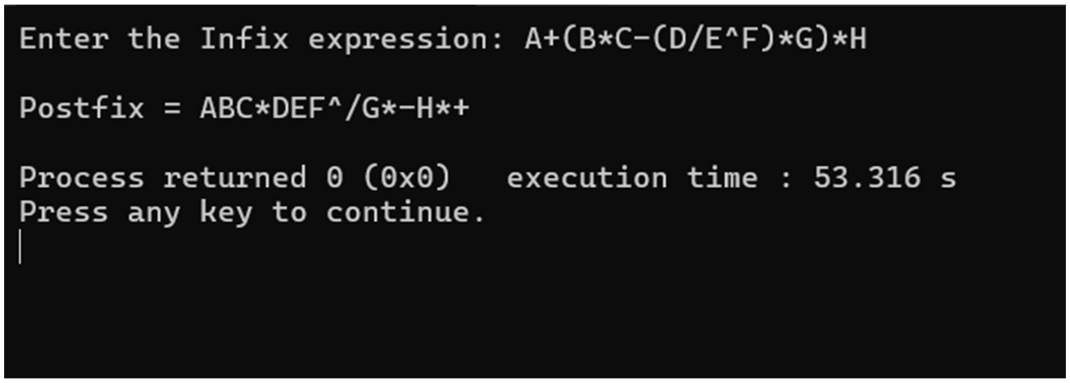

```
Enter the Infix expression: A+(B*C-(D/E^F)*G)*H

Postfix = ABC*DEF^/G*-H*+

Process returned 0 (0x0)   execution time : 53.316 s
Press any key to continue.
```

06/10

LAB PROGRAM-2

WAP to convert a given valid parenthesized infix arithmetic expression to postfix expression. The expression consists of single character operands and the binary operators + (plus), minus, * multiply and / divide.

```
#include <stdio.h>
#include <c.type>
#define size 50

Char stack [size];
int top =-1;
void push (char element)
  {
    stack [++top]=element;
  }
Char pop() {
     return stack [top--];
  }
int pr(char symbol) {
    if (symbol = ='A')
     {
        return 3;
      }
   else if (Symbol ='*' || Symbol ='/')
   {
    return 2;
     }
   else if (Symbol ='+' || Symbol =' -
   {
     return 1;
       }
   else return 0;
```

```
int main () {
  char infix [50], postfix [50], ch, element
   int i=0, k=0;
  printf ("Enter the expression");
  gets ("%s", infix); push ('#')
  While ((ch = infix [i++]) != '\0'){
        if (ch= '(') {
              push (ch);
           }
      else if (isalnum (unsigned char) ch)){
          postfix [k++] = ch;
        }
      else if (ch=')'{
        While (stack [top] != '(')
                postfix [k++] = pop();
          element = pop();
          (void) element;
                                  }
      else {
        ~~et~~ While (stack [top] != '(' && Pr(stack
                                        [top])
                    >= Pr (ch))
              post fix [k++] = pop ();
          push (ch);
      }
       While (stack [top] != '#')
              postfix [k++] = pop ();
              post fix [k] = '\0';
          printf ("\n Postfix = %s\n", Postfix);
              return 0;
                }
```

Output

```
Enter the infix expression : (A+(B*C-(D
G^F)*G)*H)
postfix = ABC* DEF^/G*-H*+
```

6/10