```
1  //############### COMPILER DESIGN (2) LAB #############
2  //############ Intermediate Code Generator ###########
3  //############### Input: math. expression #############
4  //############## Output: Intermediate Code ############
5  //############## Author: AHMED A. ALSAMMAN ############
6  using System;
7  using System.Collections.Generic;
8  using System.IO;
9  using System.Linq;
10 using System.Text;
11
12 namespace IMCG
13 {
14     class Program
15     {
16         static Stack  <char>  opStack = new Stack<char>();
17         static Stack<string> argStack = new Stack<string>();
18         static int t = 0;
19         static StreamWriter sw;
20
21         static void Main(string[] args)
22         {
23            string lhs;
24
25             while (true)
26             {
27                 // id=b+c*d/e
28                 Console.Write("Enter LHS (var): ");
29                 lhs = Console.ReadLine().Trim().Replace(" ","");
30                 Console.Write("Enter RHS: ");
31                 string infix = Console.ReadLine().Trim().Replace(" ", ""); ;
32                 Console.WriteLine("{0}={1}", lhs, infix);
33
34                 if (validExpression(infix))
35                 {
36                     sw = new StreamWriter(@"d:\imc.txt");
37                     t = 1;
38                     foreach (char c in infix)
39                     {
40
41                         if (Char.IsLetter(c) || Char.IsDigit(c))
42
43                             argStack.Push(c.ToString());
44
45                         if (c == '(')
46                             opStack.Push(c);
47                         if (c == ')')
48                           {
49                             while (opStack.Count != 0 && opStack.Peek() != '(')
50                                 applyOpr();
51                             opStack.Pop();
52                           }
```

AHMED A. ALSAMMAN

```
53
54                          if (isOperator(c))
55                            {
56                              while (opStack.Count != 0 && Priority(opStack.Peek())
                        >= Priority(c))
57                                    applyOpr();
58                              opStack.Push(c);
59                            }
60                      }
61
62                      while (opStack.Count != 0)//if any operator remains in the
                        stack,
63                          //pop all elements until stack is empty
64                          applyOpr();
65                      sw.WriteLine("{0} = t{1}", lhs, t - 1);
66                      sw.Close();
67                      Console.WriteLine("Intermediate code generated
                        successfully");
68
69                  }
70                  else
71                  {
72                      sw.WriteLine("Invalid Expresion");
73                  }
74              }
75          }
76
77          private static void applyOpr()
78          {
79              char opr = opStack.Pop();
80              string right = argStack.Pop();
81              string  left = argStack.Pop();
82              sw.WriteLine("t{0} = {1} {2} {3}", t, left, opr, right);
83              argStack.Push("t" + t);
84              t++;
85          }
86
87          private static bool validExpression(string infix)
88          {
89              foreach (char c in infix)
90                  if (!Char.IsDigit(c) && !Char.IsLetter(c) && !isOperator(c)
91                      && c != '(' && c != ')')
92                      return false;
93              return true;
94          }
95
96
97
98
99
100
101
```

AHMED A. ALSAMMAN

```
102
103         static int Priority(char c)
104         {
105             switch (c)
106             {
107                 case '^': return 3;
108                 case '*':
109                 case '/': return 2;
110                 case '+':
111                 case '-': return 1;
112                  default: return 0;
113             }
114         }
115
116         static bool isOperator(char c)
117         {
118             if (c == '+' || c == '-' || c == '*' || c == '/' || c == '^')
119                 return true;
120             return false;
121         }
122     }
123 }
```

AHMED A. ALSAMMAN