```
// Program to implement Pass-1 assembler.

import java.io.*;
import java.util.*;

class Pass-1_Assembler
{
public static void main(String args[])throws Exception
{
int loc=0,loc1=0;
int j=0,k,sym=0,lit=0,v=0,w=0;

FileReader f1=new FileReader("in.txt");
BufferedReader b1=new BufferedReader(f1);

FileWriter f3=new FileWriter("symboltab.txt");
BufferedWriter b3=new BufferedWriter(f3);

FileWriter f4=new FileWriter("intermediate.txt");
BufferedWriter  b4=new BufferedWriter(f4);

FileWriter f5=new FileWriter("literaltab.txt");
BufferedWriter b5=new BufferedWriter(f5);

StringBuffer sb = new StringBuffer();
String s,s3,s4,tmp,sd,s1;
String s1[]=new String[4];
String s2[]=new String[4];
String s5[][]=new String[4][2];
while((s=b1.readLine())!=null)
{
 int flag=0;
 int m=0;
 StringTokenizer st=new StringTokenizer(s);
 while(st.hasMoreTokens())
 {
 s1[m]=st.nextToken();
 m++;
 }
 if(s1[1].equals("START"))
 {
  b4.write("AD 01");
  b4.write(" C "+s1[2]);
  loc=Integer.parseInt(s1[2]);
  b4.newLine();
 }
 else if(s1[1].equals("END"))
 {
  for(int i=1;i<=lit;i++)
  {
  b4.write(loc+"  ");
  b4.write("AD 02");

  b4.write(" C "+i);
  loc=loc+1;
  b4.newLine();
 }
 }
 else
 {
  b4.write(loc+"  ");
  if(s1[0].equals("-")!=true)
  {
  sym++;
  b3.write(" "+s1[0]);
  b3.write(" "+loc+" ");
```

```
  }
FileReader f2=new FileReader("op.txt");
BufferedReader b2=new BufferedReader(f2);
  while((sl=b2.readLine())!=null)
  {
   j=0;
   StringTokenizer se=new StringTokenizer(sl);
   while(se.hasMoreTokens())
   {
    s2[j]=se.nextToken();
    j++;
   }
   if(s1[1].equals(s2[0])==true)
   {
    for(int a=1;a<3;a++)
    {
     b4.write("  ");
     b4.write("   "+s2[a]);
    }
   }
   if(s1[2].equals(s2[0])==true)
   {
    for(int a=1;a<2;a++)
    {
     b4.write("  ");
     b4.write("   "+s2[a]);
    }
   }

  }
  if(s1[3].contains("="))
   {
int i;flag=0;
for( i=0;i<lit;i++)
{
if(s5[i][1].equals(s1[3]))
{flag=1;
break;}
}
if(flag!=1)
{
    lit++;
s5[v][0]=Integer.toString(lit);
s5[v][1]=s1[3];
v++;
    b5.newLine();
    b5.write(" "+lit);
    b5.write(" "+s1[3]);
    //b6.write(" "+loc1+" ");
    b4.write(" ");
    b4.write(" "+"L  "+lit);
}
else
{
b4.write(" ");
b4.write(" "+"L  "+s5[i][0]);
}

   }

   else if((s1[3]).matches("[a-zA-Z]"))
   {
    //b4.write(" ");
    b4.write(" "+"S  "+sym);

   }
```

```
  loc=loc+1;

  b4.newLine();

  b4.newLine();
  b3.newLine();

 }

}
b4.close();
b3.close();
b5.close();
}
}


// Input files:

// in.txt

// - START 100
// L1 MOVER AREG =3
// - MOVEM BREG X
// - SUB AREG =1
// - LTORG
// - MOVEM AREG Y
// - BC any L1
// - ADD CREG,4
// X DC 6
// Y DS 2
// - END

// op.txt

// STOP IS 01
// ADD IS 02
// SUB IS 03
// MUL IS 04
// MOVER IS 05
// MOVEM IS 06
// START AD 01
// END AD 02
// ORIGIN AD 03
// LTORG AD 04
// AREG R1
// BREG R2
// CREG R3
// DREG R4
// DC DL 01
// DS DL 02


// ************************************OUTPUT*********************************

// symboltab.txt

//  L1 100
//  X 107
//  Y 108


// intermediate.txt

// AD 01 C 100
// 100       IS     05     R1  L  1
```

```
// 101       IS     06     R2 S  1

// 102       IS     03     R1  L  2

// 103       AD     04     R1  L  2

// 104       IS     06     R1 S  1

// 105

// 106       IS     02

// 107       DL     01

// 108       DL     02

// 109  AD 02 C 1

// 110  AD 02 C 2

// literaltab.txt

//  1 =3
//  2 =1


// This code is contributed by Prof. Anand Gharu
```