SPOS 1

INPUT:

```
import java.io.*;
public class pass1
{
	public static void main(String args[])throws Exception
	{
		FileReader FP=new FileReader("/home/student/input.txt");
		BufferedReader bufferedReader = new BufferedReader(FP);


		String line=null;
		int
line_count=0,LC=0,symTabLine=0,opTabLine=0,litTabLine=0,poolTabLine=0;

		 //Data Structures
		 final int MAX=100;
		 String SymbolTab[][]=new String[MAX][3];
		 String OpTab[][]=new String[MAX][3];
		 String LitTab[][]=new String[MAX][2];
		 int PoolTab[]=new int[MAX];
		 int litTabAddress=0;
//---------------------------------------------------------------------------
--------------------/

		 System.out.println("_________________");
		    while((line = bufferedReader.readLine()) != null)
		     {
		       String[] tokens = line.split("\t");
		      if(line_count==0)
		      {
				LC=Integer.parseInt(tokens[1]);

	//set LC to operand of START
				for(int i=0;i<tokens.length;i++)          //for printing
the input program
					System.out.print(tokens[i]+"\t");
				System.out.println("");
		      }
		      else
		      {
				 for(int i=0;i<tokens.length;i++) //for printing the
input program
					System.out.print(tokens[i]+"\t");
				 System.out.println("");
				if(!tokens[0].equals(""))
				{

					//Inserting into Symbol Table
					SymbolTab[symTabLine][0]=tokens[0];
					SymbolTab[symTabLine][1]=Integer.toString(LC);
					SymbolTab[symTabLine][2]=Integer.toString(1);
					symTabLine++;
				}
				else if(tokens[1].equalsIgnoreCase("DS")||
tokens[1].equalsIgnoreCase("DC"))
				{
					//Entry into symbol table for declarative
statements
					SymbolTab[symTabLine][0]=tokens[0];
```

```
                        SymbolTab[symTabLine][1]=Integer.toString(LC);
                        SymbolTab[symTabLine][2]=Integer.toString(1);
                        symTabLine++;
                  }

                  if(tokens.length==3 && tokens[2].charAt(0)=='=')
                  {
                        //Entry of literals into literal table
                        LitTab[litTabLine][0]=tokens[2];
                        LitTab[litTabLine][1]=Integer.toString(LC);
                        litTabLine++;
                  }

                  else if(tokens[1]!=null)
                  {
                              //Entry of Mnemonic in opcode table
                        OpTab[opTabLine][0]=tokens[1];

                        if(tokens[1].equalsIgnoreCase("START")||
tokens[1].equalsIgnoreCase("END")||tokens[1].equalsIgnoreCase("ORIGIN")||
tokens[1].equalsIgnoreCase("EQU")||tokens[1].equalsIgnoreCase("LTORG"))
      //if Assembler Directive
                        {
                              OpTab[opTabLine][1]="AD";
                              OpTab[opTabLine][2]="R11";

                        }
                        else if(tokens[1].equalsIgnoreCase("DS")||
tokens[1].equalsIgnoreCase("DC"))
                        {
                              OpTab[opTabLine][1]="DL";
                              OpTab[opTabLine][2]="R7";

                        }
                        else
                        {
                              OpTab[opTabLine][1]="IS";
                              OpTab[opTabLine][2]="(04,1)";
                        }
                  opTabLine++;
                  }
               }
               line_count++;
               LC++;
            }

              System.out.println("_____________________");

              //print symbol table
              System.out.println("\n\n      SYMBOL TABLE            ");
              System.out.println("--------------------------");
              System.out.println("SYMBOL\tADDRESS\tLENGTH");
              System.out.println("--------------------------");
              for(int i=0;i<symTabLine;i++)
                    System.out.println(SymbolTab[i][0]+"\t"+SymbolTab[i]
[1]+"\t"+SymbolTab[i][2]);
              System.out.println("--------------------------");


              //print opcode table
              System.out.println("\n\n      OPCODE TABLE            ");
              System.out.println("---------------------------");
```

```
                  System.out.println("MNEMONIC\tCLASS\tINFO");
                  System.out.println("----------------------------");
                  for(int i=0;i<opTabLine;i++)
                        System.out.println(OpTab[i][0]+"\t\t"+OpTab[i]
[1]+"\t"+OpTab[i][2]);
                  System.out.println("----------------------------");

                  //print literal table
                  System.out.println("\n\n   LITERAL TABLE        ");
                  System.out.println("-----------------");
                  System.out.println("LITERAL\tADDRESS");
                  System.out.println("-----------------");
                  for(int i=0;i<litTabLine;i++)
                        System.out.println(LitTab[i][0]+"\t"+LitTab[i][1]);
                  System.out.println("------------------");


                  //intialization of POOLTAB
                  for(int i=0;i<litTabLine;i++)
                  {
                        if(LitTab[i][0]!=null && LitTab[i+1][0]!=null ) //if
literals are present
                        {
                              if(i==0)
                              {
                                    PoolTab[poolTabLine]=i+1;
                                    poolTabLine++;
                              }
                              else if(Integer.parseInt(LitTab[i]
[1])<(Integer.parseInt(LitTab[i+1][1]))-1)
                              {
                                    PoolTab[poolTabLine]=i+2;
                                    poolTabLine++;
                              }
                        }
                  }
                  //print pool table
                  System.out.println("\n\n   POOL TABLE           ");
                  System.out.println("-----------------");
                  System.out.println("LITERAL NUMBER");
                  System.out.println("-----------------");
                  for(int i=0;i<poolTabLine;i++)
                        System.out.println(PoolTab[i]);
                  System.out.println("------------------");



                // Always close files.
                bufferedReader.close();
      }
}
```

OUTPUT:

```
_________________
START 100
      READ  A
LABLE MOVER A,B
      LTORG
            ='5'
            ='1'
            ='6'
```

```
            ='7'
      MOVEM A,B
      LTORG
            ='2'
LOOP  READ  B
A     DS    1
B     DC    '1'
            ='1'
      END
```

```
      SYMBOL TABLE
--------------------------
SYMBOL      ADDRESS     LENGTH
--------------------------
LABLE 102   1
LOOP  111   1
A     112   1
B     113   1
--------------------------
```

```
      OPCODE TABLE
-----------------------------
MNEMONIC    CLASS INFO
-----------------------------
READ        IS    (04,1)
MOVER       IS    (04,1)
LTORG       AD    R11
MOVEM       IS    (04,1)
LTORG       AD    R11
READ        IS    (04,1)
DS          DL    R7
DC          DL    R7
END         AD    R11
-----------------------------
```

```
   LITERAL TABLE
-----------------
LITERAL     ADDRESS
-----------------
='5'  104
='1'  105
='6'  106
='7'  107
='2'  110
='1'  114
------------------
```

```
   POOL TABLE
-----------------
LITERAL NUMBER
-----------------
1
5
6
------------------
```