Pass1

import java.io.\*;

public class SymTab

{

public static void main(String args[])throws Exception

{

FileReader FP=new FileReader("/home/student/Desktop/Pass1/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();

}

}

Pass2

import java.io.BufferedReader;

import java.io.FileReader;

import java.io.FileWriter;

import java.io.IOException;

import java.util.HashMap;

public class Pass2 {

public static void main(String[] Args) throws IOException{

BufferedReader b1 = new BufferedReader(new FileReader("/home/student/Desktop/Pass2/intermediate.txt"));

BufferedReader b2 = new BufferedReader(new FileReader("/home/student/Desktop/Pass2/symtab.txt"));

BufferedReader b3 = new BufferedReader(new FileReader("/home/student/Desktop/Pass2/littab.txt"));

FileWriter f1 = new FileWriter("/home/student/Desktop/Pass2/Pass2.txt");

HashMap<Integer, String> symSymbol = new HashMap<Integer, String>();

HashMap<Integer, String> litSymbol = new HashMap<Integer, String>();

HashMap<Integer, String> litAddr = new HashMap<Integer, String>();

String s;

int symtabPointer=1,littabPointer=1,offset;

while((s=b2.readLine())!=null){

String word[]=s.split("\t\t\t");

symSymbol.put(symtabPointer++,word[1]);

}

while((s=b3.readLine())!=null){

String word[]=s.split("\t\t");

litSymbol.put(littabPointer,word[0]);

litAddr.put(littabPointer++,word[1]);

}

while((s=b1.readLine())!=null){

if(s.substring(1,6).compareToIgnoreCase("IS,00")==0){

f1.write("+ 00 0 000\n");

}

else if(s.substring(1,3).compareToIgnoreCase("IS")==0){

f1.write("+ "+s.substring(4,6)+" ");

if(s.charAt(9)==')'){

f1.write(s.charAt(8)+" ");

offset=3;

}

else{

f1.write("0 ");

offset=0;

}

if(s.charAt(8+offset)=='S')

f1.write(symSymbol.get(Integer.parseInt(s.substring(10+offset,s.length()-1)))+"\n");

else

f1.write(litAddr.get(Integer.parseInt(s.substring(10+offset,s.length()-1)))+"\n");

}

else if(s.substring(1,6).compareToIgnoreCase("DL,01")==0){

String s1=s.substring(10,s.length()-1),s2="";

for(int i=0;i<3-s1.length();i++)

s2+="0";

s2+=s1;

f1.write("+ 00 0 "+s2+"\n");

}

else{

f1.write("\n");

}

}

f1.close();

b1.close();

b2.close();

b3.close();

}

}