Experiment 1

RollNo:-3243

Title: Design suitable Data structures and implement Pass-I and Pass-II of a two- pass assembler for pseudo-machine. Implementation should consist of a few instructions from each category and few assembler directives. The output of Pass-I (intermediate code file and symbol table) should be input for Pass-II

```
package abc;


import java.io.*;
import java.util.*;
public class abc{


	public static void main(String[] args)throws
IOException,FileNotFoundException,ArrayIndexOutOfBoundsException
	{
		int lc=0;
		String code;
		BufferedReader br=new BufferedReader(new FileReader("E:\\SPOS
programs/Input.txt"));
		BufferedWriter wr=new BufferedWriter(new FileWriter("E:\\SPOS programs/IC.txt"));
		String line;
		INSTtable lookup=new INSTtable();
		LinkedHashMap<String,TableRow>Symtab=new LinkedHashMap();
		int symindex=0;


		while((line=br.readLine())!=null)
		{
			String parts[]=line.split("\\s+");
```

```
if(parts[1].equals("START"))
{
        lc=Integer.parseInt(parts[2]);
        code="(AD,01)\t"+"(C,"+lc+")";
        wr.write(code+"\n");

}
if(parts[1].equals("END"))
{

        code="(AD,02)\t";
        wr.write(code+"\n");
}
if(parts[1].equals("DS"))
{
        int j=Integer.parseInt(parts[2]);
        code="(DL,02)\t"+"(C,"+j+")";
        wr.write(code+"\n");
        lc=lc+j;

}
if(parts[1].equals("DC"))
{
        int j=Integer.parseInt(parts[2]);
        code="(DL,01)\t"+"(C,"+j+")";
        wr.write(code+"\n");
        lc=lc+1;
```

```
                    }
                    if(!parts[0].isEmpty())
                    {
                      if(Symtab.containsKey(parts[0]))
                              {
                                      Symtab.put(parts[0],new
TableRow(parts[0],lc,Symtab.get(parts[0]).getIndex()));
                              }
                       else
                         {
                              Symtab.put(parts[0],new TableRow(parts[0],lc,++symindex));


                         }

                    }
                    if(lookup.getType(parts[1]).equals("IS"))
                    {
                     code="(IS,0"+lookup.getCode(parts[1]) +")";
                     int j=2;
                     String code2="";
                     while(j<parts.length)
                     {
                              if(lookup.getType(parts[j]).equals("RG"))
                              {
                                      code2=code2+lookup.getCode(parts[j])+" ";


                              }
                              else
                              {
```

```
                                        if(Symtab.containsKey(parts[j]))
                                        {
                                                int ind=Symtab.get(parts[j]).getIndex();
                                                code2=code2+"(S,0"+ind+")";
                                        }
                                        else
                                        {
                                                Symtab.put(parts[j],new TableRow(parts[j],-
1,++symindex));
                                                int ind=Symtab.get(parts[j]).getIndex();
                                                code2=code2+"(S,0"+ind+")";
                                        }

                                }
                                j++;

                        }//while
                        code=code+" "+code2;
                        wr.write(code+"\n");

                    }
                }
                br.close();
                wr.close();
                BufferedWriter bws=new BufferedWriter(new FileWriter("SYMTAB.txt"));
                Iterator<String> itr=Symtab.keySet().iterator();
                System.out.println("Symbol Table\n");
                while(itr.hasNext())
                {
```

```
                        String key=itr.next().toString();
                        TableRow value=Symtab.get(key);

        System.out.println(value.getIndex()+"\t"+value.getSymbol()+"\t"+value.getAddress()+"\n");


        bws.write(value.getIndex()+"\t"+value.getSymbol()+"\t"+value.getAddress()+"\n");
                }
        bws.close();
        }

}
```

//----------INST TAB----------//

```
package abc;
import java.io.*;
import java.util.HashMap;
public class INSTtable {
        HashMap<String,Integer> AD,IS,DL,RG;
        public INSTtable()
        {
                AD=new HashMap<>();
                IS=new HashMap<>();
                DL=new HashMap<>();
                RG=new HashMap<>();

                AD.put("START",01);
                AD.put("END",02);
```

```
        AD.put("ORIGIN",03);
        AD.put("EQU",04);

        IS.put("STOP",00);
        IS.put("ADD",01);
        IS.put("SUB",02);
        IS.put("MUL",03);
        IS.put("MOVER",04);
        IS.put("MOVEM",05);
        IS.put("COMP",06);
        IS.put("BC",07);
        IS.put("DIV",8);
        IS.put("PRINT",10);
        IS.put("READ",9);

        DL.put("DC",01);
        DL.put("DS",02);

        RG.put("AREG",01);
        RG.put("BREG",02);
        RG.put("CREG",03);
        RG.put("DREG",04);


}

public String getType(String s)
{
        s=s.toUpperCase();
```

```
        if(AD.containsKey(s))
                return "AD";
        else if(IS.containsKey(s))
                return "IS";
        else if(DL.containsKey(s))
                return "DL";
        else if(RG.containsKey(s))
                return "RG";
        else
                return " ";
}

public int getCode(String s)
{
        s=s.toUpperCase();
        if(AD.containsKey(s))
                return AD.get(s);
        else if(IS.containsKey(s))
                return IS.get(s);
        else if(DL.containsKey(s))
                return DL.get(s);
        else if(RG.containsKey(s))
                return RG.get(s);
        else
                return -1;

}
```

```
}

//-----------------------------------------Table Row---------------------------------------//

package abc;



public class TableRow {
	String symbol;
	int address,index;



	public TableRow(String symbol,int address)
	{
		this.symbol=symbol;
		this.address=address;
		index=0;
	}
	public TableRow (String symbol,int address,int index)
	{
		this.symbol=symbol;
		this.index=index;
		this.address=address;


	}
	public String getSymbol()
	{
		return symbol;
	}
```

```
	public void setSymbol(String symbol)
	{
		this.symbol=symbol;


	}
	public int getAddress()
	{
		return address;
	}
	public void setAddress(int address)
	{
		this.address=address;
	}
	public int getIndex()
	{
		return index;
	}
	public void setIndex(int index)
	{
		this.index=index;
	}


}
```

//--------OUTPUT---------//

Input.txt

```
START 100
 ADD AREG A
 SUB CREG B
 MOVER AREG A
A DC 20
B DS 10
 END
```

IC.txt

(AD,01) (C,100)

(IS,01) 1 (S,01)

(IS,02) 3 (S,02)

(IS,04) 1 (S,01)

(DL,01) (C,20)

(DL,02) (C,10)

(AD,02)

//SYMTAB

Symbol Table

1 A 101

2 B 111

// pass2

```
package pass2;

import java.io.*;
import java.util.ArrayList;
import java.util.Scanner;

public class pass2{
  static ArrayList<TableRow> SYMTAB = new ArrayList<>();

   public static void main(String[] args) throws IOException {
      BufferedReader FRead = new BufferedReader(new FileReader("C:\\Users\\ADMIN\\eclipse-workspace\\abc\\SYMTAB.txt"));
      String line,code;

      while ((line = FRead.readLine()) != null){
            String temp[] = line.split("\\s+");
            SYMTAB.add(new TableRow(temp[1],Integer.parseInt(temp[2]),Integer.parseInt(temp[0])));
      }
      FRead.close();

      FRead = new BufferedReader(new FileReader("E:\\SPOS programs/out.txt"));
      BufferedWriter FWrite = new BufferedWriter(new FileWriter("E:\\SPOS programs\\pass2.txt"));

      while ((line = FRead.readLine()) != null){
```

```
String temp[] = line.split("\\s+");


if(temp[0].contains("AD") || temp[0].contains("DL,02")){
   FWrite.write("\n");
}
else if(temp.length==2){
   if(temp[0].contains("DL")){
      temp[0] = temp[0].replaceAll("[^0-9]","");
         int constant = Integer.parseInt(temp[1].replaceAll("[^0-9]",""));
         code = "00\t0\t"+String.format("%03d",constant)+"\n";
         FWrite.write(code);
   }
   else if(temp[0].contains("IS")){
      int opcode = Integer.parseInt(temp[0].replaceAll("[^0-9]",""));
            int symidx = Integer.parseInt(temp[1].replaceAll("[^0-9]",""));
            code = String.format("%02d",opcode)+"\t0\t"+String.format("%03d",SYMTAB.get(symidx-
1).getAddr())+"\n";
            FWrite.write(code);
   }
}
else if(temp.length==1 && temp[0].contains("IS")){
   int opcode = Integer.parseInt(temp[0].replaceAll("[^0-9]",""));
   code = String.format("%02d",opcode)+"\t0\t000"+"\n";
   FWrite.write(code);
}
else if(temp.length==3 && temp[0].contains("IS") ){
   int opcode = Integer.parseInt(temp[0].replaceAll("[^0-9]",""));
   int regcode = Integer.parseInt(temp[1]);
      int symidx = Integer.parseInt(temp[2].replaceAll("[^0-9]",""));
```

```
                code = String.format("%02d",opcode)+"\t"+regcode+"\t"+String.format("%03d",SYMTAB.get(symidx-1).getAddr())+"\n";
                FWrite.write(code);

            }
        }
        FWrite.close();
        FRead.close();


        System.out.println("Machine code is : ");
        Scanner sc = new Scanner(new File("E:\\SPOS programs/pass2.txt"));
        while (sc.hasNextLine()){
            System.out.println(sc.nextLine());
        }

    }
}

//-----------Table Row---------//
package pass2;

public class TableRow {

    String sym;
    int addr,idx;

    public TableRow(String s,int a) {
```

```
    sym = s;
    addr = a;
    idx=0;
}
public TableRow(String s,int a,int i){
        sym = s;
        addr = a;
        idx = i;
}

public String getSym(){
    return sym;
}
public void setSym(String s){
    sym=s;
}

public int getAddr() {
    return addr;
}
public void setAddr(int a){
    addr = a;
}

public int getIdx() {
    return idx;
}
public void setIdx(int idx) {
    this.idx = idx;
```

```
    }

}
```

//----------OUTPUT--------//

```
01      1       101
02      3       111
04      1       101
00      0       020
```