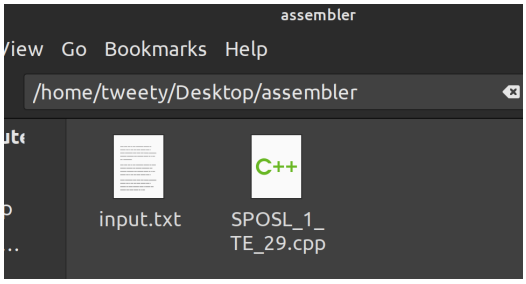
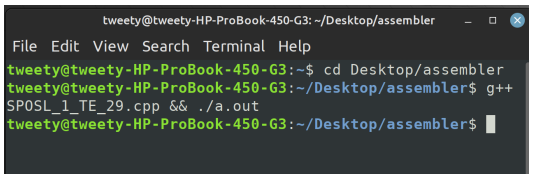
SPOSL Ass.1 : 29_Samruddhi Khairnar 17/10

Design suitable Data structures and implement both passes of a 2-pass assembler for pseudo machines. Implementation should consist of a few instructions from each category and assembler directives. Output of pass-I (Intermediate code file and symbol table) must be input for Pass-II.

Output :

Before Execution





Input File

*input.txt

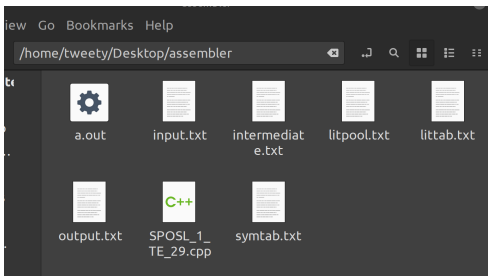
```
1  START 100
2  LAB MOVER AREG,N
3  ADD AREG,9
4  MOVEM AREG,N
5  COMP Z,LAB
6  BC GE,LAB
7  ORIGIN 200
8  MOVER BREG,='1'
9  LTORG
10 MOVER CREG,='5'
11 MOVEM CREG,T
12 STOP
13 N DC 1
14 T DS 2
15 Z EQU N
16 END
```

Intermediate Code(P-I O/P)

intermediate.txt

```
1  (AD,01) (C,100)
2  (S,00) (IS,04) (RG,01) N
3  (IS,01) (RG,01) (C,9)
4  (IS,05) (RG,01) N
5  (IS,06) Z LAB
6  (IS,07) (CC,04) LAB
7  (AD,03) (C,200)
8  (IS,04) (RG,02) (L,00)
9  ='1'
10 (IS,04) (RG,03) (L,01)
11 (IS,05) (RG,03) T
12 (IS,00)
13 (S,01) (DL,02) (C,1)
14 (S,02) (DL,01) (C,2)
15 (S,03) ='5'
```

After Execution



Pass-II Output

output.txt

```
1      100) 04 01 205
2      101) 01 01 009
3      102) 05 01 205
4      103) 06 205 100
5      104) 07 04 100
6      200) 04 02 201
7  ='1'    201)
8      202) 04 03 208
9      203) 05 03 206
10     204) 00 00 000
11     205) 1
12     206)
13 ='5'    208)
```

Symbol Table

symtab.txt

```
1 LAB 100
2 N 205
3 T 206
4 Z 205
```

Literal Table

littab.txt

```
1 ='1'
2 ='5'
```

Literal Pool Table

litpool.txt

```
1 ='1' 201
2 ='5' 208
```

Code:

```
//2-pass Assembler in Cpp - TE_29_Samruddhi Khairnar
#include <iostream>
#include <fstream>
using namespace std;
class Pass1
{
	private:
		string is[11] =
{"STOP","ADD","SUB","MULT","MOVER","MOVEM","COMP","BC","DIV","READ","PRINT"};
		int lc = 0,lptr=0;
		string buffer = "";
	public:
		void run()
		{
			fstream infile,littab,lpl;
			ofstream symout,outfile;
			infile.open("input.txt");
			outfile.open("intermediate.txt");
			littab.open("littab.txt",ios::out);littab<<"";littab.close();
			littab.open("littab.txt",ios::app);
			symout.open("symtab.txt",ios::out);symout<<"";symout.close();
			symout.open("symtab.txt",ios::app);
			lpl.open("litpool.txt",ios::out);lpl<<"";lpl.close();
			while(!infile.eof())
			{
				int nl = 0;
				string line;
				getline(infile,line);
				int ptr=0;
				//Counting no.of parts in line
				for(int i=0;i<line.length();i++)
				{
					if(line[i]==' ' || line[i]==',')
						ptr++;
				}
				//Making an array of parts:
				string *arr = new string[ptr+1];
				ptr = 0;
				for(int i=0;i<line.length();i++)
				{
					if(line[i]==' ' || line[i]==',')
						ptr++;
					else
						arr[ptr]+=line[i];
				}
				//Search Type and Store
				for(int i=0;i<ptr+1;i++)
				{
					if(arr[i]=="END")
```

```
{
	ofstream litpool;
	ifstream ilittab;
	litpool.open("litpool.txt",ios::app);
	ilittab.open("littab.txt",ios::in);
	string tmp="";
	for(int v=0;v<lptr-1;v++)
		getline(ilittab,tmp);
	while(!ilittab.eof())
	{
		getline(ilittab,tmp);
		if(tmp!="")
		{
			litpool<<tmp<<" "<<lc<<endl;
			buffer+=tmp+"\n";
			lc++;
		}
		nl=1;
	}

	ilittab.close();
	litpool.close();
	break;
}
else if(arr[i]=="START")
{
	buffer+="(AD,01) (C,"+arr[i+1]+") ";
	lc = stoi(arr[i+1]);
	i+=1;
}
else if(arr[i]=="ORIGIN")
{
	buffer+="(AD,03) (C,"+arr[i+1]+") ";
	lc = stoi(arr[i+1]);
	i+=1;
}
else if(arr[i]=="EQU")
{
	//Process Equ  - find in symbol_table and write
lc, lc++

	nl=1;
	string str="";
	symout.close();
	ifstream symin;
	symin.open("symtab.txt");
	string addr="",symbuff="";int fla=0;
	while(!symin.eof())
	{
		getline(symin,str);
		string t ="",name="";fla=0;
		for(int l=0;l<str.length();l++)
```

```
                        {
                                if(str[l]==' ')
                                        fla=1;
                                else if(fla==1)
                                        t+=str[l];
                                else if(fla==0)
                                        name+=str[l];
                        }
                        if(name==arr[i+1])
                        {
                                addr=t;
                                symbuff+=str+"\n";
                        }
                        else if(name==arr[i-1] && addr!="")
                                symbuff+=name+" "+addr+"\n";
                        else
                                symbuff+=str+"\n";
                }
                symout.open("symtab.txt",ios::out);
                symout<<symbuff;symout.close();
                symout.open("symtab.txt",ios::app);
                symin.close();
                i+=1;
        }
        else if(arr[i]=="LTORG")
        {
                //Process Pool - find in pooltab->littab->lc++
and write lc, lc++
                ofstream litpool;
                ifstream ilittab;
                litpool.open("litpool.txt",ios::app);
                ilittab.open("littab.txt",ios::in);
                string tmp="";
                for(int v=0;v<lptr-1;v++)
                        getline(ilittab,tmp);
                while(!ilittab.eof())
                {
                        getline(ilittab,tmp);
                        if(tmp!="")
                        {
                                litpool<<tmp<<" "<<lc<<endl;
                                buffer+=tmp+"\n";
                                lc++;
                        }
                        nl=1;
                }
                ilittab.close();
                litpool.close();
                i+=1;
        }
        else if(arr[i]=="DC")
```

```
                                {
                                        //Initialize Variables
                                        buffer+="(DL,02) (C,"+arr[i+1]+") ";
                                        while(i<ptr+1)
                                                i+=1;
                                        lc++;
                                }
                                else if(arr[i]=="DS")
                                {
                                        //Declare Variables
                                        buffer+="(DL,01) (C,"+arr[i+1]+") ";
                                        int incr = stoi(arr[i+1]);
                                        i+=1;
                                        lc+= incr;
                                }
                                else
                                {
                                        int flag=-1;
                                        for(int j=0;j<=10;j++)
                                        {
                                                if(is[j]==arr[i])
                                                        flag=j;
                                        }
                                        if(flag>=0)
                                        {

buffer+="(IS,"+(flag<=9?"0"+to_string(flag):to_string(flag))+") ";
                                                i++;
                                                while(i<ptr+1)
                                                {
                                                        //Process Operands
                                                        buffer+=(arr[i]=="AREG")?"(RG,01)
":(arr[i]=="BREG")?"(RG,02) ":(arr[i]=="CREG")?"(RG,03) ":"";
                                                        buffer+=(arr[i]=="LT")?"(CC,01)
":(arr[i]=="LE")?"(CC,02) ":(arr[i]=="GT")?"(CC,03) ":(arr[i]=="GE")?"(CC,04)
":(arr[i]=="EQ")?"(CC,05) ":(arr[i]=="NE")?"(CC,06) ":(arr[i]=="ANY")?"(CC,07) ":"";

                                                        if(arr[i][0]=='=')
                                                        {
                                                                littab<<arr[i]<<endl;

buffer+="(L,"+((lptr<=9)?"0"+to_string(lptr):to_string(lptr))+") ";
                                                                lptr++;
                                                        }
                                                        else if(arr[i]=="0" || arr[i]=="1"
|| arr[i]=="2" || arr[i]=="3" || arr[i]=="4" || arr[i]=="5" || arr[i]=="6" ||
arr[i]=="7" || arr[i]=="8" || arr[i]=="9")
                                                                buffer+="(C,"+(arr[i])+")
";
                                                        else if(arr[i]!="AREG" &&
arr[i]!="BREG" && arr[i]!="CREG" && arr[i]!="LE" && arr[i]!="LT" && arr[i]!="GE" &&
```

```
arr[i]!="GT" && arr[i]!="EQ" && arr[i]!="NE" && arr[i]!="ANY")
                                                        buffer+=arr[i]+" ";
                                                i++;
                                        }
                                        flag=-1;
                                        lc++;
                                }
                                else
                                {
                                        //Label Processing
                                        ifstream symin;
                                        symin.open("symtab.txt");
                                        int sptr=0,f=0;
                                        while(!symin.eof())
                                        {
                                                string t ="",temp="";
                                                getline(symin,t);
                                                sptr++;
                                        }
                                        symin.close();
                                        symout<<arr[i]<<" "<<lc<<endl;

buffer+="(S,"+((sptr-1<=9)?("0"+to_string(sptr-1)):(to_string(sptr-1)))+") ";
                                }
                        }
                        if(nl==0)
                                buffer+="\n";
                }
                outfile<<buffer;
                symout.close();
                littab.close();
                infile.close();
                outfile.close();
        }
};
class Pass2
{
        private:

        public:
                void run()
                {
                        ifstream infile;
                        infile.open("intermediate.txt");
                        ofstream outfile;
                        string buffer="";
                        int lc=0;
                        while(!infile.eof())
                        {
                                int aflag=0;
```

```
                        string line;
                        getline(infile,line);
                        int ptr=0;
                        //Counting no.of parts in line
                        for(int i=0;i<line.length();i++)
                        {
                              if(line[i]==' ')
                                    ptr++;
                        }
                        //Making an array of parts:
                        string *arr = new string[ptr+1];
                        ptr = 0;
                        for(int i=0;i<line.length();i++)
                        {
                              if(line[i]==' ')
                                    ptr++;
                              else
                                    arr[ptr]+=line[i];
                        }
                        for(int i=0;i<ptr+1;i++)
                        {
                              if(arr[i]=="(AD,01)"||arr[i]=="(AD,03)")
                              {
                                    string temp = arr[i+1];
                                    temp = temp.substr(3,temp.length()-4);
                                    lc=stoi(temp);
                                    aflag=1;
                                    i+=1;
                              }
                              else if(arr[i].substr(0,4)=="(IS,")
                              {
                                    string temp =
arr[i].substr(4,arr[i].length()-5);
                                    if(temp!="00")
                                          buffer+="    "+to_string(lc)+") "+temp+"
";
                                    else
                                          buffer+="    "+to_string(lc)+") "+temp+"
00 000";
                                    i++;
                                    while(i<ptr+1)
                                    {
                                          if(arr[i].substr(0,4)=="(RG," ||
arr[i].substr(0,4)=="(CC,")
                                          {
                                                temp =
arr[i].substr(4,arr[i].length()-5);
                                                buffer+=temp+" ";
                                          }
                                          else if(arr[i].substr(0,3)=="(C,")
                                          {
```

```
                                                temp =
arr[i].substr(3,arr[i].length()-4);

buffer+=((stoi(temp)<=9)?"00"+temp:(stoi(temp)<=99)?"0"+temp:temp)+" ";
                                        }
                                        else if(arr[i].substr(0,3)=="(L,")
                                        {
                                                temp =
arr[i].substr(3,arr[i].length()-4);
                                                ifstream littab;
                                                littab.open("litpool.txt");
                                                string t="";
                                                int flag=0;
                                                for(int j=0;j<=stoi(temp);j++)
                                                        getline(littab,t);
                                                temp="";
                                                for(int j=0;j<t.length();j++)
                                                {
                                                        if(t[j]==' ')
                                                                flag=1;
                                                        else if(flag==1)
                                                                temp+=t[j];
                                                }
                                                buffer+=temp+" ";
                                                littab.close();
                                        }
                                        else
                                        {
                                                ifstream symtab;
                                                symtab.open("symtab.txt");
                                                string t="",name="";
                                                int f=0;
                                                while(!symtab.eof())
                                                {
                                                        temp="",name="";
;                                                       getline(symtab,t);
                                                        for(int
i=0;i<t.length();i++)
                                                        {
                                                                if(t[i]==' ')
                                                                        f=1;
                                                                else if(f==1)
                                                                        temp+=t[i];
                                                                else if(f==0)
                                                                        name+=t[i];
                                                        }
                                                        if(name==arr[i] &&
temp!="")

buffer+=((stoi(temp)<=9)?"00"+temp:(stoi(temp)<=99)?"0"+temp:temp)+" ";
                                                        f=0;
```

```
                                                }
                                                symtab.close();
                                        }
                                        i++;
                                }
                                lc++;
                        }
                        else if(arr[i].substr(0,4)=="(DL,")
                        {
                                string temp =
arr[i].substr(4,arr[i].length()-5);
                                if(temp=="01") //DS
                                {
                                        buffer += "   "+to_string(lc)+")";

lc+=stoi(arr[i+1].substr(3,arr[i].length()-6));
                                        while(i<ptr+1)
                                                i++;
                                }
                                else //DC
                                {
                                        buffer += "   "+to_string(lc)+")
"+arr[i+1].substr(3,arr[i].length()-6);
                                        while(i<ptr+1)
                                                i++;
                                        lc++;
                                }
                        }
                        else if(arr[i][0]=='=')
                        {
                                buffer+=arr[i]+"    "+to_string(lc)+") ";
                                lc++;
                        }
                }
                if(aflag!=1 && arr[0]!="")
                        buffer+="\n";
        }
        outfile.open("output.txt",ios::out);
        outfile<<buffer;
        infile.close();
        outfile.close();
    }
};
int main()
{
        Pass1 pass1;
        pass1.run();
        Pass2 pass2;
        pass2.run();
        return 0;
}
```