System Software and Compiler Design

Lab Assignment 4

Name: Tushar Mittal

PRN: 1032200956

Roll No: PB68

Panel: B

Batch: B2

Code:

Macro.java

```
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.FileReader;
import java.io.FileWriter;
import java.util.ArrayList;
import java.util.Hashtable;

public class Macro {
```

```
ArrayList<String> mdt = new ArrayList<>();
ArrayList<String> alatab = new ArrayList<>();
ArrayList<String> alaac = new ArrayList<>();
Hashtable<String, MNTable> mnt = new Hashtable<>();

public void passTwo(BufferedReader br) throws Exception {
    BufferedWriter wrO = new BufferedWriter(new FileWriter("output.txt", true));
    BufferedWriter wrE = new BufferedWriter(new FileWriter("expansion.txt", true));
    String line;
    while ((line = br.readLine()) != null) {
        String[] words = line.split(" ");
        if (mnt.containsKey(words[0])) {
            wrO.write(line);
            wrO.newLine();
            ArrayList<String> actualArguments = new ArrayList<>();
            for (int i = 1; i < words.length; i++) {
                actualArguments.add(words[i].replace(",", ""));
            }
            MNTable mnTable = mnt.get(words[0]);
            Integer mdtIndex = mnTable.getMdtIndex();
            String mdtLine = mdt.get(mdtIndex);
            String[] mdtWords = mdtLine.split(" ");
            ArrayList<String> formalArguments = new ArrayList<>();
            for (int i = 1; i < mdtWords.length; i++) {
                formalArguments.add(mdtWords[i].replace(",", ""));
            }
            for (int i = 0; i < formalArguments.size(); i++) {
                String formalArgument = formalArguments.get(i);
                String actualArgument = actualArguments.get(i);
                Integer index = alatab.indexOf(formalArgument);
                alaac.set(index, actualArgument);
```

```
            }
            for (int i = mdtIndex + 1; i < mdt.size() && !(mdt.get(i)).equals("MEND"); i++) {
                mdtLine = mdt.get(i);
                String expansionFileLine = mdtLine;
                for (int j = 0; j < alatab.size(); j++) {
                    mdtLine = mdtLine.replace(alatab.get(j), "#" + (j + 1));
                    expansionFileLine = expansionFileLine.replace(alatab.get(j), alaac.get(j));
                }
                mdt.set(i, mdtLine);
                wrE.write(expansionFileLine);
                wrE.newLine();
            }
        } else {
            wrO.write(line);
            wrO.newLine();
            wrE.write(line);
            wrE.newLine();
        }
    }
    wrE.close();
    wrO.close();
    BufferedWriter wrMdt = new BufferedWriter(new FileWriter("pass2mdt.txt"));
    wrMdt.write("Index  Instruction\n");
    for (int i = 0; i < mdt.size(); i++) {
        wrMdt.write(String.format("%-6d %s%n", (i + 1), mdt.get(i)));
        wrMdt.newLine();
    }
    wrMdt.close();
    BufferedWriter wrAlaTab = new BufferedWriter(new FileWriter("alatab.txt"));
    wrAlaTab.write("Index       Formal-Argument       Actual-Argument\n");
    for (int i = 0; i < alatab.size(); i++) {
```

```
            wrAlaTab.write(String.format("#%-10d %-21s %-20s%n", (i + 1), alatab.get(i), alaac.get(i)));
            wrAlaTab.newLine();
        }
        wrAlaTab.close();
        BufferedWriter wrMnt = new BufferedWriter(new FileWriter("mnt.txt"));
        wrMnt.write("Index       Macro-Name       MDT-Index\n");
        for (String key : mnt.keySet()) {
            MNTable mnTable = mnt.get(key);
            wrMnt.write(String.format("%-11d %-16s %-10d%n", (mnTable.getMntIndex() + 1), mnTable.getMacroName(),
                    (mnTable.getMdtIndex() + 1)));
            wrMnt.newLine();
        }
        wrMnt.close();
    }

    public static void main(String[] args) throws Exception {
        Macro macro = new Macro();
        BufferedReader br2 = new BufferedReader(new FileReader("input.asm"));
        macro.passTwo(br2);
        br2.close();
    }
}
```

MNTable.java

```
public class MNTable {
    private Integer mntIndex;
    private String macroName;
    private Integer mdtIndex;
```

```
    public MNTable(Integer mntIndex, String macroName, Integer mdtIndex) {
        this.mntIndex = mntIndex;
        this.macroName = macroName;
        this.mdtIndex = mdtIndex;
    }

    public Integer getMntIndex() {
        return mntIndex;
    }

    public void setMntIndex(Integer mntIndex) {
        this.mntIndex = mntIndex;
    }

    public String getMacroName() {
        return macroName;
    }

    public void setMacroName(String macroName) {
        this.macroName = macroName;
    }

    public Integer getMdtIndex() {
        return mdtIndex;
    }

    public void setMdtIndex(Integer mdtIndex) {
        this.mdtIndex = mdtIndex;
    }
}
```

Input:

input.asm

```
MACRO
INCR &ARG1 &ARG2
ADD AREG &ARG1
ADD BREG &ARG2
MEND
START
MOVER AREG S1
MOVER BREG S1
INCR D1 D2
S1 DC 5
D1 DC 2
D2 DC 3
END
```

pass1mdt.txt

Index	Instruction
1	INCR &ARG1 &ARG2
2	ADD AREG &ARG1
3	ADD BREG &ARG2
4	MEND

mnt.txt

Index	Macro-Name	MDT-Index
1	INCR	1

alatab.txt

Index	Formal-Argument
#1	&ARG1
#2	&ARG2

Output:

pass2mdt.txt

Index	Instruction
1	INCR &ARG1 &ARG2
2	ADD AREG #1
3	ADD BREG #2
4	MEND

alatab.txt

Index	Formal-Argument	Actual-Argument
#1	&ARG1	D1
#2	&ARG2	D2

expansion.txt

```
START
MOVER AREG S1
MOVER BREG S1
ADD AREG D1
ADD BREG D2
S1 DC 5
D1 DC 2
D2 DC 3
END
```