**Assignment 2**

**Name** : Shlok Zanwar

**Batch** : C-3

**Roll No** : 323071

**Gr No.** : 21910163

Files Required for the programs (Input files etc.)

  

Programs And Output

**Week 1** (Takes InputA2W1.txt as input)

Code :

#include <bits/stdc++.h>

using namespace std;

struct mntStruct{

    string macroName;

    int mdtPointer;

};

string makeSubStr(string str, int start, int end){

    string ans = "";

    for(int i = start; i < end; i ++){

        ans += str[i];

    }

    return ans;

}

string removeCommaSpaces (string str) {

    int i;

    string commaReplaced = "", tabReplaced = "", spaceReplaced = " ";

    for(i = 0; i < str.length(); i ++){

        if(str[i] == '\t'){

            tabReplaced += " ";

        }

        else{

            tabReplaced += str[i];

        }

    }

    for(i = 0; i < tabReplaced.length(); i ++){

        if(tabReplaced[i] == ','){

            commaReplaced += " ";

        }

        else{

            commaReplaced += tabReplaced[i];

        }

    }

    for(i = 0; i < commaReplaced.length(); i ++){

        if(commaReplaced[i] == ' '){

            if( spaceReplaced[spaceReplaced.length() - 1] != ' ' ){

                spaceReplaced += " ";

            }

        }

        else{

            spaceReplaced += commaReplaced[i];

        }

    }

    if(spaceReplaced[0] == ' '){

        spaceReplaced.erase (spaceReplaced.begin());

    }

    if(spaceReplaced[spaceReplaced.length() - 1] == ' '){

        spaceReplaced.erase (spaceReplaced.end() - 1);

    }

    // cout << spaceReplaced << " heyyy\n";

    return spaceReplaced;

}

vector<string> splitStringBySpace (string str){

    vector<string> split;

    int i, lastSpace = -1;

    for(i = 0; i < str.length(); i ++){

        if(str[i] == ' '){

            split.push\_back( makeSubStr(str, lastSpace + 1, i) );

            lastSpace = i;

        }

    }

    split.push\_back( makeSubStr(str, lastSpace + 1, i) );

    return split;

}

int main(){

    vector< vector<string> > inputLinesSplit;

    vector< string > inputLines;

    string line;

    ifstream InputFile("InputA2W1.txt");

    if(InputFile.is\_open()){

        while (getline(InputFile, line)) {

            // cout << line << "\n";

            inputLines.push\_back( removeCommaSpaces(line) );

            inputLinesSplit.push\_back( splitStringBySpace( removeCommaSpaces(line) ) );

        }

    }

    else{

        cout << "Error in opening input file.\n";

    }

    int i, j;

    vector<string> mdt;

    vector<string> ic;

    vector<mntStruct> mnt;

    for(i = 0; i < inputLinesSplit.size(); i ++){

        if(inputLinesSplit[i][0] == "MACRO"){

            mntStruct newMacro;

            newMacro.macroName = inputLinesSplit[i][1];

            newMacro.mdtPointer = mdt.size();

            mnt.push\_back(newMacro);

            // cout << inputLinesSplit[i][1] << "\n";

            while(inputLinesSplit[i][0] != "MEND"){

                i += 1;

                mdt.push\_back(inputLines[i]);

            }

        }

        else{

            ic.push\_back(inputLines[i]);

        }

    }

    fstream outputFile;

    outputFile.open("OutputA2W1.txt", ios::out);

    cout << "\nIC :-\n-------------------------------------------\n";

    outputFile << "\nIC :-\n-------------------------------------------\n";

    for(i = 0; i < ic.size(); i ++){

        cout << ic[i] << "\n";

        outputFile << ic[i] << "\n";

    }

    cout << "\n\nMDT :-\n";

    cout << "-------------------------------------------\nIndex\t\tLine\n-------------------------------------------\n";

    outputFile << "\n\nMDT :-\n";

    outputFile << "-------------------------------------------\nIndex\t\tLine\n-------------------------------------------\n";

    for(i = 0; i < mdt.size(); i ++){

        cout << i << "\t\t" << mdt[i] << "\n";

        outputFile << i << "\t\t" << mdt[i] << "\n";

    }

    // cout << "\nlen - " << mnt.size();

    cout << "\n\nMNT :-\n";

    cout << "-------------------------------------------\nName\t\tPointer\n-------------------------------------------\n";

    outputFile << "\n\nMNT :-\n";

    outputFile << "-------------------------------------------\nName\t\tPointer\n-------------------------------------------\n";

    for(i = 0; i < mnt.size(); i ++){

        cout << mnt[i].macroName << "\t\t" << mnt[i].mdtPointer << "\n";

        outputFile << mnt[i].macroName << "\t\t" << mnt[i].mdtPointer << "\n";

    }

    InputFile.close();

    outputFile.close();    //close the file object

}

Output :

IC :-

-------------------------------------------

START 200

READ N1

READ N2

ADDS

SUBS

N1 DS 2

N2 DS 2

N3 DS 2

N4 DS 2

END

MDT :-

-------------------------------------------

Index           Line

-------------------------------------------

0               MOVER AREG ='1'

1               ADD AREG ='2'

2               MOVEM AREG X

3               MEND

4               MOVER BREG X

5               SUB BREG Y

6               MOVEM BREG X

7               WRITE Y

8               MEND

MNT :-

-------------------------------------------

Name            Pointer

-------------------------------------------

ADDS            0

SUBS            4

**Week 2** (Takes InputA2W2.txt as input)

Code :

#include <bits/stdc++.h>

using namespace std;

struct mntStruct{

    string macroName;

    int mdtPointer;

};

string makeSubStr(string str, int start, int end){

    string ans = "";

    for(int i = start; i < end; i ++){

        ans += str[i];

    }

    return ans;

}

string removeCommas (string str){

    int i;

    string commaReplaced = "";

    for(i = 0; i < str.length(); i ++){

        if(str[i] == ','){

            commaReplaced += " ";

        }

        else{

            commaReplaced += str[i];

        }

    }

    return commaReplaced;

}

string removeSpaces (string str){

    int i;

    string tabReplaced = "", spaceReplaced = " ";

    for(i = 0; i < str.length(); i ++){

        if(str[i] == '\t'){

            tabReplaced += " ";

        }

        else{

            tabReplaced += str[i];

        }

    }

    for(i = 0; i < tabReplaced.length(); i ++){

        if(tabReplaced[i] == ' '){

            if( spaceReplaced[spaceReplaced.length() - 1] != ' ' ){

                spaceReplaced += " ";

            }

        }

        else{

            spaceReplaced += tabReplaced[i];

        }

    }

    if(spaceReplaced[0] == ' '){

        spaceReplaced.erase (spaceReplaced.begin());

    }

    if(spaceReplaced[spaceReplaced.length() - 1] == ' '){

        spaceReplaced.erase (spaceReplaced.end() - 1);

    }

    // cout << spaceReplaced << " heyyy\n";

    return spaceReplaced;

}

string removeCommaSpaces (string str) {

    int i;

    string commaReplaced = "", tabReplaced = "", spaceReplaced = " ";

    for(i = 0; i < str.length(); i ++){

        if(str[i] == '\t'){

            tabReplaced += " ";

        }

        else{

            tabReplaced += str[i];

        }

    }

    for(i = 0; i < tabReplaced.length(); i ++){

        if(tabReplaced[i] == ','){

            commaReplaced += " ";

        }

        else{

            commaReplaced += tabReplaced[i];

        }

    }

    for(i = 0; i < commaReplaced.length(); i ++){

        if(commaReplaced[i] == ' '){

            if( spaceReplaced[spaceReplaced.length() - 1] != ' ' ){

                spaceReplaced += " ";

            }

        }

        else{

            spaceReplaced += commaReplaced[i];

        }

    }

    if(spaceReplaced[0] == ' '){

        spaceReplaced.erase (spaceReplaced.begin());

    }

    if(spaceReplaced[spaceReplaced.length() - 1] == ' '){

        spaceReplaced.erase (spaceReplaced.end() - 1);

    }

    // cout << spaceReplaced << " heyyy\n";

    return spaceReplaced;

}

vector<string> splitStringBySpace (string str){

    vector<string> split;

    int i, lastSpace = -1;

    for(i = 0; i < str.length(); i ++){

        if(str[i] == ' '){

            split.push\_back( makeSubStr(str, lastSpace + 1, i) );

            lastSpace = i;

        }

    }

    split.push\_back( makeSubStr(str, lastSpace + 1, i) );

    return split;

}

int handleMacro (

    int i,

    vector< string > inputLines,

    vector< vector<string> > inputLinesSplit,

    vector<mntStruct> &mnt,

    vector<string> &mdt

){

    int j, k;

    string toAdd;

    map <string, int>  m;

    mntStruct newMacro;

    newMacro.macroName = inputLinesSplit[i][1];

    newMacro.mdtPointer = mdt.size();

    mnt.push\_back(newMacro);

    mdt.push\_back( makeSubStr(inputLines[i], 6, inputLines[i].length()) );

    for(j = 2; j < inputLinesSplit[i].size(); j ++){

        toAdd = inputLinesSplit[i][j];

        for(k = 0; k < inputLinesSplit[i][j].length(); k ++){

            if(inputLinesSplit[i][j][k] == '='){

                toAdd = makeSubStr(inputLinesSplit[i][j], 0, k);

                break;

            }

        }

        m[toAdd] = j - 1;

    }

    while(inputLinesSplit[i][0] != "MEND"){

        i += 1;

        toAdd = inputLinesSplit[i][0] + " ";

        // mdt.push\_back( inputLinesSplit[i][0] );

        for(j = 1; j < inputLinesSplit[i].size(); j ++){

            if(m[ inputLinesSplit[i][j] ] == 0){

                toAdd += inputLinesSplit[i][j];

            }

            else{

                toAdd += "#" + to\_string(m[ inputLinesSplit[i][j] ] );

            }

            if(j != inputLinesSplit[i].size() - 1){

                toAdd += ", ";

            }

        }

        mdt.push\_back( toAdd );

    }

    return i;

}

int main(){

    vector< vector<string> > inputLinesSplit;

    vector< string > inputLines;

    string line;

    ifstream InputFile("InputA2W2.txt");

    if(InputFile.is\_open()){

        while (getline(InputFile, line)) {

            // cout << line << "\n";

            inputLines.push\_back( removeSpaces(line) );

            inputLinesSplit.push\_back( splitStringBySpace( removeSpaces( removeCommas(line) ) ) );

        }

    }

    else{

        cout << "Error in opening input file.\n";

    }

    int i, j;

    vector<string> mdt;

    vector<string> ic;

    vector<mntStruct> mnt;

    for(i = 0; i < inputLinesSplit.size(); i ++){

        // cout << inputLines[i] << "\n";

        if(inputLinesSplit[i][0] == "MACRO"){

            i = handleMacro(i, inputLines, inputLinesSplit, mnt, mdt);

        }

        else{

            ic.push\_back(inputLines[i]);

        }

    }

    fstream outputFile;

    outputFile.open("OutputA2W2.txt", ios::out);

    cout << "\nIC :-\n-------------------------------------------\n";

    outputFile << "\nIC :-\n-------------------------------------------\n";

    for(i = 0; i < ic.size(); i ++){

        cout << ic[i] << "\n";

        outputFile << ic[i] << "\n";

    }

    // cout << i;

    cout << "\n\nMDT :-\n";

    cout << "-------------------------------------------\nIndex\t\tLine\n-------------------------------------------\n";

    outputFile << "\n\nMDT :-\n";

    outputFile << "-------------------------------------------\nIndex\t\tLine\n-------------------------------------------\n";

    for(i = 0; i < mdt.size(); i ++){

        cout << i << "\t\t" << mdt[i] << "\n";

        outputFile << i << "\t\t" << mdt[i] << "\n";

    }

    // cout << "\nlen - " << mnt.size();

    cout << "\n\nMNT :-\n";

    cout << "-------------------------------------------\nName\t\tPointer\n-------------------------------------------\n";

    outputFile << "\n\nMNT :-\n";

    outputFile << "-------------------------------------------\nName\t\tPointer\n-------------------------------------------\n";

    for(i = 0; i < mnt.size(); i ++){

        cout << mnt[i].macroName << "\t\t" << mnt[i].mdtPointer << "\n";

        outputFile << mnt[i].macroName << "\t\t" << mnt[i].mdtPointer << "\n";

    }

    InputFile.close();

    outputFile.close();    //close the file object

}

Output :

IC :-

-------------------------------------------

START 200

READ N1

READ N2

ADDS1 N1,N2

SUBS1 N1,N2

READ N1

READ N2

INCR N3,N4,DREG

STOP

N1 DS 2

N2 DS 2

N3 DS 2

N4 DS 2

END

MDT :-

-------------------------------------------

Index           Line

-------------------------------------------

0               INCR &A, &B, &REG=CREG

1               MOVER #3, #1

2               ADDS1 #1, #2

3               MOVEM #3, #1

4               MEND

5               ADDS1 &C, &D

6               MOVER AREG, #1

7               ADD AREG, #2

8               MOVEM AREG, #2

9               WRITE #2

10              MEND

11              SUBS1 &C1, &D1

12              MOVER BREG, #1

13              SUB BREG, #2

14              MOVEM BREG, #2

15              WRITE #2

16              MEND

MNT :-

-------------------------------------------

Name            Pointer

-------------------------------------------

INCR            0

ADDS1           5

SUBS1           11

**Week 3** (Takes InputA2W3.txt as input)

Code :

#include <bits/stdc++.h>

using namespace std;

struct mntStruct{

    string macroName;

    int mdtPointer;

    int noOfParameters;

    map <string, int> parameters;

};

string makeSubStr(string str, int start, int end){

    string ans = "";

    for(int i = start; i < end; i ++){

        ans += str[i];

    }

    return ans;

}

string removeCommas (string str){

    int i;

    string commaReplaced = "";

    for(i = 0; i < str.length(); i ++){

        if(str[i] == ','){

            commaReplaced += " ";

        }

        else{

            commaReplaced += str[i];

        }

    }

    return commaReplaced;

}

string removeSpaces (string str){

    int i;

    string tabReplaced = "", spaceReplaced = " ";

    for(i = 0; i < str.length(); i ++){

        if(str[i] == '\t'){

            tabReplaced += " ";

        }

        else{

            tabReplaced += str[i];

        }

    }

    for(i = 0; i < tabReplaced.length(); i ++){

        if(tabReplaced[i] == ' '){

            if( spaceReplaced[spaceReplaced.length() - 1] != ' ' ){

                spaceReplaced += " ";

            }

        }

        else{

            spaceReplaced += tabReplaced[i];

        }

    }

    if(spaceReplaced[0] == ' '){

        spaceReplaced.erase (spaceReplaced.begin());

    }

    if(spaceReplaced[spaceReplaced.length() - 1] == ' '){

        spaceReplaced.erase (spaceReplaced.end() - 1);

    }

    // cout << spaceReplaced << " heyyy\n";

    return spaceReplaced;

}

vector<string> splitStringBySpace (string str){

    vector<string> split;

    int i, lastSpace = -1;

    for(i = 0; i < str.length(); i ++){

        if(str[i] == ' '){

            split.push\_back( makeSubStr(str, lastSpace + 1, i) );

            lastSpace = i;

        }

    }

    split.push\_back( makeSubStr(str, lastSpace + 1, i) );

    return split;

}

int isMacroPresent(vector<mntStruct> &mnt, string check){

    int i;

    for(i = 0; i < mnt.size(); i ++){

        if(mnt[i].macroName == check){

            return i;

        }

    }

    return -1;

}

void handleCallMacro(

    int i,

    vector< string > inputLines,

    vector< vector<string> > inputLinesSplit,

    vector<mntStruct> &mnt,

    vector<string> &mdt,

    int macroMntIndex

){

    int j = mnt[macroMntIndex].mdtPointer, k;

    vector<string> splitString;

    string toAdd;

    while(removeSpaces(mdt[j]) != "MEND"){

        splitString = splitStringBySpace( removeSpaces(removeCommas(mdt[j])) );

        toAdd = splitString[0] + " ";

        for(k = 1; k < splitString.size(); k ++){

            if( splitString[k][0] == '#'){

                // cout << splitString[k][1] << " " << inputLinesSplit[i][1] <<  " -----\n";

                string s(1, splitString[k][1]);

                toAdd += inputLinesSplit[i][ stoi(s) ];

            }

            else{

                toAdd += splitString[k];

            }

            if(k != splitString.size() - 1){

                toAdd += ", ";

            }

        }

        j += 1;

        mdt.push\_back(toAdd);

    }

}

int handleAddMacro (

    int i,

    vector< string > inputLines,

    vector< vector<string> > inputLinesSplit,

    vector<mntStruct> &mnt,

    vector<string> &mdt

){

    int j, k, checkMacro;

    string toAdd;

    map <string, int>  m;

    mntStruct newMacro;

    newMacro.macroName = inputLinesSplit[i][1];

    newMacro.mdtPointer = mdt.size();

    newMacro.noOfParameters = inputLinesSplit[i].size() - 2;

    // mdt.push\_back( makeSubStr(inputLines[i], 6, inputLines[i].length()) );

    for(j = 2; j < inputLinesSplit[i].size(); j ++){

        toAdd = inputLinesSplit[i][j];

        for(k = 0; k < inputLinesSplit[i][j].length(); k ++){

            if(inputLinesSplit[i][j][k] == '='){

                toAdd = makeSubStr(inputLinesSplit[i][j], 0, k);

                break;

            }

        }

        m[toAdd] = j - 1;

    }

    newMacro.parameters = m;

    mnt.push\_back(newMacro);

    while(inputLinesSplit[i][0] != "MEND"){

        i += 1;

        checkMacro = isMacroPresent(mnt, inputLinesSplit[i][0]);

        if(checkMacro != -1 ){

            handleCallMacro(i, inputLines, inputLinesSplit, mnt, mdt, checkMacro);

            continue;

        }

        toAdd = inputLinesSplit[i][0] + " ";

        // mdt.push\_back( inputLinesSplit[i][0] );

        for(j = 1; j < inputLinesSplit[i].size(); j ++){

            if(m[ inputLinesSplit[i][j] ] == 0){

                toAdd += inputLinesSplit[i][j];

            }

            else{

                toAdd += "#" + to\_string(m[ inputLinesSplit[i][j] ]);

            }

            if(j != inputLinesSplit[i].size() - 1){

                toAdd += ", ";

            }

        }

        mdt.push\_back( toAdd );

    }

    return i;

}

int main(){

    vector< vector<string> > inputLinesSplit;

    vector< string > inputLines;

    string line;

    ifstream InputFile("InputA2W3.txt");

    if(InputFile.is\_open()){

        while (getline(InputFile, line)) {

            // cout << line << "\n";

            inputLines.push\_back( removeSpaces(line) );

            inputLinesSplit.push\_back( splitStringBySpace( removeSpaces( removeCommas(line) ) ) );

        }

    }

    else{

        cout << "Error in opening input file.\n";

    }

    int i, j;

    vector<string> mdt;

    vector<string> ic;

    vector<mntStruct> mnt;

    for(i = 0; i < inputLinesSplit.size(); i ++){

        if(inputLinesSplit[i][0] == "MACRO"){

            i = handleAddMacro(i, inputLines, inputLinesSplit, mnt, mdt);

        }

        else{

            ic.push\_back(inputLines[i]);

        }

    }

    fstream outputFile;

    outputFile.open("OutputA2W3.txt", ios::out);

    cout << "\nIC :-\n-------------------------------------------\n";

    outputFile << "\nIC :-\n-------------------------------------------\n";

    for(i = 0; i < ic.size(); i ++){

        cout << ic[i] << "\n";

        outputFile << ic[i] << "\n";

    }

    // cout << i;

    cout << "\n\nMDT :-\n";

    cout << "-------------------------------------------\nIndex\t\tLine\n-------------------------------------------\n";

    outputFile << "\n\nMDT :-\n";

    outputFile << "-------------------------------------------\nIndex\t\tLine\n-------------------------------------------\n";

    for(i = 0; i < mdt.size(); i ++){

        cout << i << "\t\t" << mdt[i] << "\n";

        outputFile << i << "\t\t" << mdt[i] << "\n";

    }

    // cout << "\nlen - " << mnt.size();

    cout << "\n\nMNT :-\n";

    cout << "-------------------------------------------\nName\t\tPointer\t\tParameters\n-------------------------------------------\n";

    outputFile << "\n\nMNT :-\n";

    outputFile << "-------------------------------------------\nName\t\tPointer\t\tParameters\n-------------------------------------------\n";

    for(i = 0; i < mnt.size(); i ++){

        cout << mnt[i].macroName << "\t\t" << mnt[i].mdtPointer << "\t\t" << mnt[i].noOfParameters << "\n";

        outputFile << mnt[i].macroName << "\t\t" << mnt[i].mdtPointer << "\t\t" << mnt[i].noOfParameters << "\n";

    }

    cout << "\n\nMNT Variables :-\n-------------------------------------------\n";

    outputFile << "\n\nMNT Variables :-\n-------------------------------------------\n";

    for(i = 0; i < mnt.size(); i ++){

        if(mnt[i].noOfParameters == 0){

            continue;

        }

        cout << "\n";

        cout << mnt[i].macroName << " :-\n";

        cout << "--------------------------\n";

        cout << "Name\t\tValue\n";

        cout << "--------------------------\n";

        outputFile << "\n";

        outputFile << mnt[i].macroName << " :-\n";

        outputFile << "--------------------------\n";

        outputFile << "Name\t\tValue\n";

        outputFile << "--------------------------\n";

        for (auto i : mnt[i].parameters){

            cout << i.first << "\t\t#" << i.second << "\n";

            outputFile << i.first << "\t\t#" << i.second << "\n";

        }

        cout << "\n";

        outputFile << "\n";

    }

    InputFile.close();

    outputFile.close();    //close the file object

}

Output :

IC :-

-------------------------------------------

LOAD A

STORE B

ABC

ADD5 D1, D2, D3

END

MDT :-

-------------------------------------------

Index       Line

-------------------------------------------

0       LOAD p

1       SUB q

2       MEND

3       LOAD X

4       STORE #1

5       MEND

6       STORE #2

7       LOAD X

8       STORE 5

9       LOAD X

10      STORE 10

11      LOAD #1

12      LOAD #3

13      MEND

MNT :-

-------------------------------------------

Name        Pointer     Parameters

-------------------------------------------

ABC     0       0

ADD1        3       1

ADD5        6       3

MNT Variables :-

-------------------------------------------

ADD1 :-

--------------------------

Name        Value

--------------------------

ARG     #1

ADD5 :-

--------------------------

Name        Value

--------------------------

A1      #1

A2      #2

A3      #3

Thank You !