**4.MacroPass2**

import java.io.BufferedReader;

import java.io.FileReader;

import java.io.FileWriter;

import java.util.HashMap;

import java.util.Vector;

class MNTEntry {

String name;

int pp,kp,mdtp,kpdtp;

public MNTEntry(String name, int pp, int kp, int mdtp, int kpdtp) {

super();

this.name = name;

this.pp = pp;

this.kp = kp;

this.mdtp = mdtp;

this.kpdtp = kpdtp;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public int getPp() {

return pp;

}

public void setPp(int pp) {

this.pp = pp;

}

public int getKp() {

return kp;

}

public void setKp(int kp) {

this.kp = kp;

}

public int getMdtp() {

return mdtp;

}

public void setMdtp(int mdtp) {

this.mdtp = mdtp;

}

public int getKpdtp() {

return kpdtp;

}

public void setKpdtp(int kpdtp) {

this.kpdtp = kpdtp;

}

}

public class MacroPass2 {

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

BufferedReader irb=new BufferedReader(new FileReader("intermediate.txt"));

BufferedReader mdtb=new BufferedReader(new FileReader("mdt.txt"));

BufferedReader kpdtb=new BufferedReader(new FileReader("kpdt.txt"));

BufferedReader mntb=new BufferedReader(new FileReader("mnt.txt"));

FileWriter fr=new FileWriter("pass2.txt");

HashMap<String, MNTEntry> mnt=new HashMap<>();

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

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

Vector<String>mdt=new Vector<String>();

Vector<String>kpdt=new Vector<String>();

int pp,kp,mdtp,kpdtp,paramNo;

String line;

while((line=mdtb.readLine())!=null)

{

mdt.addElement(line);

}

while((line=kpdtb.readLine())!=null)

{

kpdt.addElement(line);

}

while((line=mntb.readLine())!=null)

{

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

mnt.put(parts[0], new MNTEntry(parts[0], Integer.parseInt(parts[1]), Integer.parseInt(parts[2]), Integer.parseInt(parts[3]), Integer.parseInt(parts[4])));

}

while((line=irb.readLine())!=null)

{

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

if(mnt.containsKey(parts[0]))

{

pp=mnt.get(parts[0]).getPp();

kp=mnt.get(parts[0]).getKp();

kpdtp=mnt.get(parts[0]).getKpdtp();

mdtp=mnt.get(parts[0]).getMdtp();

paramNo=1;

for(int i=0;i<pp;i++)

{

parts[paramNo]=parts[paramNo].replace(",", "");

aptab.put(paramNo, parts[paramNo]);

aptabInverse.put(parts[paramNo], paramNo);

paramNo++;

}

int j=kpdtp-1;

for(int i=0;i<kp;i++)

{

String temp[]=kpdt.get(j).split("\t");

aptab.put(paramNo,temp[1]);

aptabInverse.put(temp[0],paramNo);

j++;

paramNo++;

}

for(int i=pp+1;i<parts.length;i++)

{

parts[i]=parts[i].replace(",", "");

String splits[]=parts[i].split("=");

String name=splits[0].replaceAll("&", "");

aptab.put(aptabInverse.get(name),splits[1]);

}

int i=mdtp-1;

while(!mdt.get(i).equalsIgnoreCase("MEND"))

{

String splits[]=mdt.get(i).split("\\s+");

fr.write("+");

for(int k=0;k<splits.length;k++)

{

if(splits[k].contains("(P,"))

{

splits[k]=splits[k].replaceAll("[^0-9]", "");

String value=aptab.get(Integer.parseInt(splits[k]));

fr.write(value+"\t");

}

else

{

fr.write(splits[k]+"\t");

}

}

fr.write("\n");

i++;

}

aptab.clear();

aptabInverse.clear();

}

else

{

fr.write(line+"\n");

}

}

fr.close();

mntb.close();

mdtb.close();

kpdtb.close();

irb.close();

}

}