1. 操作手冊: 本程式是以java去寫的。此程式在執行前需有一個名為input的txt檔且要與程式在同個路徑裡(txt格式:第一行為”<x>”;x為變數個數，只能是8到10，第二行為”[“，第三行開始，每行為輸入要寫的數字，最後在換一行打”]”代表結束，don’t care的地方則在”]”的下一行先打”(“，下一行打輸入的數字，所有的數字皆在同個括弧內，最後打個”)”input.txt格式即完成)，在此txt檔內輸入要變換成QM的東西，執行程式後，便會開始簡化並寫成minimum SOP，自動創立一個output.txt檔並將結果寫進裡面，程式即宣告結束。執行方式:打開cmd，輸入javac logic1.java編譯，之後輸入java.logic1即可執行。
2. 程式撰寫流程:

基本上我分成下列步驟:

1. 將txt檔的東西分別拆開，以利之後運算:

先將檔案的內容抓出來存成String，共三個String，一個存binary(前端會補零)，另一個存decimal(上述皆有包含don’t care)，最後一個則把don’t care抓出來存10進位數字。

1. 將上述存的前兩個String排列好，大概方法如下:

判斷binary的String有幾個1，設兩個arraylist，按照1的數量依序將該binary和decimal分別加進自己的arraylist裡。

1. 第一次寫檔關於binary結合的部分(輸出非圖表的部分):

先將binary做區間，分類方式像是只有一個1的有幾個，兩個1的有幾個，以此類推。之後開始做結合，每一個區間只能和下一個區間做結合，且兩區間的1的個數只能相差1，第1區間和第2區間做完則換第2區間和第3區間做結合，以此類推。每結合一次，則在原有的依序binary和decimal的arraylist將結合的值add進最後面(關於decimal的arraylist可能會存入”,”，像是1,2 10,11)，之後新增一個新的區間在原本設的區間裡，而因為把結合的值丟進原有的arraylist裡，arraylist的長度增加了，所以會繼續執行結合過程讓結合過的區間再互相結合，重複上述步驟，一直到再也無法結合為止。

1. 微調步驟3的寫檔格式:

在步驟1中有存don’t care，比對執行完步驟3的binary，相等的話則在binary字串的前面加個”d “。而在步驟3中在結合時會稍微記一下是誰跟誰結合，等到步驟3執行完後，會將有結合的binary字串的前面加個”v “，最後再看看有沒有binary字串互相重複的打”x “，之後配合格式將binary和decimal 的arraylist配合格是輸出即可。

1. 選取做圖用的binary,decimal:

將上面未作任何標記的binary ,decimal arraylist選取出來，在最後輸出時會用到，將選取的binary轉成字母(我是用function)，將選取的decimal轉成只有一個數字(無逗號且重複的數字不會加進去,don’t care的數字也不會加進去)並依序排列。

1. 做圖:

先按照格式輸出，之後將步驟5做完的decimal依序印上上面一橫排，字母則印到左邊一直排，再將字母所對應的數字(prime\_implicant)以”x”來表示在表格裡，

即完成圖表輸出檔。

1. 找最佳解，最終輸出:

設一個String[][]將剛剛的圖表copy進來(不存字母和數字)，裡面只存prime\_implicant部分，有的話存”x”(即上個步驟的”x”)，沒有的話存” “。首先，先找essential\_prime\_implicant，若直排只有一個”x”，存取此橫排對應字母並將這一橫排的所有”x”改成”1”，而此橫排所有”x”的上下延伸遇到”x”的話也改成”1”(當作使用過)，以此類推。找完時，若此String[][]裡還存有”x”，則選取橫排最多”x”的當作選擇，存取此橫排對應字母並將這一橫排的所有”x”改成”1”， 而此橫排所有”x”的上下延伸遇到”x”的話也改成”1”(當作使用過)，以此類推，直到找完。最後將所有選到的字母也就是minimum SOP的最後solution印出來，程式執行結束。

1. Source code:

import java.io.\*;

import java.util.\*;

public class logic1{

public static String input(){ //read the input.txt

String formula = "";

try {

InputStreamReader reader = new InputStreamReader(new FileInputStream("input.txt"));

BufferedReader br = new BufferedReader(reader);

String line = " ";

while ((line= br.readLine()) != null) {

formula += line;

formula += " ";

}

} catch (FileNotFoundException e) {

e.printStackTrace();

} catch (IOException e) {

e.printStackTrace();

}

return formula;

}

public static String[] array(int a){ //create k-map

String[] abcd;

if(a==8){ //8 represent 8 variables

abcd = new String[8];

abcd[0] = "a";

abcd[1] = "b";

abcd[2] = "c";

abcd[3] = "d";

abcd[4] = "e";

abcd[5] = "f";

abcd[6] = "g";

abcd[7] = "h";

}

else{

if(a==9){ //9 represent 9 variables

abcd = new String[9];

abcd[0] = "a";

abcd[1] = "b";

abcd[2] = "c";

abcd[3] = "d";

abcd[4] = "e";

abcd[5] = "f";

abcd[6] = "g";

abcd[7] = "h";

abcd[8] = "i";

}

else{ // represent 10 variables

abcd = new String[10];

abcd[0] = "a";

abcd[1] = "b";

abcd[2] = "c";

abcd[3] = "d";

abcd[4] = "e";

abcd[5] = "f";

abcd[6] = "g";

abcd[7] = "h";

abcd[8] = "i";

abcd[9] = "j";

}

}

return abcd;

}

public static void main(String args[]){

try{ //write things in the output.txt

File file = new File("output.txt");

BufferedWriter fw = new BufferedWriter(new OutputStreamWriter(new FileOutputStream(file, true), "UTF-8"));

int column =1;

int block=0;

String []voc;

String[] binary\_number=new String[200]; //store binary number (include don't care)

String[] decimal\_number=new String[200]; //store decimal number (include don't care)

int count\_number=0; // the count of number (include don't care)

String[] notcare=new String[30]; //store don't care decimal number

int count\_notcare=0;

int have\_one=0;

String s,deci;

String binary;

String formula = input(); // read the file

String[] one = formula.split(" ");

String vary=one[0].substring(1,(one[0].length()-1));

int variable = Integer.parseInt(vary);

vary="%0"+vary;

vary+="d";

voc=array(variable);

for(int i=0;i<one.length;i++){ //read the file catch don't care and decimal

if(one[i].charAt(0)!='<'&&one[i].charAt(0)!='['&&one[i].charAt(0)!=']'&&one[i].charAt(0)!=')'){

if(one[i].charAt(0)!='('&&count\_notcare==0){

decimal\_number[count\_number]=one[i];

count\_number++;

}

else{

if(count\_notcare==0){

i++;

}

String dontcare=one[i];

decimal\_number[count\_number]=dontcare;

count\_number++;

notcare[count\_notcare]=dontcare;

count\_notcare++;

}

}

}

int []have\_one\_array=new int[count\_number];

for(int i=0;i<count\_number;i++){ //transfor decimal to binary

binary = Integer.toBinaryString(Integer.parseInt(decimal\_number[i]));

binary\_number[i]=String.format(vary,Integer.parseInt(binary));

}

for(int i=0;i<count\_number;i++){ // catch 'one' number in binary

for(int j=0;j<variable;j++){

if(binary\_number[i].charAt(j)=='1'){

have\_one++;

}

}

have\_one\_array[i]=have\_one;

have\_one=0;

}

ArrayList<Integer> seperate\_number = new ArrayList<Integer>();//each area has x number

ArrayList<Integer> seperate\_one = new ArrayList<Integer>(); //each area's number has x "1"

ArrayList<String> order\_binary\_number = new ArrayList<String>(); //let binary\_number in order

ArrayList<String> order\_decimal\_number = new ArrayList<String>(); //let decimal\_number in order

ArrayList<Integer> beused = new ArrayList<Integer>(); //choose number which is be used

for(int i=0;i<=variable;i++){ //let binary and decimal in order

for(int j=0;j<count\_number;j++){

if(have\_one\_array[j]==i){

order\_binary\_number.add(binary\_number[j]);

order\_decimal\_number.add(decimal\_number[j]);

block++;

}

if(j==(count\_number-1)&&block!=0){

seperate\_number.add(block); // the same number of 'one' has "block"

seperate\_one.add(i); // there has i "one"

block=0;

}

}

}

s="";

deci="";

int different =0;

int precise\_different=1;

int compute1=0;

int compute2=0;

int compute3=0;

int truth\_have\_one=0;

int beused\_count;

int remove=0;

have\_one=0;

int count =0;

// try to fulfill the column write file

if(seperate\_number.size()>1){ // distinguish the seperate more than 1

for(int i=0;i<(seperate\_number.size()-1);i++){ //the seperate number

compute2+=seperate\_number.get(i);

if(seperate\_one.get(i)+1==seperate\_one.get(i+1)){ //if the seperate are continuous

for(int j=0;j<seperate\_number.get(i);j++){ // choose first area

for(int k=0;k<seperate\_number.get(i+1);k++){ // choose second area

s=order\_binary\_number.get(compute1+j);

deci=order\_decimal\_number.get(compute1+j);

String[] s\_split = s.split("");

s="";

for(int l=0;l<variable;l++){

if(order\_binary\_number.get(j+compute1).charAt(l)!=order\_binary\_number.get(k+compute2).charAt(l)){

s\_split[l]="-";

different++;

}

}

for(int l=0;l<s\_split.length;l++){

s+=s\_split[l];

}

if(different==1){

beused.add(j+compute1); //choose which is be used

beused.add(k+compute2); //choose which is be used

beused\_count=beused.size();

for(int l=0;l<(beused\_count-2);l++){ // calculate which binary is used

if(beused.get(l)==j+compute1){

beused.remove(beused\_count-2); //remove repeat number in beused

remove++;

}

if(beused.get(l)==k+compute2){

if(remove==0){

beused.remove(beused\_count-1);

remove++;

}

else{

beused.remove(beused\_count-2);

remove++;

}

}

}

remove=0;

order\_binary\_number.add(s);

order\_decimal\_number.add(deci+","+order\_decimal\_number.get(k+compute2));

count++;

for(int l=0;l<s.length();l++){

if(s.charAt(l)=='1'){

have\_one++;

}

}

if(have\_one!=truth\_have\_one){

truth\_have\_one=have\_one;

}

}

deci="";

s="";

different=0;

have\_one=0;

}

}

if(count!=0){

seperate\_number.add(count);

seperate\_one.add(truth\_have\_one);

}

count=0;

}

compute1+=seperate\_number.get(i);

}

}

s="";

ArrayList<String> choosed\_binary = new ArrayList<String>(); //select which binary is be choosed

for(int i=0;i<count\_notcare;i++){ //write don't care(d) in writefile

for(int j=0;j<order\_decimal\_number.size();j++){

if(notcare[i].equals(order\_decimal\_number.get(j))){

order\_binary\_number.set(j,("d "+order\_binary\_number.get(j)));

}

}

}

for(int i=0;i<order\_binary\_number.size();i++){ //write be used(v) in writefile

for(int j=0;j<beused.size();j++){

if(i==beused.get(j)){

if(order\_binary\_number.get(i).charAt(0)=='d'){

continue;

}

order\_binary\_number.set(i,("v "+order\_binary\_number.get(i)));

}

}

}

ArrayList<String> choosed\_multiple\_decimal = new ArrayList<String>(); //select which decimal is be choosed(ex:1 2,18 5,6 ...)

for(int i=0;i<order\_binary\_number.size();i++){ //write repeat(x) and used( ) in writefile

for(int j=0;j<order\_binary\_number.size();j++){

if(i==j){

continue;

}

if(order\_binary\_number.get(i).equals(order\_binary\_number.get(j))){

if(order\_binary\_number.get(j).charAt(0)=='x'||order\_binary\_number.get(j).charAt(0)=='v'){

continue;

}

order\_binary\_number.set(j,("x "+order\_binary\_number.get(i)));

}

}

if(order\_binary\_number.get(i).charAt(0)=='0' ||order\_binary\_number.get(i).charAt(0)=='1'||order\_binary\_number.get(i).charAt(0)=='-'){ //if this binary need to print,then write " "

choosed\_multiple\_decimal.add(order\_decimal\_number.get(i));

choosed\_binary.add(order\_binary\_number.get(i)); // choose the used binary

order\_binary\_number.set(i,(" "+order\_binary\_number.get(i)));

s+=order\_decimal\_number.get(i);

s+=",";

}

}

truth\_have\_one=-1;

have\_one=0;

count=0;

different=0;

fw.append("Column "+column+"\n=====================\n");

column++;

for(int i=0;i<order\_binary\_number.size();i++){ // write the file

for(int j=0;j<variable;j++){

if(order\_binary\_number.get(i).charAt(j+2)=='1'){

have\_one++;

}

if(order\_binary\_number.get(i).charAt(j+2)=='-'){

count++;

}

}

if(different!=count){

fw.append("=====================\n"+"Column "+column+"\n=====================\n");

different=count;

truth\_have\_one=have\_one;

column++;

}

if(i==0){

truth\_have\_one=have\_one;

}

if(have\_one!=truth\_have\_one){

fw.append("=====================\n");

truth\_have\_one=have\_one;

}

fw.append(order\_binary\_number.get(i)+" : "+order\_decimal\_number.get(i)+"\n");

have\_one=0;

count=0;

}

String[] s\_split=s.split(",");

ArrayList<String> choosed\_decimal = new ArrayList<String>(); //select which decimal is be used in image(ex:1 2 6 ...)

boolean can\_be\_choose = true;

for(int i=0;i<s\_split.length;i++){

for(int j=0;j<count\_notcare;j++){

if(notcare[j].equals(s\_split[i])){

can\_be\_choose=false;

}

}

if(can\_be\_choose){

choosed\_decimal.add(s\_split[i]);

}

can\_be\_choose=true;

}

for(int j=0;j<choosed\_decimal.size();j++){ // choose decimal be used in image

for(int i=choosed\_decimal.size()-1;i>-1;i--){

if(i==j){

continue;

}

if(choosed\_decimal.get(j).equals(choosed\_decimal.get(i))){

choosed\_decimal.remove(i);

}

}

}

int first\_deci=0;

int second\_deci=0;

String change;

for(int i=0;i<choosed\_decimal.size();i++){ // let choosed\_decimal number be in order

for(int j=0;j<choosed\_decimal.size();j++){

if(i>=j){

continue;

}

first\_deci = Integer.parseInt(choosed\_decimal.get(i));

second\_deci = Integer.parseInt(choosed\_decimal.get(j));

if(first\_deci>second\_deci){

change=choosed\_decimal.get(i);

choosed\_decimal.set(i,choosed\_decimal.get(j));

choosed\_decimal.set(j,change);

}

}

}

ArrayList<String> choosed\_voc = new ArrayList<String>(); //tranfor choosed\_binary into vocbulary

boolean can\_add=true;

for(int i=0;i<choosed\_binary.size();i++){ //change binary into vocbulary

for(int j=0;j<variable;j++){

if(choosed\_binary.get(i).charAt(j)=='-'){

continue;

}

if(choosed\_binary.get(i).charAt(j)=='1'){

if(can\_add){

choosed\_voc.add(voc[j]);

can\_add=false;

}

else{

choosed\_voc.set(i,choosed\_voc.get(i)+voc[j]);

}

}

if(choosed\_binary.get(i).charAt(j)=='0'){

if(can\_add){

choosed\_voc.add(voc[j]+"'");

can\_add=false;

}

else{

choosed\_voc.set(i,choosed\_voc.get(i)+voc[j]+"'");

}

}

}

can\_add=true;

}

s="";

fw.append("\n");

String[][] image = new String[choosed\_binary.size()][choosed\_decimal.size()]; //this is used to the second output--image

for(int i=0;i<image.length;i++){ // initialize the image

for(int j=0;j<image[0].length;j++){

image[i][j]=" ";

}

}

for(int i=0;i<choosed\_binary.size();i++){ //input information in image

s=choosed\_multiple\_decimal.get(i);

s\_split=s.split(",");

for(int j=0;j<choosed\_decimal.size();j++){

for(int k=0;k<s\_split.length;k++){

if(s\_split[k].equals(choosed\_decimal.get(j))){

image[i][j]="x";

}

}

}

}

//begin to draw the image

for(int i=0;i<(22+choosed\_decimal.size()\*6);i++){

fw.append("=");

}

fw.append("\nResult\n");

for(int i=0;i<(22+choosed\_decimal.size()\*6);i++){

fw.append("=");

}

fw.append("\n |");

for(int i=0;i<choosed\_decimal.size();i++){

fw.append(String.format("%5s|",choosed\_decimal.get(i)));

/\*System.out.printf("%5s|",choosed\_decimal.get(i));\*/

}

fw.append("\n");

for(int i=0;i<(22+choosed\_decimal.size()\*6);i++){

fw.append("-");

}

fw.append("\n");

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

fw.append(String.format("%-21s",choosed\_voc.get(i)));

/\*System.out.printf("%-21s",choosed\_voc.get(i));\*/

fw.append("|");

for(int j=0;j<image[0].length;j++){

fw.append(String.format("%5s|",image[i][j]));

/\*System.out.printf("%5s|",image[i][j]);\*/

}

fw.append("\n");

for(int j=0;j<(22+choosed\_decimal.size()\*6);j++){

fw.append("-");

}

fw.append("\n");

}

int[] prime\_implicant=new int[choosed\_decimal.size()]; //understand each row's number of prime\_implicant

have\_one=0; //this choose prime\_implicant

for(int i=0;i<image[0].length;i++){

for(int j=0;j<image.length;j++){

if(image[j][i].equals("x")){

have\_one++;

}

}

prime\_implicant[i]=have\_one;

have\_one=0;

}

ArrayList<String> final\_choosed\_voc = new ArrayList<String>(); //the final result

can\_add=true;

can\_be\_choose=true;

count=-1;

compute1=0;

int[] voc\_prime\_implicant\_number=new int[image.length];

while(can\_add){ //begin to find the best solution

if(can\_be\_choose){

for(int i=0;i<prime\_implicant.length;i++){

if(prime\_implicant[i]==1){ // find essential prime implicant

for(int j=0;j<image.length;j++){

if(image[j][i].equals("x")){

final\_choosed\_voc.add(choosed\_voc.get(j));

for(int k=0;k<choosed\_decimal.size();k++){

if(image[j][k].equals("x")){

for(int l=0;l<image.length;l++){

if(image[l][k].equals("x")){

image[l][k]="1"; //1 means this is be used

}

}

}

}

}

}

}

}

can\_be\_choose=false;

}

for(int i=0;i<image.length;i++){ //initialize

voc\_prime\_implicant\_number[i]=0;

}

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

for(int j=0;j<image[0].length;j++){

if(image[i][j].equals("x")){

voc\_prime\_implicant\_number[i]++;

have\_one++; // this have\_one is distinguish wheather the "x" is all be choosed

}

}

}

compute1=voc\_prime\_implicant\_number[0];

count=0;

if(have\_one==0){

can\_add=false;

}

else{

for(int i=0;i<voc\_prime\_implicant\_number.length;i++){

/\*System.out.print(i);\*/

if(voc\_prime\_implicant\_number[i]>compute1){

/\*System.out.println("compute1:"+compute1+"voc\_prime\_implicant\_number["+i+"]"+voc\_prime\_implicant\_number[i]);\*/

compute1=voc\_prime\_implicant\_number[i];

count=i;

}

}

/\*System.out.println("compute1:"+compute1+"count"+count);

System.out.println("=========================================================================================");\*/

final\_choosed\_voc.add(choosed\_voc.get(count));

for(int i=0;i<image[0].length;i++){

if(image[count][i].equals("x")){

for(int j=0;j<image.length;j++){

if(image[j][i].equals("x")){

image[j][i]="1"; //1 means this is be used

}

}

}

}

}

have\_one=0;

}

fw.append("F(");

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

fw.append(voc[i]);

if(i!=voc.length-1){

fw.append(",");

}

else{

fw.append(")= ");

}

}

for(int i=0;i<final\_choosed\_voc.size();i++){ // initialize the image

fw.append(final\_choosed\_voc.get(i));

if(i!=final\_choosed\_voc.size()-1){

fw.append(" + ");

}

}

fw.flush();

fw.close();

}catch (IOException e) {

e.printStackTrace();

}

}

}