TUGAS

Logika Matematika

Program Fungsi Boolean Tingkat 3 dan Peta Karnaugh

*Disusun untuk memenuhi salah satu tugas mata kuliah*

*Logika Matematika*



Oleh :

Ahmad Shofi\_10118273

Mirza MY Humayung\_10118277

Ginar Cut Baktiyanesa\_10118289

Kelas : IF - 7

**TEKNIK INFORMATIKA**

**FAKULTAS TEKNIK DAN ILMU KOMPUTER**

**UNIVERSITAS KOMPUTER INDONESIA**

**2019**

**KATA PENGANTAR**

Asslamualaikum wr.wb.

Bismillahirrahmanirrahim

            Segala puji bagi allah SWT yang telah memberikan banyak sekali nikmat kepada kita semua yang atas nikmatnya itu sehinga saya bisa menyelesaikan tugas makalah yang berjudul “Program Fungsi Boolean Tingkat 3 dan Peta Karnaugh” untuk memenuhi salah satu tugas mata kuliah Logika Matematika. Shalawat seiring salam marilah kita limpah curahkan  kepada junjungan kita yaitu nabi besar Muhammad SAW yang atas perjuangnya kita bisa beranjak dari jaman jahiliyyah ke jaman ilmu pengetahuan.

Kami menyadari bahwasannya dalam pembuatan makalah ini banyak sekali kekurangan, hal ini di sebabkan atas kekuranganya pengetahuan serta sumber informasi yang saya miliki, oleh karena itu kami mengharapkan pembaca bisa memberikan kritik dan saran yang bersifat membangun kepada kami, sehingga  bisa memperbaiki makalah ini menjadi lebih baik lagi.

Akhir kata kami ucapakan semoga makalah ini bisa bermanfaat khususnya bagi kami umumnya bagi semua pembaca.

Wassalamualaikum wr.wb.

Bandung,11 Juli 2019

Penyusun

Program yang kami buat membutuhkan library processing untuk pemrosesan output dalam bentuk grafik dalam wujud kanvas.

**PROGRAM FUNGSI BOOLEAN TINGKAT 3**

import processing.core.PApplet;

import processing.core.PFont;

import processing.core.PGraphics;

import java.util.ArrayList;

public class MainClass extends PApplet {

private PGraphics canvas;

private final int functionLevel = 3;

private float textSize = 10;

private final float cellW = 80;

private final float cellH = 30;

private boolean finish;

private String currentProcess = "";

//inisialisasi library PApplet untuk keperluan grafik rendering

public static void main(String[] args) {

PApplet.main("MainClass", args);

}

//mengatur ukuran window kanvas

public void settings() {

size(floor(cellW \* (pow(2, functionLevel) + 1) + 2), 640);

}

//menjalankan blok program utama

public void setup() {

background(51);

textAlign(CENTER);

PFont font = loadFont("BloggerSans-Light-48.vlw");

textFont(font);

textSize(textSize);

thread("generateFunctions");

}

private boolean dragged;

private float startingDragY;

private float scrollY = 0;

private float tScrollY = 0;

private float scrollV = 0;

private float dragPointX;

private float dragPointY;

//untuk keperluan scrolling

//untuk tracking kecepatan scroll

public void mouseDragged() {

dragged = true;

scrollV = (startingDragY - mouseY);

}

//untuk memulai proses scrolling

public void mousePressed() {

startingDragY = mouseY;

tPointDragSize = 20;

dragPointX = mouseX;

dragPointY = mouseY;

}

//ketika mouse dilepas scrolling berhenti

public void mouseReleased() {

dragged = false;

scrollV = 0;

tPointDragSize = 0;

}

private float pointDragSize;

private float tPointDragSize;

//untuk mekanika scrolling kanvas

private void scrollCanvas() {

if (dragged) {

tScrollY = constrain(scrollY + scrollV, -(canvas.height - height), 150);

fill(255, 150);

stroke(255, 150);

line(dragPointX, dragPointY, mouseX, mouseY);

noStroke();

}

ellipse(dragPointX, dragPointY, pointDragSize, pointDragSize);

}

//untuk smoothing pada animasi

private void animationsStep(){

pointDragSize = lerp(pointDragSize, tPointDragSize, 0.2f);

scrollY = lerp(scrollY, tScrollY, 0.1f);

}

//untuk mengatur animasi dan akan berulang secara terus-menerus

public void draw() {

background(51);

animationsStep();

if (finish) {

image(canvas, 0, scrollY);

scrollCanvas();

fill(51);

stroke(0, 100);

rect(0, 0, width, 150);

textSize(40);

fill(255);

text("Semua kemungkinan fungsi boolean", width / 2, 50);

text("Tingkat " + functionLevel, width / 2, 100);

textSize(30);

text("Drag untuk scroll", width / 2, 135);

} else {

textSize(48);

fill(255);

text("Menggambar Kanvas", width / 2, height / 2);

textSize(30);

text(currentProcess, width / 2, height / 2 + 35);

}

}

//menampung satu fungsi boolean yang berisi beberapa cell

class BooleanFunc {

private String name;

float x;

float y;

private ArrayList<Cell> cells;

//untuk menerima empat parameter aktual yaitu posisi nilai boolean dan nama fungsi

BooleanFunc(float x, float y, boolean[] values, String name) {

this.name = name;

this.x = x;

this.y = y;

cells = new ArrayList<>();

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

Cell c = new Cell(x + i \* cellW, y, values[i]);

cells.add(c);

}

}

//untuk menampilkan nilai fungsinya

private void display(PGraphics p) {

p.stroke(100, 10, 10, 100);

p.fill(100, 10, 10, 100);

p.rect(x, y, cellW, cellH);

p.fill(255);

p.text(name, x + cellW / 2, y + cellH / 2 + textSize / 2);

p.pushMatrix();

p.translate(cellW, 0);

for (Cell c : cells) {

c.display(p);

}

p.popMatrix();

}

}

//cell merupakan satu kesatuan dari boolean function yang menampung tepat satu state true or false

class Cell {

boolean value;

float x;

float y;

int baseColor;

Cell(float x, float y, boolean value) {

this.x = x;

this.y = y;

this.value = value;

baseColor = color(0, 100, 255);

}

//jika dia true maka dia berwarna biru

private void display(PGraphics p) {

p.fill(value ? baseColor : 51, 200);

p.stroke(baseColor);

p.rect(x, y, cellW, cellH);

p.fill(255);

p.text(value ? 1 : 0, x + cellW / 2, y + cellH / 2 + textSize / 2);

}

}

//untuk mengkonversikan dari string binary ke array boolean

private boolean[] binaryToBoolArray(String binary) {

char[] binArray = binary.toCharArray();

boolean[] booleans = new boolean[binArray.length];

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

booleans[i] = binArray[i] == '1';

}

return booleans;

}

//untuk mengisi padding pada string binary yang tidak cukup

private String binaryFill(String binaryCode, int padding) {

while (binaryCode.length() < padding) {

binaryCode = '0' + binaryCode;

}

return binaryCode;

}

//saat program dijalankan fungsi dipanggil untuk membuat semua fungsi boolean dan menampilkannya ke kanvas

public void generateFunctions() {

ArrayList<BooleanFunc> booleanFuncs;

booleanFuncs = new ArrayList<>();

int permutCombination = (int) pow(2, functionLevel);

int permutFunction = (int) pow(2, permutCombination);

canvas = createGraphics((int) cellW \* (permutCombination + 1) + 2, (int) cellH \* permutFunction);

int processCount = 0;

for (int i = 0; i < permutFunction; i++) {

float percentage = floor((float) processCount / permutFunction \* 100 \* 100) / 100f;

currentProcess = "Membuat Fungsi " + percentage + "%";

String binary = binaryFill(Integer.toBinaryString(i), permutCombination);

boolean[] bs = binaryToBoolArray(binary);

BooleanFunc booleanFunc = new BooleanFunc(0, 0 + i \* cellH, bs, "F" + (i + 1));

booleanFuncs.add(booleanFunc);

processCount++;

}

canvas.textAlign(CENTER);

canvas.beginDraw();

canvas.background(51);

processCount = 0;

for (BooleanFunc b : booleanFuncs) {

float percentage = floor((float) processCount / permutFunction \* 100 \* 100) / 100f;

currentProcess = "Menggambar " + percentage + "%";

b.display(canvas);

processCount++;

}

canvas.endDraw();

finish = true;

canvas.save("rendered.png");

}

}



**PROGRAM PETA KARNAUGH**

import processing.core.PApplet;

import javax.swing.JOptionPane;

import java.util.\*;

public class MainClass extends PApplet {

private final int VARIABLES\_NUM = 4;

private final char[] variables = new char[]{'W', 'X', 'Y', 'Z'};

private int w = 120;

private int h = 120;

private float xOff = 0;

private float yOff = 0;

private String title = "Peta Karnaugh 4 Variabel";

private String title2 = "Arahkan kursor untuk melihat term";

private String terms = "";

private KarnaughMap karnaughMap = new KarnaughMap(VARIABLES\_NUM);

//inisialisasi library PApplet untuk keperluan grafik rendering

public static void main(String[] args) {

PApplet.main("MainClass", args);

}

//mengatur ukuran window kanvas

public void settings() {

size(600, 660);

}

//menjalankan blok program utama

public void setup() {

textFont(loadFont("BloggerSans-Light-48.vlw"));

textAlign(CENTER, CENTER);

showCredit();

rectMode(CENTER);

int[] inputs = takeInput();

terms = "{";

for (int i : inputs) {

terms += Integer.toString(i) + ',';

}

if (terms.charAt(terms.length() - 1) == ',') {

terms = terms.substring(0, terms.length() - 1);

}

terms += '}';

karnaughMap.mapInput(inputs);

}

//untuk mengatur animasi dan akan berulang secara terus-menerus

public void draw() {

background(51, noise(xOff, 500) \* 255);

textSize(20);

karnaughMap.show();

textSize(30);

text(title, width / 2, 25);

textSize(25);

text("Minterm: " + terms, width / 2, 55);

textSize(15);

text(title2, width / 2, 80);

yOff += 0.15;

xOff += 0.15;

}

private void showCredit() {

background(51);

textSize(48);

noStroke();

fill(0, 100);

rect(0, height / 2 - 50 \* 3, width, 50 \* 6);

pushMatrix();

translate(width / 2, height / 2 - 25 \* 3);

fill(255);

text("A Program by", 0, 0);

text("Mirza MY Humayung", 0, 50);

text("Ahmad Shofi Hasibuan", 0, 50 \* 2);

text("Ginar Cut Baktiyanesa", 0, 50 \* 3);

popMatrix();

delay(2000);

}

//+++++++++++++++++++++++++++++++++++++++++++++++++++++++funcs

//untuk mengkonversi dari array char menjadi array integer

private int[] toIntArray(String[] arr) {

int[] ints = new int[arr.length];

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

ints[i] = Integer.parseInt(arr[i]);

}

return ints;

}

//unutk menerjemahkan input dari user ke array integer

private int[] parseLineToIntArray(String line) {

return toIntArray(line.split(","));

}

//untuk mengkonversi binary ke graycode

private String binaryToGray(String binary) {

String gray = "";

gray += binary.charAt(0);

for (int i = 1; i < binary.length(); i++) {

gray += binary.charAt(i - 1) ^ binary.charAt(i);

}

return gray;

}

//untuk mengisi padding pada string binary yang tidak cukup

private String binaryFill(String binaryCode, int padding) {

while (binaryCode.length() < padding) {

binaryCode = '0' + binaryCode;

}

return binaryCode;

}

//untuk menerima input minterm dari user

private int[] takeInput() {

boolean validInput = false;

String message = "";

int[] inputs = null;

while (!validInput) {

try {

String line = JOptionPane.showInputDialog(message + "Masukkan nilai minterm untuk Peta Karnaugh 4 variabel\nContoh: 0,1,2,3,7,15", "0,1,2,3,7,15");

inputs = parseLineToIntArray(line);

validInput = true;

} catch (Exception e) {

message = "Masukan salah!\n";

validInput = false;

}

}

return inputs;

}

//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++CELL

//setiap cell pada peta karnaugh

class Cell {

float x, y;

boolean state = false;

int baseColor;

int hoverColor;

int neutralColor;

boolean mouseHover = false;

float cellW = w;

float cellH = h;

float rR = 0;

String term;

Cell(float x, float y, String code) {

this.x = x;

this.y = y;

this.baseColor = color(0, 100, 255); // button current color

this.hoverColor = color(0, 200, 255); // when mouse hover

term = "";

while (code.length() < 4) {

code = '0' + code;

}

term = "";

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

term += variables[i] + (code.charAt(i) == '0' ? "'" : "");

}

}

//untuk mengubah state cell menjadi true

void setTrue() {

this.state = true;

neutralColor = color(0, 100, 255);

}

//utnuk menampilkan setiap cell

Cell show() {

float noise = noise(xOff, x \* y / 10);

noStroke();

fill(lerpColor(baseColor, hoverColor, noise / 10));

rect(x, y, cellW + noise \* 2, cellH + noise \* 2, rR);

fill(255);

mouseHover = highlight();

text(mouseHover ? term : (state ? "1" : "0"), x, y);

return mouseHover ? this : null;

}

//ketika mouse mendekati cell maka cell akan berubah warna dan menunjukkan mintermnya

boolean highlight() {

boolean hover = mouseX > x - w / 2 && mouseX < x + w / 2 && mouseY > y - h / 2 && mouseY < y + h / 2;

if (hover) {

baseColor = lerpColor(baseColor, hoverColor, 0.1f);

cellW = lerp(cellW, w \* 0.1f + w, 0.08f);

cellH = lerp(cellH, h \* 0.1f + h, 0.08f);

rR = lerp(rR, 15, 0.08f);

} else {

baseColor = lerpColor(baseColor, neutralColor, 0.1f);

cellW = lerp(cellW, w, 0.08f);

cellH = lerp(cellH, h, 0.08f);

rR = lerp(rR, 0, 0.08f);

}

return hover;

}

}

//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++Karnaugh map

//satu kesatuan dari peta karnaugh yang berisi sejumlah cell

class KarnaughMap {

int varNum;

Map<String, Cell> terms = new HashMap<>();

KarnaughMap(int varNum) {

this.varNum = varNum;

init();

}

//proses inisialisasi peta karnaugh

void init() {

for (int i = 0; i < varNum; i++) {

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

String binaryCode = binaryFill(Integer.toBinaryString(i + j \* varNum), 4);

String grayCode = binaryToGray(binaryCode.substring(0, 2));

grayCode += binaryToGray(binaryCode.substring(2, 4));

float y = j \* h + h \* 1.5f;

float x = i \* w + w;

Cell cell = new Cell(x, y, grayCode);

terms.put(grayCode, cell);

}

}

}

//ketika user memasukkan inputan maka hasil inputan akan di mapping ke peta berdasarkan nilai mintermnya

void mapInput(int[] inputs) {

for (int c : inputs) {

String binaryCode = binaryFill(Integer.toBinaryString(c), 4);

terms.get(binaryCode).setTrue();

}

}

//menampilkan keseluruhan cell dari peta karnaugh

void show() {

Cell highLighted = null;

for (int i = 0; i < varNum \* varNum; i++) {

String binaryCode = binaryFill(Integer.toBinaryString(i), 4);

String grayCode = binaryToGray(binaryCode.substring(0, 2));

grayCode += binaryToGray(binaryCode.substring(2, 4));

Cell current = terms.get(grayCode).show();

if (current != null) highLighted = current;

}

if (highLighted != null) {

highLighted.show();

}

String column = variables[0] + "" + variables[1];

String row = variables[2] + "" + variables[3];

fill(100, 0, 0, 50);

float y = VARIABLES\_NUM / 2 \* h + h;

float x = VARIABLES\_NUM / 2 \* w + w / 2;

rect(3 \* w / 8, y, w / 4, h \* VARIABLES\_NUM);

rect(x, 7 \* h / 8, w \* VARIABLES\_NUM, h / 4);

fill(255);

text(row, width / 2, 7 \* h / 8);

text(column, 3 \* w / 8, y);

}

}

}

