

LAPORAN TUGAS KECIL 1

IF2211 STRATEGI ALGORITMA
PENYELESAIAN PERMAINAN KARTU 24 DENGAN
ALGORITMA BRUTE FORCE

Disusun untuk memenuhi tugas mata kuliah Strategi Algoritma
pada Semester 2 (dua) Tahun Akademik 2022/2023.



Oleh:

Muhamad Aji Wibisono

13521095

PROGRAM STUDI TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
2022

DAFTAR ISI

DAFTAR ISI.....	2
BAB I PENDAHULUAN.....	3
1.1. Latar Belakang	3
BAB II ALGORITMA.....	4
2.1. Penjelasan Algoritma Brute Force	4
2.2. Source Program dalam Bahasa Java.....	5
BAB III EKSPERIMEN	10
3.1. Hasil Program.....	10
LAMPIRAN.....	16

BAB I

PENDAHULUAN

1.1. Latar Belakang

Algoritma Brute Force adalah algoritma berupa metode lempang yang menyelesaikan permasalahan dengan mengandalkan kekuatan komputasi dan tidak menggunakan teknik – teknik lain untuk meningkatkan kemangkusan. Secara sederhana, algoritma Brute Force akan mencoba semua kemungkinan solusi untuk suatu masalah dan hanya berhenti jika menemukan solusi yang tepat. Sebagai contoh diberikan permasalahan berupa kunci untuk gembok dengan empat digit angka, algoritma Brute Force akan menyelesaikan masalah tersebut dengan mencoba angka dari 0001 hingga 9999 sampai akhirnya berhasil.

Algoritma Brute Force mempunyai kelebihan yaitu mudah untuk dipikirkan serta diimplementasikan dengan kekurangan yaitu waktu yang diperlukan dan keterandalan berlebih pada kekuatan komputasi mesin..

Permainan 24 adalah permainan kartu dengan pemain mengambil empat kartu secara acak lalu mencari cara dengan operasi tambah, kurang, kali, dan bagi sehingga mendapatkan nilai 24. Untuk laporan ini permainan nilai kartu untuk permainan 24 yang tidak berupa angka adalah sebagai berikut: kartu A (as) bernilai 1, J (jack) bernilai 11, Q (queen) bernilai 12, dan K (king) bernilai 13.

Pada laporan ini dipaparkan program yang telah dibuat untuk menyelesaikan permasalahan permainan 24 dengan algoritma Brute Force. Rincian dari program yang dipaparkan terdapat pada bab – bab selanjutnya.

BAB II

ALGORITMA

2.1. Penjelasan Algoritma Brute Force

Pada Algoritma Brute Force yang digunakan ada beberapa langkah yang dilakukan:

1. Menentukan permutasi dengan elemen sama pada kartu untuk urutan pengoperasian kartu. Hal ini dilakukan dengan mencari permutasi urutan kartu dengan indeks 0 1 2 3. Dari permutasi tersebut didapatkan urutan nilai kartu untuk urutan tersebut, lalu urutan tersebut dimasukkan kepada sebuah array yang mengingat nilai yang telah diambil jika nilai – nilai kartu belum terdapat pada array tersebut.
2. Menentukan cara pengoperasian yang dapat dilakukan. Hal ini dilakukan dengan looping operator tambah kurang kali dan bagi serta tanda kurung pada operasi tersebut. Untuk tambah kurang kali bagi dapat dilakukan dengan looping.
3. Mencoba semua kemungkinan pada permutasi kartu dan cara pengoperasian yang dapat dilakukan dengan looping untuk tiap permutasi kartu dan masing – masing operator, yaitu operator untuk kartu 0 dan 1, kartu 1 dan 2, dan kartu 2 dan 3.
4. Jika ada kombinasi kartu dan cara pengoperasian yang menghasilkan nilai 24 program akan membuat string yang sesuai dari operasi tersebut dan digabungkan pada string yang diperlukan untuk mengembalikan hasil akhir.
5. Untuk skema kurung penulis memilih empat skema kurung yang dapat merepresentasikan semua urutan operasi yang memungkinkan. Penulis memilih untuk melakukan hard-code pada setiap kemungkinan kurung yang dirasa perlu.
6. String hasil akan displit untuk tiap kombinasi yang memungkinkan dan dapat disajikan pada pengguna.

2.2. Source Program dalam Bahasa Java

a. Card.java

```
import java.util.Random;

public class Card {
    float[] Val = new float[4];

    public void readCards(float[] input){
        Val = input;
    }

    public void randomCards(){
        Random randomNum = new Random();
        int temp;
        for(byte i = 0; i < 4; i++){
            temp = randomNum.nextInt(13) + 1;
            Val[i] = ((float)temp);
            switch(temp){
                case 1:
                    System.out.print("A ");
                    break;
                case 11:
                    System.out.print("J ");
                    break;
                case 12:
                    System.out.print("Q ");
                    break;
                case 13:
                    System.out.print("K ");
                    break;
                default:
                    System.out.print(String.format("%d ", temp));
                    break;
            }
        }
        System.out.print("\n");
    }
}
```

b. Permutation.java

```
import java.util.Vector;

public class Permutation {
    Vector<float[]> cbmVector = new Vector<float[]>(1);

    boolean sameContent(float[] arr1, float[] arr2){
        boolean retval = true;
        byte i = 0;
        if(arr1.length != arr2.length){
            retval = false;
        }
        while(retval && i < arr1.length){
            if(arr1[i] != arr2[i]){
                retval = false;
            } else{
                i++;
            }
        }
        return retval;
    }

    boolean VectorContains(float[] arr){
        boolean retval = false;
        byte i;
        i = 0;
        while(!retval && i < cbmVector.size()){
            if(sameContent(arr, cbmVector.get(i))){
                retval = true;
            }
            i++;
        }
        return retval;
    }

    void GeneratePermutation(float[] Val){
```

```

        for(byte i = 0; i < 4; i++){
            for(byte j = 0; j < 4; j++){
                if(j!=i){
                    for(byte k = 0; k < 4; k++){
                        if(k!=i && k!=j){
                            for (byte l = 0; l < 4; l++){
                                if(l!=i && l!=j && l!=k){
                                    float[] getter = {Val[i], Val[j], Val[k], Val[l]};
                                    if(!VectorContains(getter)){
                                        cbmVector.add(getter);
                                    }
                                }
                            }
                        }
                    }
                }
            }
        }
    }
}

```

c. Op.java

```

import java.util.Vector;

public class Op {
    char[] Oper = {'+', '-', '*', '/'};
    Vector<String> retval = new Vector<String>(1);

    static float eval2 (float a, char op, float b){
        if(op == '+'){
            return a + b;
        }
        else if(op == '-'){
            return a - b;
        }
        else if(op == '*'){
            return a * b;
        }
        else return a / b;
    }

    public void getmodels(Vector<float[]> Cbm){
        byte i, j, k, l;
        for(i = 0; i < Cbm.size(); i++){
            for(j = 0; j < 4; j++){
                for(k = 0; k < 4; k++){
                    for(l = 0; l < 4; l++){
                        if (eval2(eval2(Cbm.get(i)[0], Oper[j], Cbm.get(i)[1]), Oper[k], eval2(Cbm.get(i)[2], Oper[l], Cbm.get(i)[3])) == 24){
                            retval.add(String.format("(%.0f %c %.0f) %c (%.0f %c %.0f)", Cbm.get(i)[0], Oper[j], Cbm.get(i)[1], Oper[k], Cbm.get(i)[2], Oper[l], Cbm.get(i)[3]));
                        }
                        if (eval2(eval2(eval2(Cbm.get(i)[0], Oper[j], Cbm.get(i)[1]), Oper[k], Cbm.get(i)[2]), Oper[l], Cbm.get(i)[3])) == 24){
                            retval.add(String.format("(%.0f %c %.0f) %c (%.0f) %c %.0f", Cbm.get(i)[0], Oper[j], Cbm.get(i)[1], Oper[k], Cbm.get(i)[2], Oper[l], Cbm.get(i)[3]));
                        }
                        if (eval2(eval2(Cbm.get(i)[0], Oper[j], eval2(Cbm.get(i)[1], Oper[k], Cbm.get(i)[2])), Oper[l], Cbm.get(i)[3])) == 24){
                            retval.add(String.format("(%.0f %c (%.0f %c %.0f)) %c %.0f", Cbm.get(i)[0], Oper[j], Cbm.get(i)[1], Oper[k], Cbm.get(i)[2], Oper[l], Cbm.get(i)[3]));
                        }
                        if (eval2(Cbm.get(i)[0], Oper[j], eval2(eval2(Cbm.get(i)[1], Oper[k], Cbm.get(i)[2]), Oper[l], Cbm.get(i)[3])) == 24){
                            retval.add(String.format("%.0f %c ((%.0f %c %.0f) %c %.0f)", Cbm.get(i)[0], Oper[j], Cbm.get(i)[1], Oper[k], Cbm.get(i)[2], Oper[l], Cbm.get(i)[3]));
                        }
                    }
                }
            }
        }
    }
}

```

d. InputHandler.java

```

import java.util.Scanner;

```

```

public class InputHandler{
    static Scanner in = new Scanner(System.in);
    String inputLine;
    String[] parsedInput;

    int Choice(int bottomLimit, int upperLimit){
        int retval = bottomLimit - 1;
        do{
            inputLine = in.nextLine();
            parsedInput = inputLine.split(" ");
            if(parsedInput.length == 1){
                try {
                    retval = Integer.parseInt(parsedInput[0]);
                } catch (NumberFormatException e) {
                    retval = bottomLimit-1;
                }
                if (retval < bottomLimit || retval > upperLimit) {
                    System.out.println("Input tidak valid");
                }
            }
            else{
                retval = bottomLimit - 1;
                System.out.println("Jumlah input tidak valid");
            }
        } while (retval < bottomLimit || retval > upperLimit);
        return retval;
    }

    String[] SpacedWords(int wordnum){
        boolean valid = false;
        do{
            inputLine = in.nextLine();
            parsedInput = inputLine.split(" ");
            if(parsedInput.length == wordnum){
                valid = true;
            }
            else{
                System.out.println("Jumlah input tidak valid");
            }
        } while (!valid);
        return parsedInput;
    }

    String StringLine(){
        return in.nextLine();
    }

    float[] CardCombination(){
        boolean valid = false;
        String[] vals;
        float[] Val = new float[4];
        while (!valid){
            vals = SpacedWords(4);
            try{
                for(byte i = 0; i < 4; i++){
                    switch(vals[i]){
                        case "A":
                            Val[i] = 1;
                            break;
                        case "J":
                            Val[i] = 11;
                            break;
                        case "Q":
                            Val[i] = 12;
                            break;
                        case "K":
                            Val[i] = 13;
                            break;
                        default:
                            Val[i] = Integer.parseInt(vals[i]);
                            if (Val[i] < 2 || Val[i] > 10){
                                throw new Exception("Ada kartu < 2 atau > 10 (Gunakan A, J, Q, K)");
                            }
                            break;
                    }
                }
                valid = true;
            } catch (Exception e){
                System.out.println("Input tidak valid!");
                System.out.println(e+"\n");
            }
        }
        return Val;
    }
}

```

e. OutputHandler.java

```
import java.io.BufferedWriter;
import java.io.FileWriter;
import java.util.Vector;

public class OutputHandler {
    BufferedWriter bw;
    public void printResult(Vector<String> retval){
        if(retval.size() == 0){
            System.out.println("Tidak ada solusi");
        }
        else{
            String output;
            System.out.println(String.format("Terdapat %d solusi:", retval.size()));
            for(int i = 0; i < retval.size(); i++){
                output = retval.get(i).replace("11", "J").replace("12", "Q").replace("13", "K").replace("10",
"T").replace("1", "A").replace("T", "10");
                System.out.println(output);
            }
        }
    }

    public void fileResult(String filename, Vector<String> retval){
        String fileOutput = filename + ".txt";

        if(retval.size() == 0){
            try{
                bw = new BufferedWriter(new FileWriter("./test/" + fileOutput));
                bw.write("Tidak ada solusi");
                bw.newLine();
                bw.flush();
                bw.close();
                System.out.println("\n\nFile berhasil disimpan pada /test/" + fileOutput);
            } catch(Exception e){
                System.out.println(e);
            }
        }
        else{
            String output;
            try{
                BufferedWriter bw = new BufferedWriter(new FileWriter("./test/" + fileOutput));
                bw.write(String.format("Terdapat %d solusi:", retval.size()));
                bw.newLine();
                for(int i = 0; i < retval.size(); i++){
                    output = retval.get(i).replace("11", "J").replace("12", "Q").replace("13", "K").replace("10",
"T").replace("1", "A").replace("T", "10");
                    bw.write(output);
                    bw.newLine();
                }
                bw.flush();
                bw.close();
                System.out.println("\n\nFile berhasil disimpan pada /test/" + fileOutput);
            } catch(Exception e){
                System.out.println(e);
            }
        }
    }
}
```

f. Interface.java

```
public class Interface{
    public void run(){

        boolean running = true;
        String filename;
        int input;
        long startTime;
        long finishTime;

        while (running){
            input = 0;
            startTime = 0;
```



```

        finishTime = 0;
        filename = "";
        Card cards = new Card();
        Op operation = new Op();
        InputHandler inputHandler = new InputHandler();
        OutputHandler outputHandler = new OutputHandler();
        Permutation cardPermutation = new Permutation();

        System.out.println("24 Solver");
        System.out.println("Pilih cara:");
        System.out.println("1. Input manual");
        System.out.println("2. Randomly generated");
        System.out.println("3. Exit");
        input = inputHandler.Choice(1, 3);

        switch(input){
            case 1:
                System.out.println("Masukkan 4 kombinasi kartu: ");
                cards.readCards(inputHandler.CardCombination());
                break;

            case 2:
                System.out.println("Kartu yang didapat: ");
                cards.randomCards();
                break;

            case 3:
                running = false;
                break;
        }

        if (running == true){
            startTime = System.nanoTime();
            cardPermutation.GeneratePermutation(cards.Val);
            operation.getmodels(cardPermutation.cbmVector);
            finishTime = System.nanoTime() - startTime;

            System.out.println("Selesai");
            System.out.println("Waktu eksekusi (milisekon): " + (double)finishTime / 1000000);
            System.out.println("Pilih output:");
            System.out.println("1. Terminal");
            System.out.println("2. File");
            input = inputHandler.Choice(1, 2);

            switch(input){
                case 1:
                    outputHandler.printResult(operation.retval);
                    break;

                case 2:
                    System.out.println("Masukkan nama file: ");
                    filename = inputHandler.StringLine();
                    outputHandler.fileResult(filename, operation.retval);
                    break;
            }

            System.out.println("\n\n");
        }

        System.out.println("Program Selesai");
    }
}

```

g. Main.java

```

public class Main {
    public static void main(String[] args) {
        Interface solver24 = new Interface();
        solver24.run();
    }
}

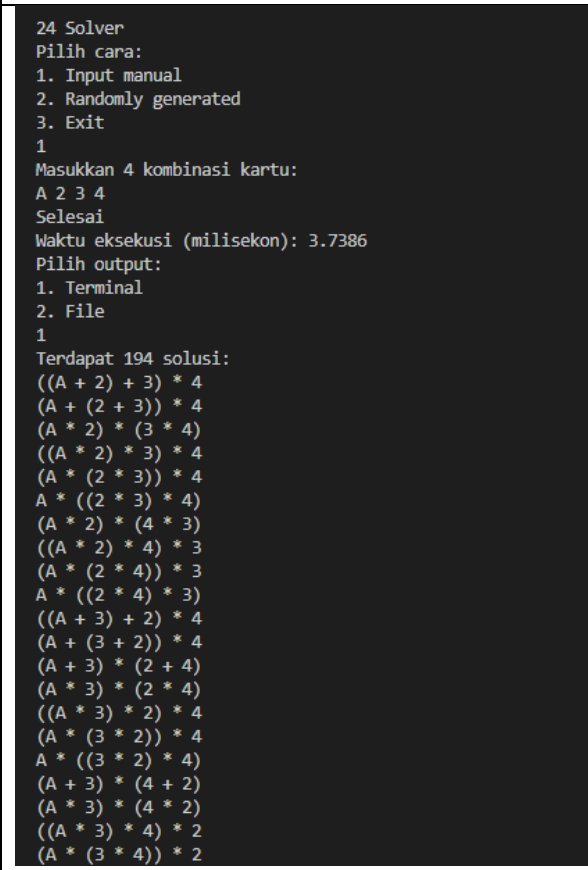
```

BAB III

EKSPERIMEN

3.1. Hasil Program

3.1.1. test1.txt

Gambar	Keterangan
	Keluaran hasil jika memberikan 4 kartu berbeda (1 2 3 4)

3.1.2. test2.txt

Gambar	Keterangan
--------	------------

<pre> 24 Solver Pilih cara: 1. Input manual 2. Randomly generated 3. Exit 1 Masukkan 4 kombinasi kartu: 6 6 6 6 Selesai Waktu eksekusi (milisekon): 0.8195 Pilih output: 1. Terminal 2. File 1 Terdapat 6 solusi: (6 + 6) + (6 + 6) ((6 + 6) + 6) + 6 (6 + (6 + 6)) + 6 6 + ((6 + 6) + 6) (6 * 6) - (6 + 6) ((6 * 6) - 6) - 6 </pre>	<p>Keluaran hasil jika memberikan 4 kartu yang sama (6 6 6 6)</p>
--	---

3.1.3. test3.txt

Gambar	Keterangan
<pre> 24 Solver Pilih cara: 1. Input manual 2. Randomly generated 3. Exit 1 Masukkan 4 kombinasi kartu: 2 2 2 2 Selesai Waktu eksekusi (milisekon): 0.0793 Pilih output: 1. Terminal 2. File 1 Tidak ada solusi </pre>	<p>Keluaran hasil jika memberikan 4 angka yang tidak memiliki solusi (2 2 2 2)</p>

3.1.4. test4.txt

Gambar	Keterangan
--------	------------

<pre> 24 Solver Pilih cara: 1. Input manual 2. Randomly generated 3. Exit 1 Masukkan 4 kombinasi kartu: A J Q K Selesai Waktu eksekusi (milisekon): 1.8319 Pilih output: 1. Terminal 2. File 1 Terdapat 26 solusi: (A * Q) * (K - J) ((A * K) - J) * Q (A * (K - J)) * Q A * ((K - J) * Q) (Q * A) * (K - J) Q * ((A * K) - J) (Q / A) * (K - J) Q * ((K * A) - J) Q * ((K / A) - J) (Q * (K - J)) * A Q * ((K - J) * A) (Q * (K - J)) / A Q * ((K - J) / A) (K - (A * J)) * Q ((K * A) - J) * Q ((K / A) - J) * Q (K - J) * (A * Q) ((K - J) * A) * Q (K - (J * A)) * Q ((K - J) / A) * Q (K - (J / A)) * Q (K - J) / (A / Q) (K - J) * (Q * A) ((K - J) * Q) * A (K - J) * (Q / A) ((K - J) * Q) / A </pre>	<p>Keluaran hasil jika memberikan 4 kartu yang bernilai huruf</p>
---	---

3.1.5. test5.txt

Gambar	Keterangan
--------	------------

<pre> 24 Solver Pilih cara: 1. Input manual 2. Randomly generated 3. Exit 2 Kartu yang didapat: 4 7 J Q Selesai Waktu eksekusi (milisekon): 1.2689 Pilih output: 1. Terminal 2. File 1 Terdapat 12 solusi: 4 * ((7 + J) - Q) 4 * ((7 - Q) + J) 4 * ((J + 7) - Q) 4 * ((J - Q) + 7) ((7 + J) - Q) * 4 (7 + (J - Q)) * 4 ((7 - Q) + J) * 4 (7 - (Q - J)) * 4 ((J + 7) - Q) * 4 (J + (7 - Q)) * 4 ((J - Q) + 7) * 4 (J - (Q - 7)) * 4 </pre>	<p>Keluaran hasil jika memilih kartu secara random</p>
---	--

3.1.6. test6.txt

Gambar	Keterangan
--------	------------

<pre> 24 Solver Pilih cara: 1. Input manual 2. Randomly generated 3. Exit 2 Kartu yang didapat: Q 8 2 4 Selesai Waktu eksekusi (milisekon): 1.7238 Pilih output: 1. Terminal 2. File 1 Terdapat 51 solusi: (Q + (8 * 2)) - 4 Q + ((8 * 2) - 4) ((Q - 8) + 2) * 4 (Q - (8 - 2)) * 4 (Q - 8) * (2 + 4) Q * ((8 - 2) - 4) (Q - 8) * (4 + 2) Q * ((8 - 4) - 2) (Q * (8 - 4)) / 2 Q * ((8 - 4) / 2) ((Q + 2) - 8) * 4 (Q + (2 - 8)) * 4 (Q + (2 * 8)) - 4 Q + ((2 * 8) - 4) (Q / 2) * (8 - 4) (Q - 4) + (8 * 2) ((Q + 4) * 2) - 8 (Q - 4) + (2 * 8) ((Q - 4) * 2) + 8 8 + ((Q - 4) * 2) (8 * 2) + (Q - 4) ((8 * 2) + Q) - 4 (8 - (2 + 4)) * Q ((8 - 2) - 4) * Q ((8 * 2) - 4) + Q </pre>	<p>Keluaran hasil lain jika memilih kartu secara random</p>
---	---

3.1.7. Contoh saat memilih output ke file

Gambar	Keterangan
--------	------------

```
24 Solver
Pilih cara:
1. Input manual
2. Randomly generated
3. Exit
1
Masukkan 4 kombinasi kartu:
6 6 6 6
Selesai
Waktu eksekusi (milisekon): 0.2022
Pilih output:
1. Terminal
2. File
2
Masukkan nama file:
test2

File berhasil disimpan pada /test/test2.txt
```

Contoh 3.1.6. dengan keluaran dipilih untuk pada file

LAMPIRAN

Cek List Spesifikasi

Poin	Ya	Tidak
1. Program berhasil dikompilasi tanpa kesalahan	✓	
2. Program berhasil <i>running</i>	✓	
3. Program dapat membaca input/generate sendiri dan memberikan luaran	✓	
4. Solusi yang diberikan	✓	
5. Program dapat menyimpan solusi dalam file text	✓	

Repository Github

Berikut adalah link repository GitHub untuk program penulis.

<https://github.com/MuhamadAjiW/Tucil1Stima>