

Tugas Kecil I IF2211 Strategi Algoritma
Semester II Tahun 2022/2023

Penyelesaian *24 Card Game* dengan Algoritma *Brute Force*



Disusun oleh :

Syarifa Dwi Purnamasari K03 / 13521018

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

2023

1. Algoritma *Brute Force*

Algoritma *brute force* merupakan salah satu metode untuk menemukan solusi yang dapat terbilang cukup praktis dan sederhana. Dengan menggunakan algoritma *brute force*, solusi dari permasalahan tersebut pasti ditemukan walaupun dengan waktu eksekusi yang cukup lama. Algoritma *brute force* sering digunakan karena metode pemecahan masalah ini dapat diterapkan hampir pada seluruh permasalahan.

Cara kerja dari algoritma *brute force* adalah dengan mencoba semua kemungkinan yang mungkin terjadi secara berurutan untuk mengidentifikasi dari seluruh kemungkinan tersebut, kemungkinan mana saja yang memenuhi untuk menjadi solusi permasalahan. Hal tersebut menjadikan algoritma *brute force* membutuhkan waktu eksekusi yang cukup lama. Maka dari itu, algoritma *brute force* normalnya digunakan saat *input* yang dimasukkan pengguna tidak terlalu besar dan pengguna tidak terlalu mengutamakan waktu eksekusi.

Permasalahan yang diangkat pada tugas kecil kali ini adalah penyelesaian permainan kartu 24 dengan pendekatan *brute force*. Permainan kartu 24 adalah permainan kartu dengan tujuan mencari cara untuk mengubah 4 kartu random sehingga mendapatkan hasil akhir sejumlah 24 dengan beberapa operasi aritmatika yaitu penjumlahan (+), pengurangan (-), perkalian (*), divisi (/), dan tanda kurung (). Prinsip dasar dari permainan ini adalah setiap kartu harus digunakan tepat sekali dan urutan penggunaannya bebas. Beberapa langkah yang dapat digunakan untuk mendapatkan solusi dari permainan ini menggunakan algoritma *brute force* antara lain.

1. Program meminta input berupa tepat 4 kartu dari user yang harus terdapat dalam 1 set kartu (A, 2, 3, 4, 5, 6, 7, 8, 9, 10, J, Q, K).
2. Program memunculkan semua kombinasi yang mungkin dari urutan 4 kartu tersebut dan operasi dasar aritmatika seperti pada spesifikasi.
3. Jika hasil dari ekspresi aritmatika tersebut menghasilkan angka 24, maka ekspresi berupa string tersebut akan dimasukkan dalam list solusi.

```
import java.util.Scanner;
import java.util.ArrayList;
import java.util.Random;

public class solver {
    // scanner
    public static Scanner input = new Scanner(System.in);

    // main
    public static void main(String[] args) {
        inputCard();

    }

    public static void splashScreen() {
        System.out.println("");
        System.out.println("\033[0;95m" + " _____
_____
_____");
        System.out.println("\033[0;91m" + " / ____ / 
/_ / / / _____ / / 
_/_ / / / ");
        System.out.println("\033[0;92m" + " /___/ / / 
____/ / / / / _____ / / 
_ / / / ");
        System.out.println("\033[0;96m" + " / / 
____ / / / _____ / / 
/ / / / / ");
```

```

        System.out.println("\033[0;94m" + "      /  /
/  /      /  / / ____ /  /  /  /  /  / ____/  /
/  /      /  /      /");
        System.out.println("\033[0;95m" + "      /  /
/  /      /  / / ____/  /  /  /  /  /  /
/  /      /  /      /");
        System.out.println("\033[0;96m" + "      /  /
/  /      /  / / ____/  /  /  /  /  /  /
/  /      /  /      /");
        System.out.println("\033[0;91m" + "      /  / ____
/  /  / ____/  /  /  /  /  /  /  /  /
/  /      /  /      /");
        System.out.println("\033[0;94m" + "      / ____
/ ____/      / ____/  / ____/  / ____/
/ ____/      / ____/");
        System.out.println("\033[0m");
    }

    // menu input
    public static void menu() {
        System.out.println("");
        System.out.println("      SELAMAT DATANG DI 24 GAME SOLVER
");
        System.out.println("");
        System.out.println("Pilih jenis input");
        System.out.println("1. Keyboard");
        System.out.println("2. Random");
        System.out.print("Masukan pilihan: ");
    }

    // menu output
    public static void outputOption() {
        System.out.println("");
        System.out.println("Apakah ingin menyimpan solusi? (y/n)");
        System.out.print("Masukan pilihan: ");
    }
}

```

2. *Input, Validasi Card, Print, dan Remove Same Solution*

```

// masukkan input 4 card
public static void inputCard() {
    ArrayList<String> listCard = new ArrayList<String>();
    ArrayList<Integer> lc = new ArrayList<Integer>();
    ArrayList<String> listSolve = new ArrayList<String>();
}

```

```

        boolean flag;
        do {
            flag = false;
            splashScreen();
            menu();
            String pilihan = input.nextLine();
            switch (pilihan) {
                case "1":
                    listCard = inputKeyboard();
                    lc = strToInt(listCard);
                    // printCard(lc);
                    long startTime = System.currentTimeMillis();
                    listSolve = solver24(lc);
                    printSolution(listSolve);
                    long endTime = System.currentTimeMillis();
                    long executeTime = (endTime - startTime);
                    System.out.println("Waktu yang diperlukan untuk
menghitung solusi " + executeTime + " ms.");
                    break;
                case "2": // masukkan input random
                    listCard = inputRandom();
                    lc = strToInt(listCard);
                    // printCard(lc);
                    startTime = System.currentTimeMillis();
                    listSolve = solver24(lc);
                    printSolution(listSolve);
                    endTime = System.currentTimeMillis();
                    executeTime = (endTime - startTime);
                    System.out.println("Waktu yang diperlukan untuk
menghitung solusi " + executeTime + " ms.");
                    break;
                default:
                    System.out.println("Pilihan tidak tersedia.
Silakan ulangi input.");
                    flag = true;
                    break;
            }
        } while (flag);
        do {
            flag = false;
            outputOption();
            String pilihan = input.nextLine();
            switch (pilihan) {
                case "y":

```

```

        save saver = new save();
        saver.saveToFile();
        printCard(listCard);
        listSolve = solver24(lc);
        printSolution(listSolve);
        saver.closeFile();
        break;
    case "n":
        System.out.println("\nTerima kasih telah
menggunakan program ini.");
        break;
    default:
        System.out.println("Pilihan tidak tersedia.
Silakan ulangi input.");
        flag = true;
        break;
    }
} while (flag);
}

// masukkan input random
public static ArrayList<String> inputRandom() {
    ArrayList<String> listCard = new ArrayList<String>();
    String[] card = new
String[]{"A", "2", "3", "4", "5", "6", "7", "8", "9", "10", "J", "Q", "K"};
    Random rand = new Random();

    System.out.println("");
    System.out.println("Kartu akan diacak secara random!");
    String s1 = card[rand.nextInt(card.length)];
    listCard.add(s1);
    String s2 = card[rand.nextInt(card.length)];
    listCard.add(s2);
    String s3 = card[rand.nextInt(card.length)];
    listCard.add(s3);
    String s4 = card[rand.nextInt(card.length)];
    listCard.add(s4);

    printCard(listCard);
    return listCard;
}

// masukkan input dari keyboard
public static ArrayList<String> inputKeyboard() {

```

```

        ArrayList<String> listCard = new ArrayList<String>();

        System.out.println("");
        System.out.println("Masukkan 4 kartu: ");

        String s = input.nextLine();
        String[] elements = s.split(" ");

        while ((elements.length != 4) || (!validCard(elements[0]))
|| (!validCard(elements[1])) || (!validCard(elements[2])) ||
(!validCard(elements[3]))) {
            System.out.println("Input tidak valid! Masukkan lagi
input.");
            s = input.nextLine();
            elements = s.split(" ");
        }

        listCard.add(elements[0]);
        listCard.add(elements[1]);
        listCard.add(elements[2]);
        listCard.add(elements[3]);

        // String s1 = input.nextLine();
        // while (!validCard(s1)) {
        //     System.out.println("Input tidak valid! Masukkan lagi
input.");
        //     s1 = input.nextLine();
        // }
        // listCard.add(s1);
        // String s2 = input.nextLine();
        // while (!validCard(s2)) {
        //     System.out.println("Input tidak valid! Masukkan lagi
input.");
        //     s2 = input.nextLine();
        // }
        // listCard.add(s2);
        // String s3 = input.nextLine();
        // while (!validCard(s3)) {
        //     System.out.println("Input tidak valid! Masukkan lagi
input.");
        //     s3 = input.nextLine();
        // }
        // listCard.add(s3);
        // String s4 = input.nextLine();

```

```

        // while (!validCard(s4)) {
        //     System.out.println("Input tidak valid! Masukkan lagi
input.");
        //     s4 = input.nextLine();
        // }
        // listCard.add(s4);

        // printCard(listCard);
        return listCard;
    }

    // print card
    public static void printCard(ArrayList<String> listCard) {
        System.out.println("");
        for (int i = 0; i < listCard.size(); i++) {
            System.out.print(listCard.get(i));
            System.out.print(" ");
        }
        System.out.println("");
    }

    // print solution
    public static void printSolution(ArrayList<String> listCard) {
        for (int i = 0; i < listCard.size(); i++) {
            System.out.println(listCard.get(i));
            // System.out.print(" ");
        }
    }

    // menghapus solusi yang sama dalam list
    public static ArrayList<String> removeSameSol(ArrayList<String>
listCard) {
        for (int d = 0; d < listCard.size()-1; d++) {
            for (int e = d+1; e < listCard.size(); e++) {
                if (listCard.get(d).equals(listCard.get(e))) {
                    listCard.remove(e);
                }
            }
        }
        return listCard;
    }

    // memvalidasi kartu yang diperbolehkan
    public static boolean validCard(String s) {

```



```

        s = new String(s).intern();
        return ((s == "A") || (s == "J") || (s == "Q") || (s ==
"K") || (s == "2") || (s == "3") || (s == "4") || (s == "5") || (s
== "6") || (s == "7") || (s == "8") || (s == "9") || (s == "10")));
    }

    // mengubah string ke integer
    public static ArrayList<Integer> strToInt(ArrayList<String> l)
    {
        ArrayList<Integer> lc = new ArrayList<Integer>();
        for (int i = 0; i < l.size(); i++) {
            String s = l.get(i).intern();
            if (s == "A") {
                lc.add(1);
            }
            else if (s == "J") {
                lc.add(11);
            }
            else if (s == "Q") {
                lc.add(12);
            }
            else if (s == "K") {
                lc.add(13);
            }
            else {
                lc.add(Integer.parseInt(s));
            }
        }
        return lc;
    }
}

```

3. Operator

```

    public static Double operator(Double card1, Double card2, int
op) {
        Double result = 0.0;
        if (op == 0) {
            result = (card1 + card2);
        }
        else if (op == 1) {
            result = (card1 - card2);
        }
        else if (op == 2) {
            result = (card1 * card2);
        }
    }
}

```

```

    }
    else if (op == 3) {
        result = (card1 / card2);
    }
    return result;
}

public static String operatorStr(int card1, int card2, int op)
{
    String result = "";
    String cardStr1 = Integer.toString(card1);
    String cardStr2 = Integer.toString(card2);
    if (op == 0) {
        result = "(" + cardStr1 + " + " + cardStr2 + ")";
    }
    else if (op == 1) {
        result = "(" + cardStr1 + " - " + cardStr2 + ")";
    }
    else if (op == 2) {
        result = "(" + cardStr1 + " * " + cardStr2 + ")";
    }
    else if (op == 3) {
        result = "(" + cardStr1 + " / " + cardStr2 + ")";
    }
    return result;
}

public static String operatorStr2(String card1, int card2, int
op) {
    String result = "";
    String cardStr2 = Integer.toString(card2);
    if (op == 0) {
        result = "(" + card1 + " + " + cardStr2 + ")";
    }
    else if (op == 1) {
        result = "(" + card1 + " - " + cardStr2 + ")";
    }
    else if (op == 2) {
        result = "(" + card1 + " * " + cardStr2 + ")";
    }
    else if (op == 3) {
        result = "(" + card1 + " / " + cardStr2 + ")";
    }
    return result;
}

```

```

    }

    public static String operatorStr3(int card1, String card2, int
op) {
        String result = "";
        String cardStr1 = Integer.toString(card1);
        if (op == 0) {
            result = "(" + cardStr1 + " + " + card2 + ")";
        }
        else if (op == 1) {
            result = "(" + cardStr1 + " - " + card2 + ")";
        }
        else if (op == 2) {
            result = "(" + cardStr1 + " * " + card2 + ")";
        }
        else if (op == 3) {
            result = "(" + cardStr1 + " / " + card2 + ")";
        }
        return result;
    }

    public static String operatorStr4(String card1, String card2,
int op) {
        String result = "";
        if (op == 0) {
            result = "(" + card1 + " + " + card2 + ")";
        }
        else if (op == 1) {
            result = "(" + card1 + " - " + card2 + ")";
        }
        else if (op == 2) {
            result = "(" + card1 + " * " + card2 + ")";
        }
        else if (op == 3) {
            result = "(" + card1 + " / " + card2 + ")";
        }
        return result;
    }
}

```

4. Solver

```

    public static ArrayList<String> solver24(ArrayList<Integer> lc)
{
    ArrayList<String> listCard = new ArrayList<String>();
}

```

```

        Double result1 = 0.0;
        Double result2 = 0.0;
        Double result3 = 0.0;
        Double result4 = 0.0;
        String resultStr = "";

        Double[] lCard = new Double[lc.size()];
        for (int i = 0; i < lc.size(); i++) {
            lCard[i] = (double) lc.get(i);
        }

        for (int i = 0; i < lc.size(); i++) {
            for (int j = 0; j < lc.size(); j++) {
                for (int k = 0; k < lc.size(); k++) {
                    for (int l = 0; l < lc.size(); l++) {
                        for (int a = 0; a < 4; a++) {
                            for (int b = 0; b < 4; b++) {
                                for (int c = 0; c < 4; c++) {
                                    if ((i != j) && (i != k) && (i
!= l) && (j != k) && (j != l) && (k != l)) {

                                        // ((lCard[i] .. lCard[j])
.. lCard[k]) .. lCard[l]

                                        result1 =
operator(operator(operator(lCard[i], lCard[j], a), lCard[k], b),
lCard[l], c);

                                        if (result1 == 24.00) {
                                            resultStr =
operatorStr2(operatorStr2(operatorStr(lc.get(i), lc.get(j), a),
lc.get(k), b), lc.get(l), c);

listCard.add(resultStr);

                                        }

                                        // (lCard[i] .. (lCard[j]
.. lCard[k])) .. lCard[l]

                                        result2 =
operator(operator(lCard[i], operator(lCard[j], lCard[k], b), a),
lCard[l], c);

                                        if (result2 == 24.00) {
                                            resultStr =
operatorStr2(operatorStr3(lc.get(i), operatorStr(lc.get(j),
lc.get(k), b), a), lc.get(l), c);

```

[illegible]

```

        } else {
            System.out.println("");
            System.out.println("Solusi ditemukan sebanyak " +
listCard.size());
        }
        return listCard;
    }
}

```

B. save.java

```

import java.io.*;
import java.util.Calendar;
import java.text.SimpleDateFormat;

public class save {
    public File file;

    public static final String dateFormat = "[yyyy-MM-dd_HH.mm.ss]";
    public static String now() {
        Calendar cal = Calendar.getInstance();
        SimpleDateFormat sdf = new SimpleDateFormat(dateFormat);
        return sdf.format(cal.getTime());
    }

    public void saveToFile() {
        try {
            String tanggal = now();
            this.file = new
File("../test\\output\\solution"+tanggal+".txt");
            PrintStream printStream = new PrintStream(this.file);
            System.setOut(printStream);
        } catch (FileNotFoundException e) {
            System.out.println("File tidak ditemukan");
        }
    }

    public void closeFile() {
        PrintStream consoleStream = new PrintStream(
            new FileOutputStream(FileDescriptor.out));
        System.setOut(consoleStream);
        System.out.println("");
    }
}

```

```

        System.out.println("Berhasil menyimpan file sebagai
"+this.file.getName());
    }
}

```

3. Screenshot Input dan Output dari Program

```

                SELAMAT DATANG DI 24 GAME SOLVER

Pilih jenis input
1. Keyboard
2. Random
Masukan pilihan: 1

Masukkan 4 kartu:
12 13 11 7
Input tidak valid! Masukkan lagi input.
Q K J 7
Solusi tidak ditemukan
Waktu yang diperlukan untuk menghitung solusi 5 ms.

Apakah ingin menyimpan solusi? (y/n)
Masukan pilihan: n

Terima kasih telah menggunakan program ini.
PS C:\Users\ASUS\Tucil1_13521018\bin>

```

Gambar 3.1 Testcase 1

```

                SELAMAT DATANG DI 24 GAME SOLVER

Pilih jenis input
1. Keyboard
2. Random
Masukan pilihan: 1

Masukkan 4 kartu:
4 7 9 10

Solusi ditemukan sebanyak 24
(4 - ((7 - 9) * 10))
(4 * ((7 + 9) - 10))
(4 * ((7 - 10) + 9))
(4 + ((9 - 7) * 10))
(4 * ((9 + 7) - 10))
(4 * ((9 - 10) + 7))
((4 * 10) - (7 + 9))
(((4 * 10) - 7) - 9)
((4 * 10) - (9 + 7))
(((4 * 10) - 9) - 7)
(((7 + 9) - 10) * 4)
((7 + (9 - 10)) * 4)
(((7 - 10) + 9) * 4)
((7 - (10 - 9)) * 4)
(((9 + 7) - 10) * 4)
((9 + (7 - 10)) * 4)
(((9 - 7) * 10) + 4)
(((9 - 10) + 7) * 4)
((9 - (10 - 7)) * 4)
((10 * 4) - (7 + 9))
(((10 * 4) - 7) - 9)
((10 * 4) - (9 + 7))
(((10 * 4) - 9) - 7)
((10 * (9 - 7)) + 4)
Waktu yang diperlukan untuk menghitung solusi 52 ms.

Apakah ingin menyimpan solusi? (y/n)
Masukan pilihan: n

```

Gambar 3.2 Testcase 2

```

24 GAME SOLVER

SELAMAT DATANG DI 24 GAME SOLVER

Pilih jenis input
1. Keyboard
2. Random
Masukan pilihan: 1

Masukkan 4 kartu:
6 6 6 6
Input tidak valid! Masukkan lagi input.
6 6 6 6

Solusi ditemukan sebanyak 6
(((6 + 6) + 6) + 6)
((6 + (6 + 6)) + 6)
((6 + 6) + (6 + 6))
(6 + ((6 + 6) + 6))
((6 * 6) - (6 + 6))
(((6 * 6) - 6) - 6)
Waktu yang diperlukan untuk menghitung solusi 47 ms.

Apakah ingin menyimpan solusi? (y/n)
Masukan pilihan: n

Terima kasih telah menggunakan program ini.
PS C:\Users\ASUS\Tucil1_13521018\bin>

```

Gambar 3.3 Testcase 3

```

24 GAME SOLVER

SELAMAT DATANG DI 24 GAME SOLVER

Pilih jenis input
1. Keyboard
2. Random
Masukan pilihan: 2

Kartu akan diacak secara random!

2 9 8 10

Solusi ditemukan sebanyak 16
((2 * (9 + 8)) - 10)
(((2 - 9) + 10) * 8)
((2 - (9 - 10)) * 8)
((2 * (8 + 9)) - 10)
(((2 + 10) - 9) * 8)
((2 + (10 - 9)) * 8)
(((9 + 8) * 2) - 10)
(8 * ((2 - 9) + 10))
(8 * ((2 + 10) - 9))
(((8 + 9) * 2) - 10)
(8 * ((10 + 2) - 9))
(8 * ((10 - 9) + 2))
(((10 + 2) - 9) * 8)
((10 + (2 - 9)) * 8)
(((10 - 9) + 2) * 8)
((10 - (9 - 2)) * 8)
Waktu yang diperlukan untuk menghitung solusi 45 ms.

Apakah ingin menyimpan solusi? (y/n)
Masukan pilihan: y

Berhasil menyimpan file sebagai solution[2023-01-24_23.28.39].txt

```

Gambar 3.4 Testcase 4

Solusi ditemukan sebanyak 16

$$\begin{array}{l} ((2 * (9 + 8)) - 10) \\ (((2 - 9) + 10) * 8) \\ ((2 - (9 - 10)) * 8) \\ ((2 * (8 + 9)) - 10) \\ (((2 + 10) - 9) * 8) \\ ((2 + (10 - 9)) * 8) \\ (((9 + 8) * 2) - 10) \\ (8 * ((2 - 9) + 10)) \\ (8 * ((2 + 10) - 9)) \\ (((8 + 9) * 2) - 10) \\ (8 * ((10 + 2) - 9)) \\ (8 * ((10 - 9) + 2)) \\ (((10 + 2) - 9) * 8) \\ ((10 + (2 - 9)) * 8) \\ (((10 - 9) + 2) * 8) \\ ((10 - (9 - 2)) * 8) \end{array}$$

Gambar 3.5 File txt dari Testcase 4

```

    _____
   /             \
  /               \
 /                 \
/                   \
\                   /
 \                 /
  \               /
   \             /
    \           /
     \         /
      \       /
       \     /
        \___/

SELAMAT DATANG DI 24 GAME SOLVER

Pilih jenis input
1. Keyboard
2. Random
Masukan pilihan: 2

Kartu akan diacak secara random!

9 6 J Q

Solusi ditemukan sebanyak 4
((6 * (11 - 9)) + 12)
(((11 - 9) * 6) + 12)
(12 - ((9 - 11) * 6))
(12 + ((11 - 9) * 6))
Waktu yang diperlukan untuk menghitung solusi 42 ms.

Apakah ingin menyimpan solusi? (y/n)
Masukan pilihan: y

Berhasil menyimpan file sebagai solution[2023-01-24_23.35.15].txt
PS C:\Users\ASUS\Tucil1_13521018\bin>

```

Gambar 3.6 Testcase 5

```

9 6 J Q

Solusi ditemukan sebanyak 4
((6 * (11 - 9)) + 12)
(((11 - 9) * 6) + 12)
(12 - ((9 - 11) * 6))
(12 + ((11 - 9) * 6))

```

Gambar 3.7 File txt dari Testcase 5

```

24 GAMES

SELAMAT DATANG DI 24 GAME SOLVER

Pilih jenis input
1. Keyboard
2. Random
Masukan pilihan: 2

Kartu akan diacak secara random!

A K A 7
Solusi tidak ditemukan
Waktu yang diperlukan untuk menghitung solusi 6 ms.

Apakah ingin menyimpan solusi? (y/n)
Masukan pilihan: y

Berhasil menyimpan file sebagai solution[2023-01-24_23.39.08].txt
PS C:\Users\ASUS\Tucil1_13521018\bin>

```

Gambar 3.8 Testcase 6

```

A K A 7
Solusi tidak ditemukan

```

Gambar 3.9 File txt dari Testcase 6

4. Link Repository GitHub

https://github.com/syrifaa/Tucil1_13521018.git

5. Checklist

<i>Poin</i>	<i>Ya</i>	<i>Tidak</i>
1. Program berhasil dikompilasi tanpa kesalahan	<i>V</i>	
2. Program berhasil <i>running</i>	<i>V</i>	
3. Program dapat membaca input / generate sendiri dan memberikan luaran	<i>V</i>	
4. Solusi yang diberikan program memenuhi (berhasil mencapai 24)	<i>V</i>	
5. Program dapat menyimpan solusi dalam file teks	<i>V</i>	