LAPORAN TUGAS KECIL 1
IF2211 STRATEGI ALGORITMA

PENYELESAIAN PERMAINAN KARTU 24 DENGAN ALGORITMA *BRUTE FORCE*

Oleh :

Muhammad Rizky Sya'ban

13521119

PROGRAM STUDI TEKNIK INFORMATIKA

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

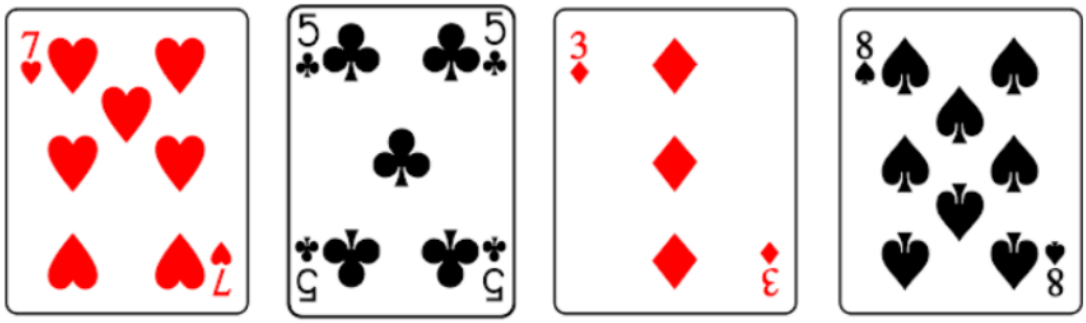
INSTITUT TEKNOLOGI BANDUNG

2022

Daftar Isi

BAB I
Deskripsi Program

Permainan kartu 24 adalah permainan kartu aritmatika dengan tujuan mencari cara untuk mengubah 4 buah angka random sehingga mendapatkan hasil akhir sejumlah 24. Permainan ini menarik cukup banyak peminat dikarenakan dapat meningkatkan kemampuan berhitung serta mengasah otak agar dapat berpikir dengan cepat dan akurat. Permainan Kartu 24 biasa dimainkan dengan menggunakan kartu remi. Kartu remi terdiri dari 52 kartu yang terbagi menjadi empat suit (sekop, hati, keriting, dan wajik) yang masing-masing terdiri dari 13 kartu (As, 2, 3, 4, 5, 6, 7, 8, 9, 10, Jack, Queen, dan King). Yang perlu diperhatikan hanyalah nilai kartu yang didapat (As, 2, 3, 4, 5, 6, 7, 8, 9, 10, Jack, Queen, dan King). As bernilai 1, Jack bernilai 11, Queen bernilai 12, King bernilai 13, sedangkan kartu bilangan memiliki nilai dari bilangan itu sendiri. Pada awal permainan moderator atau salah satu pemain mengambil 4 kartu dari dek yang sudah dikocok secara random. Permainan berakhir ketika pemain berhasil menemukan solusi untuk membuat kumpulan nilainya menjadi 24. **Pengubahan nilai tersebut dapat dilakukan** menggunakan operasi dasar matematika penjumlahan (+), pengurangan (-), perkalian (×), divisi (/) dan tanda kurung (()). Tiap kartu harus digunakan tepat sekali dan urutan penggunaannya bebas. (Paragraf di atas dikutip dari : https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2015-2016/Makalah-2016/MakalahStima-2016-038.pdf).



Gambar 1. Permainan Kartu 24

BAB II
Algoritma Program

Pada program ini digunakan algoritma brute-force untuk menemukan semua solusi yang mungkin untuk membentuk nilai 24. Adapun langkah-langkah yang digunakan pada program yaitu:

1. Program menerima masukan dari *user* lalu dievaluasi masukannya dengan batasan harus 4 buah kartu yang dipisahkan oleh spasi.
2. Kemudian dievaluasi semua kemungkinan yang mungkin dibentuk oleh kombinasi 4 angka dan 3 operasi menggunakan permutasi yang dalam program dilakukan menggunakan *for loop* sebanyak 7 kali yaitu 4 kali *looping* untuk mengiterasi *list* yang berisi masukan *user* dimana terjadi penghapusan elemen pada list untuk setiap kali *looping*-nya. Sedangkan 3 *looping* sisanya untuk permutasi 4 operasi yang digunakan. Sehingga terdapat $(4 \times 3 \times 2 \times 1) \times (4 \times 4 \times 4)$ pengevaluasian. Dan disetiap evaluasi terdapat 5 jenis pengelompokkan yaitu:
   - (((num1 op1 num2) op2 num3) op3 num4)
   - ((num1 op1 (num2 op2 num3)) op3 num4)
   - (num1 op1 (num2 op2 (num3 op3 num4)))
   - (num1 op1 ((num2 op2 num3) op3 num4))
   - (num1 op1 num2) op2 (num3 op3 num4)

   Kurung menandakan operasi tersebut dihitung lebih dulu. Sehingga total terdapat $(4 \times 3 \times 2 \times 1) \times (4 \times 4 \times 4) \times 5 = 7680$ kali evaluasi.
3. Hasil evaluasi yang menghasilkan solusi bernilai 24 akan disimpan pada *vector* listSol berupa string unik.
4. Terakhir, program akan menampilkan semua solusi unik yang tersimpan dalam listSol ke layar.

Dengan algoritma ini, semua solusi yang ditampilkan unik, namun sifat komutatif operasi diabaikan yang berarti $(a \times b) \neq (b \times a)$ begitupun dengan $(a + b) \neq (b + a)$.

Pranala *Repository* : https://github.com/mrsyaban/Tucil1_13521119

BAB III
Source Code

```cpp
#include <iostream>
#include <sstream>
#include <fstream>
#include <vector>
#include <chrono>
#include <string>

using namespace std;
using namespace std::chrono;

/* variabel global*/
vector<string> listSol; // list yang menampung string solusi
bool valid = true; // false berarti masukan user salah

/***
 * Mengembalikan angka berdasarkan masukan input
 */
int convertInput(string input){
    int num;
    if (input == "A"){
        return 1;
    } else if (input == "J"){
        return 11;
    } else if (input == "Q") {
        return 12;
    } else if (input == "K") {
        return 13;
    } else {
        stringstream ss;
        ss << input;
        ss >> num;
        if (num < 2 || num >10){
            valid = false;
        }
        return num;
    }
}

/***
 * Mengembalikan hasil operasi dari : num1 op num2
 */
double operate(double num1, double num2, char op){
    if (op == '+'){
        return num1 + num2;
    } else if (op == '-'){
        return num1 - num2;
    } else if (op == 'x'){
        return num1 * num2;
    } else {
        return num1 / num2;
    }
}
```

```
/***
 * Mengembalikan string yang merupakan hasil konkatenasi : (num1 op num2)
 */
string concat(string num1, string num2, char op){
    string res;
    string opStr;
    res = "("+ num1 +" " + opStr.append(1, op) + " " + num2 + ")";
    return res;
}

/***
 * Mengembalikan string hasil konversi num
 */
string convert(double num){
    return to_string(int(num));
}

/**
 * Menambahkan solution pada listSol jika solution belum ada di listSol
 */
void addSolution(string solution){
    bool ada = false;
    for (auto sol: listSol){
        if (solution == sol){
            ada= true;
        }
    }
    if (!ada){
        listSol.push_back(solution);
    }
}

/***
 * menambahkan semua solusi yang memenuhi dari 1 kombinasi masukan dan urutan operasi
 */
void findSol(double num1, double num2, double num3, double num4, char op1, char op2, char op3){
    string solution;
    // ((num1 op1 num2) op2 num3) op3 num4
    if (operate(operate(operate(num1, num2, op1), num3, op2), num4, op3) == 24) {
        solution = concat(concat(concat(convert(num1), convert(num2), op1), convert(num3), op2), convert(num4), op3);
        addSolution(solution);
    }
    // (num1 op1 (num2 op2 num3)) op3 num4
    if (operate(operate(num1, operate(num2, num3, op2), op1), num4, op3) ==  24){
        solution = concat(concat(convert(num1), concat(convert(num2), convert(num3), op2), op1), convert(num4), op3);
        addSolution(solution);
    }
    // num1 op1 (num2 op2 (num3 op3 num4))
    if (operate(num1, operate(num2, operate(num3, num4, op3), op2), op1) == 24){
        solution = concat(convert(num1), concat(convert(num2), concat(convert(num3), convert(num4), op3), op2), op1);
        addSolution(solution);
    }
    // num1 op1 ((num2 op2 num3) op3 num4)
    if (operate(num1, operate(operate(num2, num3, op2), num4, op3), op1) == 24){
        solution = concat(convert(num1), concat(concat(convert(num2), convert(num3), op2), convert(num4), op3), op1);
        addSolution(solution);
    }
    // (num1 op1 num2) op2 (num3 op3 num4)
    if (operate(operate(num1, num2, op1), operate(num3, num4, op3), op2) ==  24){
        solution = concat(concat(convert(num1), convert(num2), op1), concat(convert(num3), convert(num4), op3), op2);
        addSolution(solution);
    }
}
```

```
/***
 * Menghapus num pada listNum, num pasti ada di listNum
 */
vector<int> removeNum(vector<int> listNum, int num){
    int count = 0;
    while(listNum[count] != num){
        count++;
    }
    listNum.erase(listNum.begin()+count);
    return listNum;
}

/***
 * Mengembalikan jumlah semua kemungkinan yang memenuhi dari 4 angka masukan dengan
 * melakukan permutasi pada 4 angka masukan dan 4 operasi yang menghasilkan (4!)*(4^3)*5
 * urutan angka-operasi yang berbeda
 */
void permut(vector<int> listNum, vector<char> listOp){
    for (auto num1: listNum){
        vector<int> listNum2 = removeNum(listNum, num1);
        for (auto num2: listNum2){
            vector<int> listNum3 = removeNum(listNum2, num2);
            for (auto num3: listNum3){
                vector<int> listNum4 = removeNum(listNum3, num3);
                for (auto num4: listNum4){
                    for(auto op1: listOp){
                        for(auto op2: listOp){
                            for(auto op3: listOp){
                                findSol(num1, num2, num3, num4, op1, op2, op3);
                            }
                        }
                    }
                }
            }
        }
    }
}

/***
 * Mengembalikan list of string yang berisi hasil split str by space
 */
vector<string> splitString(string str){
    string res = "";
    vector<string> splitRes;
    str = str + " ";
    for(auto word: str){
        if (word == ' '){
            splitRes.push_back(res);
            res ="";
        } else{
            res += word;
        }
    }

    if (splitRes.size() != 4){
        valid = false;
    }
    return splitRes;
}
```

```
/* *** MAIN PROGRAM *** */
int main(){
    string input;
    cout << R"(
                                        __
                                    _.-''--'----_.
                                ,''.-''| .---/ _`-._
                                ,' \ \  ;| | ,/ / `-._`-.
                            ,' ,',\ \( | |// /,-._  / /
                            ;.`. `,\ \`| |/ / |   )/ /
                            / /`_ .\_\ \| /_.-.'-''/ /
                            / /_|_:.`. \ |;'`..')  / /
                            `-._`-._`.`.;`.\  ,'  / /
                                `-._`.`/    ,'-._/ /
                                : `-/     \`-.._/
                                |  :       ;._ (
                                :  |       \  `\.
                                \           \   |
                                    :        :   ;
                                    |            /
                                    ;          ,'
                                /          /
                                /          /
                                        /
 _ _ _       _                          _           ___  ___       _____       _
| | | | ___ | | ___  ___  _____  ___   | |_  ___   |_  || | | ___ |   __| ___ | | _ _  ___  ___
| | | || -_|| ||  _|| . ||     || -_|  |  _|| . |  |  _||_  ||___||__   || . || || | || -_||  _|
|_____||___||_||___||___||_|_|_||___|  |_|  |___|  |___|  |_|     |_____||___||_| \_/ |___||_|

            )" << endl;
    cout << endl << "Enter 4 Cards [separated by space] : ";
    getline (cin, input);
    auto start = high_resolution_clock::now();

    vector<string> listInput = splitString(input);
    int a = convertInput(listInput[0]), b = convertInput(listInput[1]), c = convertInput(listInput[2]), d = convertInput(
listInput[3]);

    if (!valid) {
        cout << "Incorrect input." << endl;
    } else {
        vector<int> listNum{a,b,c,d};
        vector<char> listOp{'+','-','x','/'};
        permut(listNum, listOp);
        auto stop = high_resolution_clock::now();
        auto duration = duration_cast<microseconds>(stop - start);

        int numOfSolution = listSol.size();
        if (numOfSolution > 0){
            cout << numOfSolution << " Solutions found" << endl << endl;
            for(auto sol: listSol){
                cout << sol << endl;
            }

            string ans;
            cout << "Do you want to save the solution? [yes/no] : ";
            cin >> ans;
            if (ans == "yes"){
                string filesName;
                cout << "Enter file's name (.txt) : ";
                cin >> filesName;
                ofstream out("..\\test\\" + filesName);
                out << numOfSolution << " Solutions found" << endl;
                for (auto solution: listSol){
                    out << solution << endl;
                }
                out.close();
            } else if (ans != "no") {
                cout << "Incorrect Input";
            }
        } else {
            cout << "There Is No Solution." << endl;
        }
        cout << endl << "Time taken : " << duration.count() << " microseconds" << endl;
    }
}
```

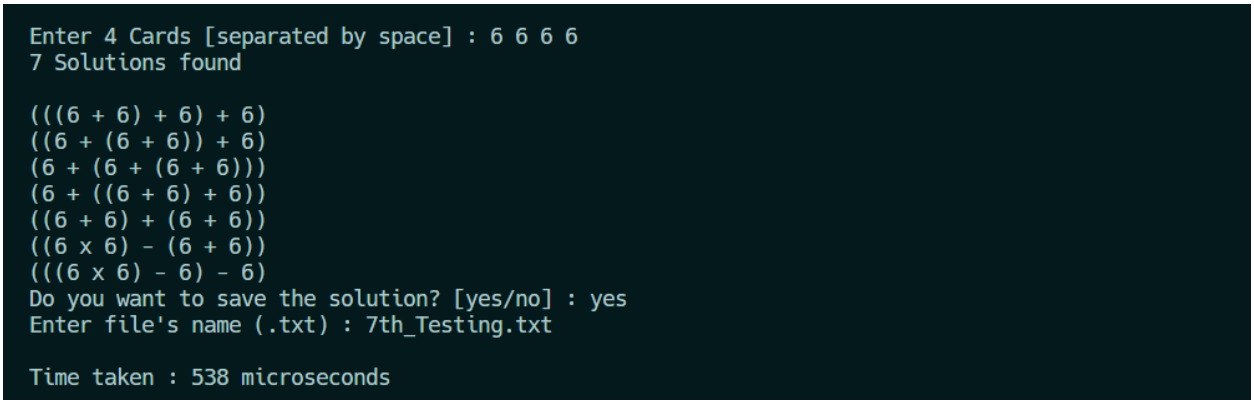
BAB IV
Eksperimen

Berikut adalah beberapa contoh keluaran program:

A. Masukan tidak valid

B. Masukan valid tapi tidak ada solusi yang memenuhi

C. Masukan benar dan terdapat solusi yang memenuhi



```
Enter 4 Cards [separated by space] : J Q K A
32 Solutions found

((12 x (13 - 11)) x 1)
(12 x (13 - (11 x 1)))
(12 x ((13 - 11) x 1))
((12 x (13 - 11)) / 1)
(12 x (13 - (11 / 1)))
(12 x ((13 - 11) / 1))
(12 x (13 - (1 x 11)))
(12 x ((13 x 1) - 11))
(12 x ((13 / 1) - 11))
(12 x (1 x (13 - 11)))
(12 x ((1 x 13) - 11))
((12 x 1) x (13 - 11))
((12 / 1) x (13 - 11))
(12 / (1 / (13 - 11)))
(((13 - 11) x 12) x 1)
((13 - 11) x (12 x 1))
(((13 - 11) x 12) / 1)
((13 - 11) x (12 / 1))
(((13 - 11) x 1) x 12)
((13 - (11 x 1)) x 12)
((13 - 11) x (1 x 12))
(((13 - 11) / 1) x 12)
((13 - (11 / 1)) x 12)
((13 - 11) / (1 / 12))
((13 - (1 x 11)) x 12)
(((13 x 1) - 11) x 12)
(((13 / 1) - 11) x 12)
(1 x (12 x (13 - 11)))
((1 x 12) x (13 - 11))
(((1 x 13) - 11) x 12)
((1 x (13 - 11)) x 12)
(1 x ((13 - 11) x 12))
Do you want to save the solution? [yes/no] : yes
Enter file's name (.txt) : 5th_Testing.txt

Time taken : 378 microseconds
```

```
Enter 4 Cards [separated by space] : A 7 7 Q
4 Solutions found

((1 + (7 / 7)) x 12)
(((7 / 7) + 1) x 12)
(12 x (1 + (7 / 7)))
(12 x ((7 / 7) + 1))
Do you want to save the solution? [yes/no] : yes
Enter file's name (.txt) : 6th_Testing

Time taken : 366 microseconds
```

```
Enter 4 Cards [separated by space] : 7 8 9 10
8 Solutions found

(8 x (9 / (10 - 7)))
((8 x 9) / (10 - 7))
((8 / (10 - 7)) x 9)
(8 / ((10 - 7) / 9))
(9 x (8 / (10 - 7)))
((9 x 8) / (10 - 7))
((9 / (10 - 7)) x 8)
(9 / ((10 - 7) / 8))
Do you want to save the solution? [yes/no] : yes
Enter file's name (.txt) : 8th_Testing.txt

Time taken : 366 microseconds
```

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