LAPORAN TUGAS KECIL

IF2211 STRATEGI ALGORITMA

PENYELESAIAN KARTU 24 DENGAN ALGORITMA BRUTE FORCE

Disusun oleh :

Muhammad Abdul Aziz Ghazali - 13521128

PROGRAM STUDI TEKNIK INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG BANDUNG

DAFTAR ISI

BAB I

DESKRIPSI MASALAH

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. Dibuatlah program sederhana dalam C++ yang mengimplementasikan *algoritma Brute Force* untuk mencari seluruh solusi permainan kartu 24

BAB II

ALGORITMA BRUTE FORCE

Dalam permainan kartu 24, terdapat 52 kartu yang berbeda, dan ada 4 jenis bentuk kartu, yaitu hati, wajik, sekop, dan keriting.

Dalam algoritma brute force yang dipakai, kita akan membentuk 4 urutan kartu dengan nilainya masing-masing, dan diantara nilai tersebut terdapat 3 operasi antarnilai yang berjenis 4 (tambah, kurang, kali, bagi).

Yaitu sebagai berikut gambarannya :

Kartu1 operasi1 kartu2 operasi2 kartu3 operasi3 kartu4

Karena kita akan menelusuri semua kemungkinan yang ada dengan brute force, kita akan mencoba semua 4 kartu pada setiap masing-masing posisi kartu, dimana setiap kartu hanya boleh mengisi 1 posisi dan keempat jenis operasi dapat mengisi pada posisi operasi tanpa harus berbeda jenis.

Ide kartu tersebut diimplementasikan dengan looping permutasi pada bahasa C++. Contohnya :

(2 operasi1 4 operasi2 6 operasi3 9) diubah dengan looping menjadi (2 operasi1 4 operasi2 9 operasi3 6).

Penelusuran semua kemungkinan operasi juga dilakukan dengan looping pada 4 jenis operasi. Contohnya :

(kartu1 + kartu2 x kartu3 + kartu4) kemudian pada looping berikutnya menjadi (kartu1 + kartu2 x kartu3 - kartu4)

Semua posisi kartu dan operasi sudah diketahui, langkah selanjutnya adalah menentukan urutan operasi yang dilakukan. Hanya ada 5 kemungkinan dari tiap posisi kartu dan operasi pada permainan kartu 24. Yaitu :

((Kartu1 operasi1 kartu2) operasi2 kartu3) operasi3 kartu4

((Kartu1 operasi1 (kartu2 operasi2 kartu3)) operasi3 kartu4

Kartu1 operasi1 ((kartu2 operasi2 kartu3) operasi3 kartu4)

Kartu1 operasi1 (kartu2 operasi2 (kartu3 operasi3 kartu4))

(Kartu1 operasi1 kartu2) operasi2 (kartu3 operasi3 kartu4)

Dengan tanda kurung menunjukan prioritas yang harus dihitung terlebih dahulu, dan diimplementasikan secara manual pada program.

Solusi sudah dapat ditemukan. Namun, terdapat jawaban yang komutatif dan akan dibiarkan karena tetap memenuhi peraturan permainan 24 yang beredar.

BAB III

IMPLEMENTASI DASAR SOURCE PROGRAM

```
#include <iostream>
#include <string>
#include <cstring>
#include <ctime>
#include <fstream>
#include <sstream>
#include <cstdlib>


using namespace std;

int jumlahSolusi = 0;
string solusi;
stringstream ss;

bool isCardValid(int * cards){
    for(int i = 0; i < 4; i++){

        if(cards[i] < 1 || cards[i] > 13){
            cout << "Masukan kartu tidak valid" << endl;
            return false;
        }
    }
    return true;
}


int convertToValue(char X){
    if (X == 'J'){
        return 11;
    }
    else if(X == 'Q'){
        return 12;
    }
    else if(X == 'K'){
        return 13;
```

```
    }
    else if(X == 'A'){
        return 1;
    }
    else if(X - '0' > 0 && X - '0' < 10){
        return X - '0';

    }
    else{
        return 999999999;
    }
}

char convertToChar(int X){
    if(X == 1){
        return 'A';
    }
    else if(X == 11){
        return 'J';
    }
    else if(X == 12){
        return 'Q';
    }
    else if(X == 13){
        return 'K';
    }
    else{
        return X + '0';
    }
}


float operateOp(float a, float b, int op){
    if( a == 999999999 || b == 999999999){
        return 999999999;
    }
    else{
        if(op == 0){
            return a + b;
        }
        else if(op == 1){
            return a - b;
        }
        else if(op == 2){
            return a * b;
```

```
        }
        else if(op == 3){
            if( b == 0)
                return 999999999;
            else
                return a / b;
        }
    }
}

bool is24(float result){
    if( result > 23.9 && result < 24.1){
        return true;
    }
    else{
        return false;
    }
}

char charOp(int op){
    if(op == 0){
        return '+';
    }
    else if(op == 1){
        return '-';
    }
    else if(op == 2){
        return '*';
    }
    else if(op == 3){
        return '/';
    }
}


void calculateCard(int a, int b, int c, int d, int op1, int op2, int op3, int *
listCards){
    float w,x, y, z;

    w = float(listCards[a]);
    x = float(listCards[b]);
    y = float(listCards[c]);
    z = float(listCards[d]);

    float result = 0;
```

```
    /* Kurung ke 1 */


    result = operateOp(operateOp(operateOp(w, x, op1), y, op2), z, op3);
    if(is24(result)){
        cout << "((" << convertToChar(listCards[a]) << " " << charOp(op1) << " "
<< convertToChar(listCards[b]) << ") " << charOp(op2) << " " <<
convertToChar(listCards[c]) << ") " << charOp(op3) << " " <<
convertToChar(listCards[d]) << endl;
        ss << "((" << convertToChar(listCards[a]) << " " << charOp(op1) << " " <<
convertToChar(listCards[b]) << ") " << charOp(op2) << " " <<
convertToChar(listCards[c]) << ") " << charOp(op3) << " " <<
convertToChar(listCards[d]) << '\n';

        jumlahSolusi++;
    }

    result = operateOp(operateOp(w, operateOp(x, y, op2), op1), z, op3);
    if(is24(result)){
        cout << "(" << convertToChar(listCards[a]) << " " << charOp(op1) << " ("
<< convertToChar(listCards[b]) << " " << charOp(op2) << " " <<
convertToChar(listCards[c]) << ")) " << charOp(op3) << " " <<
convertToChar(listCards[d]) << endl;
        ss << "(" << convertToChar(listCards[a]) << " " << charOp(op1) << " (" <<
convertToChar(listCards[b]) << " " << charOp(op2) << " " <<
convertToChar(listCards[c]) << ")) " << charOp(op3) << " " <<
convertToChar(listCards[d]) << '\n';
        jumlahSolusi++;
    }

    result = operateOp(w, operateOp(operateOp(x, y, op2), z, op3), op1);
    if(is24(result)){
        cout << convertToChar(listCards[a]) << " " << charOp(op1) << " ((" <<
convertToChar(listCards[b]) << " " << charOp(op2) << " " <<
convertToChar(listCards[c]) << ") " << charOp(op3) << " " <<
convertToChar(listCards[d]) << ")" << endl;
        ss << convertToChar(listCards[a]) << " " << charOp(op1) << " ((" <<
convertToChar(listCards[b]) << " " << charOp(op2) << " " <<
convertToChar(listCards[c]) << ") " << charOp(op3) << " " <<
convertToChar(listCards[d]) << ")" << '\n';
        jumlahSolusi++;
    }

    result = operateOp(w, operateOp(x, operateOp(y, z, op3), op2), op1);
```

```
    if(is24(result)){
        cout << convertToChar(listCards[a]) << " " << charOp(op1) << " (" <<
convertToChar(listCards[b]) << " " << charOp(op2) << " (" <<
convertToChar(listCards[c]) << " " << charOp(op3) << " " <<
convertToChar(listCards[d]) << "))" << endl;
        ss << convertToChar(listCards[a]) << " " << charOp(op1) << " (" <<
convertToChar(listCards[b]) << " " << charOp(op2) << " (" <<
convertToChar(listCards[c]) << " " << charOp(op3) << " " <<
convertToChar(listCards[d]) << "))" << '\n';
        jumlahSolusi++;
    }

    result = operateOp(operateOp(w, x, op1), operateOp(y, z, op3), op2);
    if(is24(result)){
        cout << "(" << convertToChar(listCards[a]) << " " << charOp(op1) << " "
<< convertToChar(listCards[b]) << ") " << charOp(op2) << " (" <<
convertToChar(listCards[c]) << " "<< charOp(op3) << " " <<
convertToChar(listCards[d]) << ")" << endl;
        ss << "(" << convertToChar(listCards[a]) << " " << charOp(op1) << " " <<
convertToChar(listCards[b]) << ") " << charOp(op2) << " (" <<
convertToChar(listCards[c]) << " "<< charOp(op3) << " " <<
convertToChar(listCards[d]) << ")" << '\n';
        jumlahSolusi++;
    }



}


int main(){
    char opsiInput;
    int cards [4];
    char cardsChar [4];
    int counter;


    /* Memilih masukan dari kartu yang akan dipakai */
    cout << "Input Keyboard? (Y/N)\n";
    cin >> opsiInput;
    do{
    if(opsiInput == 'Y'){

        cout << "Masukkan Input 4 Kartu\n";
```

```
        ws(cin);


        string S, T;

        getline(cin, S);

        stringstream X(S);

        counter = 0;
        while (getline(X, T, ' ')) {
            counter++;
        }

        if(counter != 4){
            cout << "Inputan tidak valid" << endl;
            continue;
        }

        else{
            stringstream Y(S);
            for(int i = 0; i < 4; i++){
                getline(Y, T, ' ');
                cardsChar[i] = T[0];
                cards[i] = convertToValue(cardsChar[i]);
            }
        }


    }
    else if(opsiInput == 'N'){
        srand(time(NULL));
        for(int i = 0; i < 4 ; i++){
            cards[i] = (rand())%13 + 1;
            cardsChar[i] = convertToChar(cards[i]);

        }
        cout << "Kartu yang dipakai adalah: " << cardsChar[0] << " " <<
cardsChar[1] << " " << cardsChar[2] << " " << cardsChar[3] << endl;
    }
    else{
        cout << "Mohon Masukan Input Yang Valid\n";
        cout << "Input Keyboard? (Y/N)\n";
        cin >> opsiInput;
        continue;
```

```
    }
    } while(isCardValid(cards) == false);


    /* Brute Force */

    /* Looping Cards*/
    for(int i = 0; i < 4; i++){
        for(int j = 0; j < 4; j++){
            if(j != i){
                for(int k = 0; k < 4; k++){
                    if(k != i && k != j){
                        for(int l = 0; l < 4; l++){
                            if(l != i && l != j && l != k){


                                /* Operation Loops*/
                                for(int m = 0; m < 4 ; m++){
                                    for(int n = 0; n < 4; n++){
                                        for(int o = 0; o < 4 ; o++){
                                            calculateCard(i,j,k,l,m,n,o,cards);
                                        }
                                    }
                                }
                            }
                        }


                    }
                }
            }
        }
    }
    if(jumlahSolusi == 0){
        cout << "Tidak ada solusi" << endl;
        cout << "Apakah Anda ingin menyimpannya?(Y/N)\n";
        cin >> opsiInput;
        if(opsiInput == 'Y'){
            cout << "Masukkan nama file: \n";
            string namaSolusi;
            ws(cin);
            getline(cin,namaSolusi);
            namaSolusi = "../test/" + namaSolusi + ".txt";
            int length = namaSolusi.length();
            char* namaSolusiChar = new char[length + 1];
```

```
            strcpy(namaSolusiChar, namaSolusi.c_str());
            ofstream file;

            file.open(namaSolusiChar);
            file << "Kartu yang dipakai adalah: " << cardsChar[0] << " " <<
cardsChar[1] << " " << cardsChar[2] << " " << cardsChar[3] << endl;
            file << "Tidak Ditemukan solusi" << endl;


            file.close();
            cout << "Solusi disimpan di " << namaSolusi << endl;
        }
        else if(opsiInput == 'N'){
            cout << "Solusi tidak disimpan" << endl;
        }
        else{
            cout << "Input tidak valid" << endl;
        }
    }
    else {
        cout << "Ditemukan " << jumlahSolusi << " solusi" << endl;
        cout << "Apakah Anda ingin menyimpan solusi?(Y/N)\n";
        cin >> opsiInput;
        if(opsiInput == 'Y'){
            cout << "Masukkan nama file solusi: \n";
            string namaSolusi;
            ws(cin);
            getline(cin,namaSolusi);
            namaSolusi = "../test/" + namaSolusi + ".txt";

            int length = namaSolusi.length();
            char* namaSolusiChar = new char[length + 1];
            strcpy(namaSolusiChar, namaSolusi.c_str());
            ofstream file;
            solusi = ss.str();
            file.open(namaSolusiChar);
            file << "Kartu yang dipakai adalah: " << cardsChar[0] << " " <<
cardsChar[1] << " " << cardsChar[2] << " " << cardsChar[3] << endl;
            file << "Ditemukan " << jumlahSolusi << " solusi" << endl;
            file << "Solusi: " << endl;
            file << solusi;
            file.close();
            cout << "Solusi disimpan di " << namaSolusi << endl;
        }
        else if(opsiInput == 'N'){
```

```
            cout << "Solusi tidak disimpan" << endl;
        }
        else{
            cout << "Input tidak valid" << endl;
        }
    }



}
```

BAB IV

PENGUJIAN

1. Case 1
   Tidak ditemukan solusi dan disimpan pada file.

```
Input Keyboard? (Y/N)
Y
Masukkan Input 4 Kartu
A 6 7 K
Tidak ada solusi
Apakah Anda ingin menyimpannya?(Y/N)
Y
Masukkan nama file:
case1
Solusi disimpan di case1.txt
```

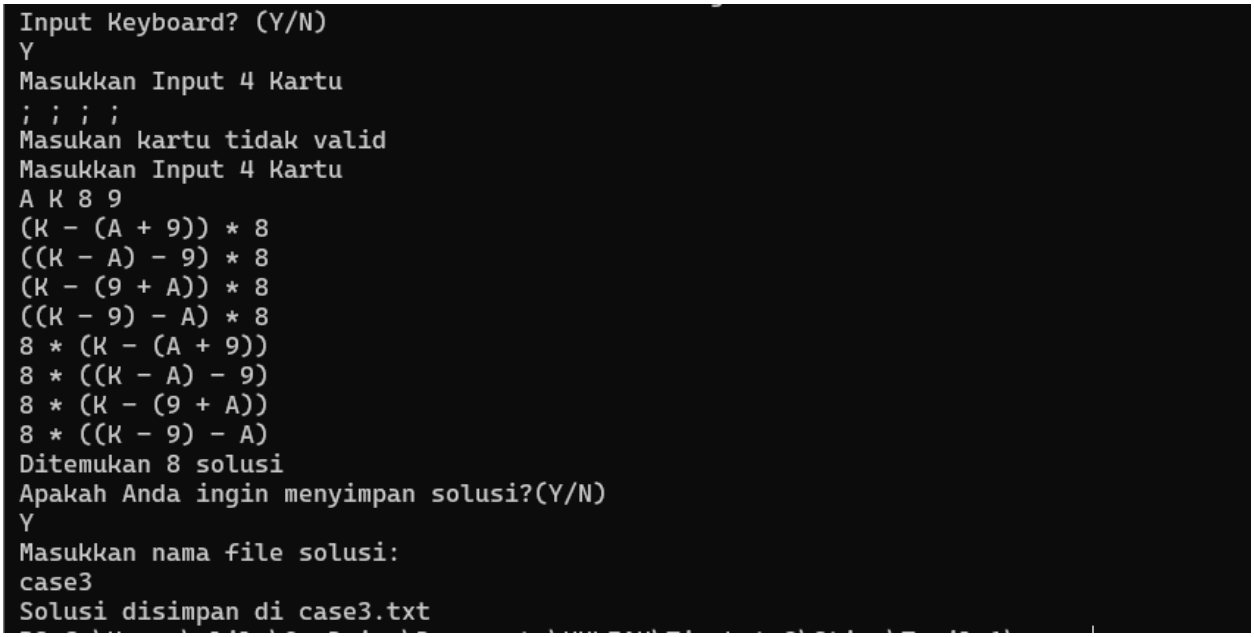
2. Case 2

Ditemukan banyak solusi dan disimpan pada file.

```
PS C:\Users\alilo\OneDrive\Documents\KULIAH\Tingkat 2\Stima\Tucil 1\src> ./main
Input Keyboard? (Y/N)
Y
Masukkan Input 4 Kartu
J 8 10 K
8 * ((A - J) + K)
8 * (A - (J - K))
8 * ((A + K) - J)
8 * (A + (K - J))
8 * ((K - J) + A)
8 * (K - (J - A))
8 * ((K + A) - J)
8 * (K + (A - J))
((A - J) + K) * 8
(A - (J - K)) * 8
((A + K) - J) * 8
(A + (K - J)) * 8
((K - J) + A) * 8
(K - (J - A)) * 8
((K + A) - J) * 8
(K + (A - J)) * 8
Ditemukan 16 solusi
Apakah Anda ingin menyimpan solusi?(Y/N)
Y
Masukkan nama file solusi:
case2
```

3. Case 3

Input kartu tidak sesuai



4. Case 4

Masukkan semua angka dan disimpan solusinya.

```
PS C:\Users\alilo\OneDrive\Documents\KULIAH\Tingkat 2\Stima\Tucil 1\src> ./main
Input Keyboard? (Y/N)
Y
Masukkan Input 4 Kartu
3 4 7 2
((3 + 7) * 2) + 4
4 + ((3 + 7) * 2)
4 + ((7 + 3) * 2)
4 * ((7 - 3) + 2)
4 * (7 - (3 - 2))
4 * ((7 + 2) - 3)
4 * (7 + (2 - 3))
4 + (2 * (3 + 7))
4 * ((2 - 3) + 7)
4 * (2 - (3 - 7))
4 + (2 * (7 + 3))
(4 + 2) * (7 - 3)
4 * ((2 + 7) - 3)
4 * (2 + (7 - 3))
(7 - 3) * (4 + 2)
((7 + 3) * 2) + 4
((7 - 3) + 2) * 4
(7 - (3 - 2)) * 4
(7 - 3) * (2 + 4)
((7 + 2) - 3) * 4
(7 + (2 - 3)) * 4
((2 - 3) + 7) * 4
(2 - (3 - 7)) * 4
(2 * (3 + 7)) + 4
(2 + 4) * (7 - 3)
((2 + 7) - 3) * 4
(2 + (7 - 3)) * 4
(2 * (7 + 3)) + 4
Ditemukan 28 solusi
Apakah Anda ingin menyimpan solusi?(Y/N)
Y
Masukkan nama file solusi:
case4
Solusi disimpan di case4.txt
```

5. Case 5

Kartu dipilih acak dan disimpan pada file

```
PS C:\Users\alilo\OneDrive\Documents\KULIAH\Tingkat 2\Stima\Tucil 1\src> ./main
Input Keyboard? (Y/N)
N
Kartu yang dipakai adalah: 5 8 2 5
((5 / 5) + 2) * 8
8 * ((5 / 5) + 2)
8 * (2 + (5 / 5))
8 * (2 + (5 / 5))
8 * ((5 / 5) + 2)
(2 + (5 / 5)) * 8
(2 + (5 / 5)) * 8
((5 / 5) + 2) * 8
Ditemukan 8 solusi
Apakah Anda ingin menyimpan solusi?(Y/N)
Y
Masukkan nama file solusi:
case5
Solusi disimpan di case5.txt
```

6. Case 6

Kartu dipilih acak dan tidak disimpan pada file.

```
PS C:\Users\alilo\OneDrive\Documents\KULIAH\Tingkat 2\Stima\Tucil 1\src> ./main
Input Keyboard? (Y/N)
N
Kartu yang dipakai adalah: Q 6 4 7
Q * (6 / (7 - 4))
(Q * 6) / (7 - 4)
(Q / (7 - 4)) * 6
Q / ((7 - 4) / 6)
6 * (Q / (7 - 4))
(6 * Q) / (7 - 4)
6 * (7 - (Q / 4))
(6 / (7 - 4)) * Q
6 / ((7 - 4) / Q)
(7 - (Q / 4)) * 6
Ditemukan 10 solusi
Apakah Anda ingin menyimpan solusi?(Y/N)
N
Solusi tidak disimpan
```

Point	Ya	Tidak
Program berhasil dikompilasi tanpa kesalahan	**V**	
Program berhasil *running*	**V**	
Program dapat membaca input / generate sendiri dan memberikan luaran	**V**	
Solusi yang diberikan program memenuhi (berhasil mencapai 24)	**V**	
Program dapat menyimpan solusi dalam file teks	**V**	

BAB V

LINK REPOSITORY

semifinal-com/Tucil1_13521128 (github.com)