

Tugas Kecil 1 IF2211 Strategi Algoritma

Semester II tahun 2022/2023

Penyelesaian *24 Card Game Problem* Menggunakan Algoritma Brute Force

Disusun oleh:

Muhammad Haidar Akita Tresnadi 13521025



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

2023

BAB I

ALGORITMA BRUTE FORCE

Brute force merupakan algoritma yang memiliki prinsip utama yaitu mencoba segala kemungkinan yang ada hingga ditemukan solusi yang tepat. Untuk menyelesaikan *24 card game problem* menggunakan algoritma brute force, maka semua kemungkinan yang bisa dicoba adalah kemungkinan – kemungkinan dalam kombinasi 3 operasi bilangan sehingga hasil 24 dapat dibentuk. Operasi bilangan yang digunakan pada metode brute force ini adalah +, -, *, / (plus, minus, kali, dan bagi). Dalam algoritma ini, penulis menggunakan 5 tata peletakan kurung sehingga kurang lebih operasi yang dilakukan untuk mencapai hasil 24 adalah sebagai berikut :

1. $(W \text{ op } X) \text{ op } (Y \text{ op } Z)$
2. $((W \text{ op } X) \text{ op } Y) \text{ op } Z$
3. $(W \text{ op } (X \text{ op } Y)) \text{ op } Z$
4. $W \text{ op } ((X \text{ op } Y) \text{ op } Z)$
5. $W \text{ op } (X \text{ op } (Y \text{ op } Z))$

Dalam algoritma yang dibuat penulis, hanya 5 jenis tata peletakan kurung yang direalisasikan di mana op merupakan operasi bilangan dan W,X,Y,Z merupakan angka di antara 1 hingga 13. Berdasarkan tata peletakan tersebut, maka bila total kombinasi dari 4 operasi bilangan yang mungkin adalah 64, dan terdapat 24 kombinasi 4 bilangan yang mungkin, didapat bahwa diperlukan sekitar 7680 pengecekan solusi.

SOURCE CODE PROGRAM

- Library dan fungsi untuk melakukan operasi bilangan

```
#include <iostream>
#include <string>
#include <bits/stdc++.h>
#include <vector>
#include <windows.h>

using namespace std;

double operate(double num1, double num2, char op){
    if (op == '*'){
        return num1 * num2;
    } else if (op == '+'){
        return num1 + num2;
    } else if (op == '-'){
        return num1 - num2;
    } else {
        return num1 / num2;
    }
}
```

- Program utama

[illegible]

```

if (pilihan == "1"){
    cout << "    > ";
    for (int i = 0; i < 4; i++){
        cin >> angka[i];
        while (find(begin(validate), end(validate), angka[i]) == end(validate)){
            cout << "    Invalid input" << endl;
            cout << "    > ";
            cin >> angka[i];
        }
    }
    cout << angka[0] << " " << angka[1] << " " << angka[2] << " " << angka[3] << endl;
    for (int i = 0; i < 4; i++){
        if (angka[i] == "J" || angka[i] == "j"){
            angka_akhhir[i] = 11;
        } else if (angka[i] == "Q" || angka[i] == "q"){
            angka_akhhir[i] = 12;
        } else if (angka[i] == "K" || angka[i] == "k"){
            angka_akhhir[i] = 13;
        } else if (angka[i] == "A" || angka[i] == "a"){
            angka_akhhir[i] = 1;
        } else {
            angka_akhhir[i] = stod(angka[i]);
        }
    }
}

```

```

} else if (pilihan == "2"){
    srand(time(0));
    for (int i = 0; i < 4; i++){
        angka_akhhir[i] = (rand() % 13) + 1;
        if (angka_akhhir[i] == 13){
            angka[i] = "K";
        } else if (angka_akhhir[i] == 12){
            angka[i] = "Q";
        } else if (angka_akhhir[i] == 11){
            angka[i] = "J";
        } else if (angka_akhhir[i] == 1){
            angka[i] = "A";
        } else {
            angka[i] = to_string(angka_akhhir[i]);
        }
    }
    cout << angka[0] << " " << angka[1] << " " << angka[2] << " " << angka[3] << endl;
} else if (pilihan == "3"){
    exit = true;
    cout << "    Have a great day ^^" << endl;
}
else {
    cout << "Invalid input" << endl;
}
}

```

- Proses algoritma brute force untuk mendapatkan seluruh kombinasi angka dan operasi

```

if (pilihan == "1" || pilihan == "2"){
    flag = false;
    start = clock();
    for (int i = 0; i < 4; i++){
        for (int j = 0; j < 4; j++){
            for (int k = 0; k < 4; k++){
                for (int l = 0; l < 4; l++){
                    if (i != j && j != k && i != k && j != l && k != l && i != l){
                        for (int m = 0; m < 4; m++){
                            for (int n = 0; n < 4; n++){
                                for (int o = 0; o < 4; o++){
                                    // (a op b) op (c op d)
                                    if (operate(operate(angka_akhhir[i], angka_akhhir[j], operators[m]), operate(angka_akhhir[k],
                                        angka_akhhir[l], operators[o]), operators[n]) == 24) {
                                        hasil = "(" + to_string(angka_akhhir[i]) + " " + operators[m] + " " + to_string(angka_akhhir[j]) +
                                            " " + operators[n] + " (" + to_string(angka_akhhir[k]) + " " + operators[o] + " " + to_string
                                                (angka_akhhir[l]) + ")";
                                        history.push_back(hasil);
                                    }
                                }
                            }
                        }
                    }
                }
            }
        }
    }
    // ((a op b) op c) op d
    if (operate(operate(operate(angka_akhhir[i], angka_akhhir[j], operators[m]), angka_akhhir[k], operators
        [n]), angka_akhhir[l], operators[o]) == 24) {
        hasil = "((( " + to_string(angka_akhhir[i]) + " " + operators[m] + " " + to_string(angka_akhhir[j])
            + " " + operators[n] + " " + to_string(angka_akhhir[k]) + " " + operators[o] + " " + to_string
                (angka_akhhir[l]) + ")))";
        history.push_back(hasil);
    }
}

```

```

// (a op (b op c)) op d
if (operate(operate(angka_akhhir[i], operate(angka_akhhir[j], angka_akhhir[k], operators[n]), operators
    [m]), angka_akhhir[l], operators[o]) == 24) {
    hasil = "(" + to_string(angka_akhhir[i]) + " " + operators[m] + " (" + to_string(angka_akhhir[j])
        + " " + operators[n] + " " + to_string(angka_akhhir[k]) + " " + operators[o] + " " + to_string
            (angka_akhhir[l]) + ")";
    history.push_back(hasil);
}

// a op ((b op c) op d)
if (operate(angka_akhhir[i], operate(operate(angka_akhhir[j], angka_akhhir[k], operators[n]),
    angka_akhhir[l], operators[o]), operators[m]) == 24) {
    hasil = to_string(angka_akhhir[i]) + " " + operators[m] + " (( " + to_string(angka_akhhir[j]) + " "
        + operators[n] + " " + to_string(angka_akhhir[k]) + " " + operators[o] + " " + to_string
            (angka_akhhir[l]) + ")))";
    history.push_back(hasil);
}

// a op (b op (c op d))
if (operate(angka_akhhir[i], operate(angka_akhhir[j], operate(angka_akhhir[k], angka_akhhir[l],
    operators[o]), operators[n]), operators[m]) == 24) {
    hasil = to_string(angka_akhhir[i]) + " " + operators[m] + " (" + to_string(angka_akhhir[j]) + " "
        + operators[n] + " (" + to_string(angka_akhhir[k]) + " " + operators[o] + " " + to_string
            (angka_akhhir[l]) + ")))";
    history.push_back(hasil);
}

```


- Proses pengeluaran output solusi dan penyimpanan solusi ke dalam file

```
ends = clock();
double time_taken = double(ends - start) / double(CLOCKS_PER_SEC);
cout << "Time taken : " << time_taken << setprecision(5) << " sec " << endl;
if (history.size() == 0){
    cout << "No Solution!" << endl;
}
else{
    sort( history.begin(), history.end() );
    history.erase( unique( history.begin(), history.end() ), history.end() );
    cout << history.size() << " Solution found!" << endl;
    for (int i = 0; i < history.size(); i++){
        cout << history[i] << endl;
    }

    // save file
    while(!flag){
        cout << "Apakah ingin menyimpan solusi ? (y/n)" << endl;
        cout << "    > ";
        cin >> pilihan;
        if (pilihan == "y" || pilihan == "Y"){
            flag = true;
            cout << "Masukkan nama file untuk menyimpan hasil solusi!" << endl;
            cout << "    > ";
            cin >> file_name;
            cout << "Saving file..." << endl;
            ofstream save_file ("../test/" + file_name + ".txt");
            for (int i = 0; i < history.size(); i++){
                save_file << history[i] << endl;
            }
            save_file.close();
            Sleep(2000);
        }
    }
}
```

```
} else if (pilihan == "n" || pilihan == "N"){
    flag = true;
    cout << "File tidak disimpan!" << endl;
} else {
    cout << "Invalid input" << endl;
}
```

BAB III

TEST CASE PROGRAM

3.1 Input by user

- TC1 J Q 2 4

```
24 Game solver
1. Input by user
2. Random generate number
3. Exit
> 1
> J Q 2 4
J Q 2 4
Time taken : 0.001 sec
2 Solution found!
(( 11 - 2 ) * 4 ) - 12
(4 * ( 11 - 2 )) - 12
Apakah ingin menyimpan solusi ? (y/n)
> ☐
```

- TC2 3 5 9 10

```
> 3 5 9 10
3 5 9 10
Time taken : 0.001 sec
16 Solution found!
(( 10 - 5 ) * 3 ) + 9
(( 3 + 9 ) * 10 ) / 5
(( 9 + 3 ) * 10 ) / 5
(10 * ( 3 + 9 )) / 5
(10 * ( 9 + 3 )) / 5
(10 / 5 ) * ( 3 + 9 )
(10 / 5 ) * ( 9 + 3 )
(3 * ( 10 - 5 )) + 9
(3 + 9 ) * (10 / 5)
(3 + 9 ) / (5 / 10)
(9 + 3 ) * (10 / 5)
(9 + 3 ) / (5 / 10)
9 + ( 3 * ( 10 - 5 ))
9 + (( 10 - 5 ) * 3)
9 - ( 3 * ( 5 - 10 ))
9 - (( 5 - 10 ) * 3)
Apakah ingin menyimpan solusi ? (y/n)
> ☐
```

- TC3 4 A 10 K

```
> 4 A 10 K
4 A 10 K
Time taken : 0.003 sec
No Solution!
```

3.2 Random Generate Number

- TC4 12 4 10 10

```
24 Game solver
1. Input by user
2. Random generate number
3. Exit
> 2
2 4 10 10
Time taken : 0.001 sec
No Solution!
```

- TC5 K 9 10 9

```
> 2
K 9 10 9
Time taken : 0.002 sec
12 Solution found!
(( 9 / 9 ) + 10 ) + 13
(( 9 / 9 ) + 13 ) + 10
(10 + ( 9 / 9 )) + 13
(10 + 13 ) + ( 9 / 9 )
(13 + ( 9 / 9 )) + 10
(13 + 10 ) + ( 9 / 9 )
( 9 / 9 ) + (10 + 13)
( 9 / 9 ) + (13 + 10)
10 + ( 13 + ( 9 / 9 ))
10 + (( 9 / 9 ) + 13)
13 + ( 10 + ( 9 / 9 ))
13 + (( 9 / 9 ) + 10)
```

- TC6 6 2 6 7

```
> 2
6 2 6 7
Time taken : 0.002 sec
8 Solution found!
(( 6 * 7 ) + 6 ) / 2
(( 7 * 6 ) + 6 ) / 2
(( 7 - 2 ) * 6 ) - 6
( 6 * ( 7 - 2 ) ) - 6
( 6 + ( 6 * 7 ) ) / 2
( 6 + ( 7 * 6 ) ) / 2
( 7 - ( 6 / 2 ) ) * 6
6 * ( 7 - ( 6 / 2 ) )
```


LINK REPOSITORY

CHECKLIST

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	✓	