

**LAPORAN
TUGAS KECIL 1 IF2211
STRATEGI ALGORITMA**

**Penyelesaian Permainan Kartu 24 dengan Algoritma Brute
Force**



Arsa Izdiyar Islam

13521101

**SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
BANDUNG
2022/2023**

BAGIAN 1

ALGORITMA *BRUTE FORCE*

Langkah-langkah algoritma *brute force (exhaustive search)* untuk mencari penyelesaian permainan kartu 24 adalah sebagai berikut:

1. Ambil masukan dari user untuk 4 kartu yang akan dicari solusinya (bisa dengan input manual atau ditentukan secara acak).
2. Menentukan hasil permutasi semua urutan angka yang mungkin. Karena terdapat 4 angka, maka total urutan angka yang mungkin adalah sejumlah $4!$. Akan tetapi, kasus saat ada angka yang sama perlu ditangani sehingga tidak ada kemungkinan yang berulang.
3. Menentukan permutasi semua operasi yang digunakan dan urutannya yang mungkin. Karena akan selalu terdapat 3 operasi dan 4 macam operasi (+, -, *, /) maka jumlah permutasinya adalah 4^3 .
4. Menentukan semua jenis pengurungan ekspresi matematika yang mungkin. Terdapat 5 macam pengurungan untuk 4 angka dan 3 operasi, yaitu $((x \text{ op } x) \text{ op } x) \text{ op } x$, $(x \text{ op } (x \text{ op } x)) \text{ op } x$, $(x \text{ op } x) \text{ op } (x \text{ op } x)$, $x \text{ op } ((x \text{ op } x) \text{ op } x)$, dan $x \text{ op } (x \text{ op } (x \text{ op } x))$. Alhasil, dengan tiga poin kemungkinan ini, jumlah kemungkinan terbanyak adalah $4! \times 4^3 \times 5$ kemungkinan.
5. Setelah seluruh ekspresi yang mungkin dari 4 kartu sudah dibuat, lakukan pengecekan hasil semua ekspresi dan catat semua ekspresi yang menghasilkan angka 24 dalam bentuk string.
6. Setelah semua solusi ditemukan, tampilkan semua string solusi tersebut.

BAGIAN 2

SOURCE PROGRAM

Berikut adalah *source* program untuk penyelesaian permainan kartu 24 dengan algoritma *brute force* dalam bahasa C++:

```
#include <iostream>
#include <time.h>
#include <string>
#include <vector>
#include <array>
#include <fstream>
#include <chrono>
using namespace std;

string inputs[] = {"A", "2", "3", "4", "5", "6", "7", "8", "9",
"10", "J", "Q", "K"};
char ops[] = {'*', '/', '+', '-'};

void get_input(double arr[4])
{
    // menerima input pilihan dari user
    string raw[4];
    while (true)
    {
        cout << "Pilih opsi untuk masukkan angka:\n"
              << "1. Input manual\n2. Dipilih secara random\n";
        cout << "Masukkan angka 1 atau 2: ";
        int chosen;
        cin >> chosen;
        cin.ignore();

        // validasi input pilihan
        if (chosen < 1 || chosen > 2)
        {
            cout << "Pilihan tidak valid. Silahkan coba lagi.\n";
            continue;
        }

        if (chosen == 1)
        {
            // mengambil masukan 4 kartu dari user (manual)
            while (true)
```

```

{
    bool input_valid = true;
    cout << "Masukkan 4 angka/huruf yang terdiri dari (A, 2, 3,
4, 5, 6, 7, 8, 9, 10, J, Q, K) dipisah dengan spasi:\n";
    for (int i = 0; i < 4; i++)
    {
        cin >> raw[i];
    }
    cin.ignore();
    for (int i = 0; i < 4; i++)
    {
        bool valid = false;
        for (int j = 0; !valid && j < 13; j++)
        {
            if (raw[i] == inputs[j])
            {
                arr[i] = (double)(j + 1);
                valid = true;
            }
        }
        if (!valid)
        {
            cout << "Masukkan tidak valid. Silahkan coba lagi\n";
            input_valid = false;
            break;
        }
    }
    if (input_valid)
        break;
}
}
else
{
    // mengambil 4 kartu secara random
    for (int i = 0; i < 4; i++)
    {
        int rand_num = rand() % 13 + 1;
        arr[i] = rand_num;
        raw[i] = inputs[rand_num - 1];
    }
}
}

```

```

    printf("Kartu yang terpilih: %s, %s, %s, %s\n", raw[0].c_str(),
raw[1].c_str(), raw[2].c_str(), raw[3].c_str());
    break;
}
}

double count_by_op(double a, double b, char op)
{
    switch (op)
    {
        case '+':
            return a + b;
        case '-':
            return a - b;
        case '*':
            return a * b;
        default:
            return a / b;
    }
}

void try_every_paranthesis(vector<string> *answers, double nums[4],
char ops[3])
{
    // semua peletakan kurung yang mungkin
    // (x op x) op (x op x), ((x op x) op x) op x, (x op (x op x)) op
x, x op ((x op x) op x), dan x op (x op (x op x))
    int paranthesis[5][2][2] = {
        {{0, 1}, {2, 3}},
        {{0, 1}, {0, 2}},
        {{1, 2}, {0, 2}},
        {{1, 2}, {1, 3}},
        {{2, 3}, {1, 3}}};

    // mencoba semua kurung
    for (int i = 0; i < 5; i++)
    {
        int *par1 = paranthesis[i][0];
        int *par2 = paranthesis[i][1];
        int ops_idx[3];

```

```

ops_idx[0] = par1[1] - 1; // operasi pertama merupakan operasi
yang ada pada kurung pertama
if (par1[1] == par2[1]) // operasi kedua merupakan operasi
yang ada pada kurung kedua
    ops_idx[1] = par2[0];
else
    ops_idx[1] = par2[1] - 1;

ops_idx[2] = 3 - ops_idx[0] - ops_idx[1]; // operasi ketiga
merupakan operasi yang tersisa

// menghitung operasi pertama
double res1 = count_by_op(nums[par1[0]], nums[par1[1]],
ops[ops_idx[0]]);
bool dec1 = false;
bool dec2 = false;

// menentukan 3 angka hasil operasi pertama
double next1[3];
for (int i = 0; i < 3; i++)
{
    if (i < par1[0])
        next1[i] = nums[i];
    else if (i < par1[1])
        next1[i] = res1;
    else
        next1[i] = nums[i + 1];
}

if (par2[0] >= par1[1])
{
    par2[0]--;
    dec1 = true;
}
if (par2[1] >= par1[1])
{
    par2[1]--;
    dec2 = true;
}

// menghitung operasi kedua

```

```

    double res2 = count_by_op(next1[par2[0]], next1[par2[1]],
ops[ops_idx[1]]);

// menentukan 2 angka hasil operasi kedua
double next2[2];
for (int i = 0; i < 2; i++)
{
    if (i < par2[0])
    {
        next2[i] = next1[i];
    }
    else if (i < par2[1])
    {
        next2[i] = res2;
    }
    else
    {
        next2[i] = next1[i + 1];
    }
}
if (dec1)
    par2[0]++;
if (dec2)
    par2[1]++;

// hasil akhir
double res3 = count_by_op(next2[0], next2[1], ops[ops_idx[2]]);

// jika hasil akhir adalah 24, tambahkan jawaban ke list
jawaban
if (res3 == 24)
{
    string ans = "";
    for (int i = 0; i < 4; i++)
    {
        if (i == par1[0])
        {
            ans += "(";
        }
        if (i == par2[0])
        {

```

```

        ans += "(";
    }
    ans += to_string((int)nums[i]);
    if (i == par1[1])
    {
        ans += ")";
    }
    if (i == par2[1])
    {
        ans += ")";
    }
    if (i < 3)
    {
        ans += " " + string(1, ops[i]) + " ";
    }
}
answers->push_back(ans);
}
}

void show_answers(double nums[4])
{
    auto start = chrono::high_resolution_clock::now();
    vector<array<int, 4>> possibilities; // semua urutan indeks angka
yang mungkin
    vector<array<char, 3>> poss_ops;    // semua operasi yang
mungkin
    vector<string> answers;             // untuk menyimpan semua
solusi

    // mencari semua kemungkinan urutan indeks angka
    for (int i = 0; i < 4; i++)
    {
        for (int j = 0; j < 4; j++)
        {
            if (i == j)
                continue;
            for (int k = 0; k < 4; k++)
            {
                if (k == i || k == j)

```



```

        continue;
    for (int l = 0; l < 4; l++)
    {
        if (l == i || l == j || l == k)
            continue;
        array<int, 4> arr = {i, j, k, l};

        // mengecek duplikat angka
        bool valid = true;
        for (int i = 0; i < 3 && valid; i++)
        {
            for (int j = i + 1; j < 4 && valid; j++)
            {
                if (arr[i] > arr[j] && nums[arr[i]] == nums[arr[j]])
                {
                    valid = false;
                }
            }
        }

        if (valid)
            possibilities.push_back(arr);
    }
}

// mencari semua kemungkinan operasi
for (int i = 0; i < 4; i++)
{
    for (int j = 0; j < 4; j++)
    {
        for (int k = 0; k < 4; k++)
        {
            array<char, 3> arr = {ops[i], ops[j], ops[k]};
            poss_ops.push_back(arr);
        }
    }
}

```

```

// mencoba semua kemungkinan berdasarkan urutan angka, operasi,
dan kurung
for (int i = 0; i < possibilities.size(); i++)
{
    for (int j = 0; j < poss_ops.size(); j++)
    {
        double cur_nums[4] = {
            nums[possibilities[i][0]],
            nums[possibilities[i][1]],
            nums[possibilities[i][2]],
            nums[possibilities[i][3]]};
        char cur_ops[3] = {
            poss_ops[j][0],
            poss_ops[j][1],
            poss_ops[j][2]};
        try_every_paranthesis(&answers, cur_nums, cur_ops);
    }
}

// menghitung waktu eksekusi
auto end = chrono::high_resolution_clock::now();
auto duration = chrono::duration_cast<chrono::microseconds>(end -
start);
double duration_sec = (double)duration.count() / 1000000;

// menampilkan hasil
if (answers.size() == 0)
{
    cout << "No solution found\n";
    cout << "Execution time: " << duration_sec << " seconds\n";
    return;
}

cout << answers.size() << " solutions found\n";
for (int i = 0; i < answers.size(); i++)
{
    cout << answers[i] << endl;
}
cout << "Execution time: " << duration_sec << " seconds\n";

// mengambil input dari user untuk menyimpan solusi/tidak

```

```

char ans;
cout << "Apakah ingin menyimpan solusi (y/N)? ";
cin >> ans;
cin.ignore();
if (ans == 'y' || ans == 'Y')
{
    // menyimpan solusi ke file
    string name;
    cout << "Masukkan nama file: ";
    getline(cin, name);
    ofstream file("test/" + name);
    file << nums[0] << " " << nums[1] << " " << nums[2] << " " <<
nums[3] << endl;
    file << answers.size() << " solutions found\n";
    for (int i = 0; i < answers.size(); i++)
    {
        file << answers[i] << endl;
    }
    file.close();
    cout << "Solusi berhasil disimpan ke file test/" << name <<
endl;
}
}

int main()
{
    // set random seed
    srand(time(NULL));

    // jalankan program
    double nums[4];
    get_input(nums);
    show_answers(nums);
    return 0;
}

```

BAGIAN 3

HASIL INPUT/OUTPUT

1. Angka acak

```
C:\code\kuliah\sem4\stima\tucil1>bin\main.exe
Pilih opsi untuk masukkan angka:
1. Input manual
2. Dipilih secara random
Masukkan angka 1 atau 2: 2
Kartu yang terpilih: 7, 8, 10, 6
30 solutions found
(7 * (8 - 6)) + 10
((7 - 10) + 6) * 8
(7 - (10 - 6)) * 8
(7 * 6) - (8 + 10)
((7 * 6) - 8) - 10
(7 * 6) - (10 + 8)
((7 * 6) - 10) - 8
((7 + 6) - 10) * 8
(7 + (6 - 10)) * 8
8 * ((7 - 10) + 6)
8 * (7 - (10 - 6))
8 * ((7 + 6) - 10)
8 * (7 + (6 - 10))
8 * ((6 + 7) - 10)
8 * (6 + (7 - 10))
((8 - 6) * 7) + 10
8 * ((6 - 10) + 7)
8 * (6 - (10 - 7))
10 + (7 * (8 - 6))
10 - (7 * (6 - 8))
10 + ((8 - 6) * 7)
10 - ((6 - 8) * 7)
(6 * 7) - (8 + 10)
((6 * 7) - 8) - 10
(6 * 7) - (10 + 8)
((6 * 7) - 10) - 8
((6 + 7) - 10) * 8
(6 + (7 - 10)) * 8
((6 - 10) + 7) * 8
(6 - (10 - 7)) * 8
Execution time: 0.000347 seconds
Apakah ingin menyimpan solusi (y/N)? y
Masukkan nama file: test1.txt
Solusi berhasil disimpan ke file test/test1.txt

C:\code\kuliah\sem4\stima\tucil1>
```

2. Seluruh angka sama (6, 6, 6, 6)

```
C:\code\kuliah\sem4\stima\tucil1>bin\main.exe
Pilih opsi untuk masukkan angka:
1. Input manual
2. Dipilih secara random
Masukkan angka 1 atau 2: 1
Masukkan 4 angka/huruf yang terdiri dari (A, 2, 3, 4, 5, 6, 7, 8, 9, 10, J, Q, K) dipisah dengan spasi:
6 6 6 6
Kartu yang terpilih: 6, 6, 6, 6
7 solutions found
(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 + (6 + (6 + 6))
Execution time: 3.8e-05 seconds
Apakah ingin menyimpan solusi (y/N)? y
Masukkan nama file: test2.txt
Solusi berhasil disimpan ke file test/test2.txt

C:\code\kuliah\sem4\stima\tucil1>
```

3. Dua pasang angka sama (4, 4, 2, 2)

```
C:\code\kuliah\sem4\stima\tucil1>bin\main.exe
Pilih opsi untuk masukkan angka:
1. Input manual
2. Dipilih secara random
Masukkan angka 1 atau 2: 1
Masukkan 4 angka/huruf yang terdiri dari (A, 2, 3, 4, 5, 6, 7, 8, 9, 10, J, Q, K) dipisah dengan spasi:
4 4 2 2
Kartu yang terpilih: 4, 4, 2, 2
12 solutions found
4 * ((4 * 2) - 2)
(4 + (4 * 2)) * 2
4 * ((2 * 4) - 2)
((4 * 2) + 4) * 2
(4 + (2 * 4)) * 2
((4 * 2) - 2) * 4
((2 * 4) + 4) * 2
2 * (4 + (4 * 2))
2 * ((4 * 2) + 4)
2 * (4 + (2 * 4))
((2 * 4) - 2) * 4
2 * ((2 * 4) + 4)
Execution time: 0.000123 seconds
Apakah ingin menyimpan solusi (y/N)? y
Masukkan nama file: test3.txt
Solusi berhasil disimpan ke file test/test3.txt

C:\code\kuliah\sem4\stima\tucil1>
```

4. Tiga angka sama (A, A, A, Q)

```
C:\code\kuliah\sem4\stima\tucil1>bin\main.exe
Pilih opsi untuk masukkan angka:
1. Input manual
2. Dipilih secara random
Masukkan angka 1 atau 2: 1
Masukkan 4 angka/huruf yang terdiri dari (A, 2, 3, 4, 5, 6, 7, 8, 9,
10, J, Q, K) dipisah dengan spasi:
A A A Q
Kartu yang terpilih: A, A, A, Q
28 solutions found
((1 * 1) + 1) * 12
(1 * (1 + 1)) * 12
1 * ((1 + 1) * 12)
((1 / 1) + 1) * 12
(1 + 1) * (1 * 12)
((1 + 1) * 1) * 12
(1 + (1 * 1)) * 12
((1 + 1) / 1) * 12
(1 + (1 / 1)) * 12
(1 + 1) / (1 / 12)
(1 + 1) * (12 * 1)
((1 + 1) * 12) * 1
(1 + 1) * (12 / 1)
((1 + 1) * 12) / 1
(1 * 12) * (1 + 1)
1 * (12 * (1 + 1))
(12 * 1) * (1 + 1)
12 * ((1 * 1) + 1)
12 * (1 * (1 + 1))
12 * ((1 / 1) + 1)
(12 * (1 + 1)) * 1
12 * ((1 + 1) * 1)
12 * (1 + (1 * 1))
(12 * (1 + 1)) / 1
12 * ((1 + 1) / 1)
12 * (1 + (1 / 1))
(12 / 1) * (1 + 1)
12 / (1 / (1 + 1))
Execution time: 0.000101 seconds
Apakah ingin menyimpan solusi (y/N)? y
Masukkan nama file: test4.txt
Solusi berhasil disimpan ke file test/test4.txt

C:\code\kuliah\sem4\stima\tucil1>
```

5. Tidak ada solusi

```
C:\code\kuliah\sem4\stima\tucil1>bin\main.exe
Pilih opsi untuk masukkan angka:
1. Input manual
2. Dipilih secara random
Masukkan angka 1 atau 2: 1
Masukkan 4 angka/huruf yang terdiri dari (A, 2, 3, 4, 5, 6, 7, 8, 9,
10, J, Q, K) dipisah dengan spasi:
J K A 3
Kartu yang terpilih: J, K, A, 3
No solution found
Execution time: 0.000323 seconds

C:\code\kuliah\sem4\stima\tucil1>
```

6. Opsi tidak valid

```
C:\code\kuliah\sem4\stima\tucil1>bin\main.exe
Pilih opsi untuk masukkan angka:
1. Input manual
2. Dipilih secara random
Masukkan angka 1 atau 2: 3
Pilihan tidak valid. Silahkan coba lagi.
Pilih opsi untuk masukkan angka:
1. Input manual
2. Dipilih secara random
Masukkan angka 1 atau 2: 2
Kartu yang terpilih: 7, 10, 8, J
16 solutions found
((10 + 11) / 7) * 8
(10 + 11) / (7 / 8)
(10 + 11) * (8 / 7)
((10 + 11) * 8) / 7
(8 / 7) * (10 + 11)
8 / (7 / (10 + 11))
(8 / 7) * (11 + 10)
8 / (7 / (11 + 10))
(8 * (10 + 11)) / 7
8 * ((10 + 11) / 7)
(8 * (11 + 10)) / 7
8 * ((11 + 10) / 7)
((11 + 10) / 7) * 8
(11 + 10) / (7 / 8)
(11 + 10) * (8 / 7)
((11 + 10) * 8) / 7
Execution time: 0.000344 seconds
Apakah ingin menyimpan solusi (y/N)? y
Masukkan nama file: test6.txt
Solusi berhasil disimpan ke file test/test6.txt
```

7. Kartu tidak valid

```
C:\code\kuliah\sem4\stima\tucil1>bin\main.exe
Pilih opsi untuk masukkan angka:
1. Input manual
2. Dipilih secara random
Masukkan angka 1 atau 2: 1
Masukkan 4 angka/huruf yang terdiri dari (A, 2, 3, 4, 5, 6, 7, 8, 9,
10, J, Q, K) dipisah dengan spasi:
11 12 H W
Masukkan tidak valid. Silahkan coba lagi
Masukkan 4 angka/huruf yang terdiri dari (A, 2, 3, 4, 5, 6, 7, 8, 9,
10, J, Q, K) dipisah dengan spasi:
A A A A
Kartu yang terpilih: A, A, A, A
No solution found
Execution time: 3.1e-05 seconds
```

8. Solusi banyak (sis input di file test/test8.txt repository)

```
C:\code\kuliah\sem4\stima\tucil1>bin\main.exe
Pilih opsi untuk masukkan angka:
1. Input manual
2. Dipilih secara random
Masukkan angka 1 atau 2: 1
Masukkan 4 angka/huruf yang terdiri dari (A, 2, 3, 4, 5, 6, 7, 8, 9,
10, J, Q, K) dipisah dengan spasi:
2 A 8 6
Kartu yang terpilih: 2, A, 8, 6
166 solutions found
(2 * (1 + 8)) + 6
(2 * (8 + 1)) + 6
(1 / 2) * (8 * 6)
((1 / 2) * 8) * 6
(1 / (2 / 8)) * 6
1 / (2 / (8 * 6))
1 / ((2 / 8) / 6)
(1 / 2) * (6 * 8)
((1 / 2) * 6) * 8
(1 / (2 / 6)) * 8
1 / (2 / (6 * 8))
1 / ((2 / 6) / 8)
((1 * 8) / 2) * 6
(1 * (8 / 2)) * 6
1 * ((8 / 2) * 6)
(1 * 8) / (2 / 6)
1 * (8 / (2 / 6))
((1 + 8) * 2) + 6
(1 * 8) * (6 / 2)
((1 * 8) * 6) / 2
(1 * (8 * 6)) / 2
1 * ((8 * 6) / 2)
1 * (8 * (6 / 2))
((1 * 6) / 2) * 8
(1 * (6 / 2)) * 8
1 * ((6 / 2) * 8)
(1 * 6) / (2 / 8)
1 * (6 / (2 / 8))
(1 * 6) * (8 / 2)
((1 * 6) * 8) / 2
(1 * (6 * 8)) / 2
1 * ((6 * 8) / 2)
1 * (6 * (8 / 2))
(8 / 2) * (1 * 6)
((8 / 2) * 1) * 6
(8 / (2 * 1)) * 6
```


BAGIAN 4 LINK REPOSITORY

Berikut link repository Github: https://github.com/arsaizdihar/Tucil1_13521101

BAGIAN 5 *CHECKLIST PROGRAM*

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 program memenuhi (berhasil mencapai 24)	✓	
5. Program dapat menyimpan solusi dalam file teks	✓	