Tugas Kecil 1 IF 2211 Strategi Algoritma

Penyelesaian Permainan Kartu 24 dengan Algoritma Brute Force

Hosea Nathanael Abetnego 13521057

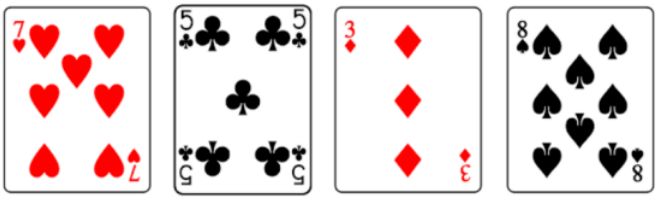
Teknik Informatika

Sekolah Teknik Elektro dan Informatika

Institut Teknologi Bandung

2023

I. Deskripsi Persoalan



MAKE IT 24

Permainan kartu 24 adalah permainan kartu aritmatika dengan tujuan mencari cara untuk mengubah 4 buah angka random sehingga mendapatkan hasil akhir sejumlah 24. 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.

II. Algoritma

Program ini menggunakan paradigma Algoritma Brute Force, yaitu algoritma yang menggunakan pendekatan yang lempang untuk memecahkan suatu persoalan. Pada program ini, berikut adalah pendekatan yang dilakukan dalam menyelesaikan masalah

1. Program harus memperoleh 4 buah angka yang akan dihitung melalui input user maupun secara random.
2. 4 angka yang diperoleh tersebut kemudian dicari semua kombinasi posisi angka.
3. Dicari pula semua kombinasi operator, sebanyak 3 operator, untuk perhitungan.
4. Perhitungan waktu *runtime* dimulai dan semua kombinasi angka dan operator tersebut kemudian dihitung berdasarkan 5 kombinasi posisi tanda kurung sehingga diperoleh hasilnya. Contoh : ((a [op] b) [op] c) [op] d)
5. Hasil yang diperoleh tersebut kemudian dibandingkan dengan nilai 24 dan jika sesuai, jumlah solusi bertambah dan solusi tersebut disimpan.
6. Jumlah solusi dan semua solusinya kemudian ditampilkan kepada pengguna beserta waktu *runtime*nya.

Penerapan pendekatan program ini dilakukan menggunakan bahasa C++.

III. Kode Program

Keseluruhan program berada pada *file* main.cpp. Terdapat beberapa fungsi berbeda yang digunakan untuk menerapkan pendekatan-pendekatan di atas.

A. Fungsi *count*

Fungsi ini digunakan untuk mengoperasikan 2 buah nilai dengan operator yang diinginkan. Parameter fungsi tersebut dalah 2 buah nilai float dan sebuah *char* yang merupakan operatornya.

```
float count(float num1, float num2, char op)
{
    if (op == '+')
    {
        return num1 + num2;
    }
    else if (op == '-')
    {
        return num1 - num2;
    }
    else if (op == '*')
    {
        return num1 * num2;
    }
    else if (op == '/')
    {
        return num1 / num2;
    }
    else
    {
        return 1.0;
    }
}
```

B. Fungsi *combination*

Fungsi ini digunakan untuk menghitung keseluruhan operasi dari 4 buah angka tersebut dengan 5 posisi tanda kurung yang berbeda-beda. Parameter *comb* menentukan kombinasi posisi tanda kurung yang mana yang akan digunakan.

```
float combination(float numbers[4], char ops[3], int comb)
{
    if (comb == 1) /*((A B) C) D*/
    {
        return count(count(count(numbers[0], numbers[1], ops[0]), numbers[2],
ops[1]), numbers[3], ops[2]);
    }
    else if (comb == 2) /*(A (B C)) D*/
```

```
    {
        return count(count(numbers[0], count(numbers[1], numbers[2], ops[1]),
ops[0]), numbers[3], ops[2]);
    }
    else if (comb == 3) /*A (B (C D))*/
    {
        return count(numbers[0], count(numbers[1], count(numbers[2],
numbers[3], ops[2]), ops[1]), ops[0]);
    }
    else if (comb == 4) /*A ((B C) D)*/
    {
        return count(numbers[0], count(count(numbers[1], numbers[2], ops[1]),
numbers[3], ops[2]), ops[0]);
    }
    else if (comb == 5) /*(A B) (C D)*/
    {
        return count(count(numbers[0], numbers[1], ops[0]), count(numbers[2],
numbers[3], ops[2]), ops[1]);
    }
}
```

C. Fungsi *Ops*

Fungsi ini digunakan untuk memperoleh semua kombinasi operator untuk memenuhi 3 posisi operator yang diperlukan dalam perhitungan 4 angka tersebut

```
void Ops(char allOps[64][3])
{
    char ops[4] = {'+', '-', '*', '/'};
    int cnt = 0;
    for (int i = 0; i < 4; i++)
    {
        for (int j = 0; j < 4; j++)
        {
            for (int k = 0; k < 4; k++)
            {
                allOps[cnt][0] = ops[i];
                allOps[cnt][1] = ops[j];
                allOps[cnt][2] = ops[k];
                cnt++;
            }
        }
    }
}
```

D. Fungsi *Nums*

Fungsi ini digunakan untuk memperoleh semua kombinasi posisi 4 angka yang ingin dioperasikan.

```
void Ops(char allOps[64][3])
{
    char ops[4] = {'+', '-', '*', '/'};
    int cnt = 0;
    for (int i = 0; i < 4; i++)
    {
        for (int j = 0; j < 4; j++)
        {
            for (int k = 0; k < 4; k++)
            {
                allOps[cnt][0] = ops[i];
                allOps[cnt][1] = ops[j];
                allOps[cnt][2] = ops[k];
                cnt++;
            }
        }
    }
}
```

E. Fungsi *numsManual*

Fungsi ini digunakan untuk mengambil input 4 buah angka secara manual dari user. Fungsi ini menerima input string yang kemudian memanfaatkan *hash* untuk memperoleh nilai float dari input tersebut dan memastikan input sesuai kemudian input yang sudah menjadi float tersebut disimpan dalam sebuah *array*.

```
void numsManual(float numbers[4])
{
    map<string, float> HashInput;
    HashInput["2"] = 2.0;
    HashInput["3"] = 3.0;
    HashInput["4"] = 4.0;
    HashInput["5"] = 5.0;
    HashInput["6"] = 6.0;
    HashInput["7"] = 7.0;
    HashInput["8"] = 8.0;
    HashInput["9"] = 9.0;
    HashInput["10"] = 10.0;
    HashInput["A"] = 1.0;
    HashInput["J"] = 11.0;
    HashInput["Q"] = 12.0;
    HashInput["K"] = 13.0;

    cout << "Masukkan 4 angka atau huruf :\n(A, 2-10, J, Q, K)\n";
    for (int i = 0; i < 4; i++)
    {
        string input;
        cin >> input;
```

```
        while (HashInput.find(input) == HashInput.end())
        {
            cout << "Input tidak sesuai, silahkan input perbaikan sebanyak
input yang tidak sesuai\n";
            cin >> input;
        }
        numbers[i] = HashInput[input];
    }
}
```

F. Fungsi numsRandom

Fungsi ini digunakan untuk memperoleh 4 angka secara *random* untuk dioperasikan. Akan diambil sebuah angka *random* yang kemudian akan dicari nilainya tersebut pada sebuah list yang berisikan pilihan-pilihan angka dan huruf yang dapat dioperasikan kemudian disimpan pada semua *array of float*.

```
void numsRandom(float numbers[])
{
    string cardList[13] = {"A",
                           "2",
                           "3",
                           "4",
                           "5",
                           "6",
                           "7",
                           "8",
                           "9",
                           "10",
                           "J",
                           "Q",
                           "K"};

    cout << "Kartu yang diperoleh :\n";
    srand(time(0));
    for (int i = 0; i < 4; i++)
    {
        int number = rand() % 13;
        cout << cardList[number] << " ";
        numbers[i] = (float)(number + 1);
    }
    cout << "\n\n";
}
```

G. Fungsi mainCounter

Fungsi ini merupakan fungsi utama dari program. Fungsi ini yang akan mengolah 4 angka yang telah diperoleh melalui semua kombinasi poisisi tanda kurung, memastikan hasilnya, dan menyimpannya. Semua hasil yang diperoleh kemudian ditampilkan bersama dengan jumlah solusi dan waktu pengolahan. User kemudian diberi pilihan untuk menyimpan solusi dalam file atau tidak. File yang ingin disimpan kemudian disimpan pada folder *test*.

```
void mainCounter(float numbers[4])
{
    char allOps[64][3];
    float allNums[24][4];

    auto timeStart = std::chrono::high_resolution_clock::now();

    Ops(allOps);
    Nums(allNums, numbers);

    std::stringstream solutions;
    int cntSolution = 0;
    for (int i = 0; i < 24; i++)
    {
        for (int j = 0; j < 64; j++)
        {
            if (abs(combination(allNums[i], allOps[j], 1) - 24) < 0.00001)
/*((A B) C) D*/
            {
                solutions << "((" << allNums[i][0] << allOps[j][0] <<
allNums[i][1] << ")" << allOps[j][1] << allNums[i][2] << ")" << allOps[j][2]
<< allNums[i][3] << endl;
                cntSolution++;
            }
            if (abs(combination(allNums[i], allOps[j], 2) - 24) < 0.00001)
/*(A (B C)) D*/
            {
                solutions << "(" << allNums[i][0] << allOps[j][0] << "(" <<
allNums[i][1] << allOps[j][1] << allNums[i][2] << "))" << allOps[j][2] <<
allNums[i][3] << endl;
                cntSolution++;
            }
            if (abs(combination(allNums[i], allOps[j], 3) - 24) < 0.00001) /*A
(B (C D))*/
            {
                solutions << allNums[i][0] << allOps[j][0] << "(" <<
allNums[i][1] << allOps[j][1] << "(" << allNums[i][2] << allOps[j][2] <<
allNums[i][3] << "))" << endl;
                cntSolution++;
```

```
            }
            if (abs(combination(allNums[i], allOps[j], 4) - 24) < 0.00001) /*A
((B C) D)*/
            {
                solutions << allNums[i][0] << allOps[j][0] << "((" <<
allNums[i][1] << allOps[j][1] << allNums[i][2] << ")" << allOps[j][2] <<
allNums[i][3] << ")" << endl;
                cntSolution++;
            }
            if (abs(combination(allNums[i], allOps[j], 5) - 24) < 0.00001)
/*(A B) (C D)*/
            {
                solutions << "(" << allNums[i][0] << allOps[j][0] <<
allNums[i][1] << ")" << allOps[j][1] << "(" << allNums[i][2] << allOps[j][2]
<< allNums[i][3] << ")" << endl;
                cntSolution++;
            }
        }
    }
    auto timeEnd = std::chrono::high_resolution_clock::now();
    auto runTime =
std::chrono::duration_cast<std::chrono::nanoseconds>(timeEnd - timeStart);

    cout << "Terdapat " << cntSolution << " solusi\n";
    cout << solutions.str() << endl;
    cout << "Runtime :" << runTime.count() * 1e-9 << " Detik\n" << endl;

    cout << "Apakah ingin disimpan dalam file?\n(y/Y untuk ya n/N untuk
tidak)\n";
    char choice;
    cin >> choice;
    if (choice == 'y' || choice == 'Y')
    {
        string path;
        cout << "Masukkan nama file :\n";
        cin >> path;
        ofstream file;
        file.open("./test/" + path + ".txt");
        file << "Terdapat " << cntSolution << " solusi" << endl
             << solutions.str() << endl
             << "Runtime :" << runTime.count() * 1e-9 << endl;
        file.close();
        cout << "Berhasil menuliskan hasil pada file " << path << ".txt\nThank
you and goodbye :)\n\n";
    }
    else if (choice == 'n' || choice == 'N')
    {
        cout << "Ok, bye >:(\n\n";
```

```
    }
}
```

H. Fungsi Main

Fungsi ini yang akan memulai program. Akan ditampilkan pilihan untuk memperoleh 4 angka melalui manual dari user atau secara random atau menghentikan program. Angka yang diperoleh diberikan kepada fungsi mainCounter.

```
int main()
{
    float numbers[4];
    bool running = true;
    while (running)
    {
        int choice;
        cout << "24 Game Solver with Brute Force\nBy Hosea N.A.\n\n";
        cout << "Pilih cara input\n";
        cout << "Ketik 1 untuk input manual\nKetik 2 untuk input random\nKetik
3 untuk keluar dari program\n";
        cin >> choice;
        if (choice == 1)
        {
            numsManual(numbers);
            mainCounter(numbers);
        }
        else if (choice == 2)
        {
            numsRandom(numbers);
            mainCounter(numbers);
        }
        else if (choice == 3)
        {
            running = false;
        }
    }
    return 0;
}
```

IV. Percobaan

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 menumpan solusi dalam file teks	✓	

Berikut adalah beberapa hasil percobaan dari program di atas.

1. Percobaan 1

```
PS D:\Insitut Teknologi Bandung\ITB STEI-G\SMT 4\Stima\Tucil\Tucil1_13521057> ./bin/run
24 Game Solver with Brute Force
By Hosea N.A.

Pilih cara input
Ketik 1 untuk input manual
Ketik 2 untuk input random
Ketik 3 untuk keluar dari program
1
Masukkan 4 angka atau huruf :
(A, 2-10, J, Q, K)
A 2 3 4
Terdapat 242 solusi
((1+2)+3)*4
(1+(2+3))*4
((1*2)*3)*4
(1*(2*3))*4
1*(2*(3*4))
1*((2*3)*4)
(1*2)*(3*4)
((4*3)*2)/1
(4*(3*2))/1
4*(3*(2/1))
4*((3*2)/1)
(4*3)*(2/1)

Runtime :0.0011479 Detik
Apakah ingin disimpan dalam file?
(y/Y untuk ya n/N untuk tidak)
n
Ok, bye >:(
```

2. Percobaan 2

```
PS D:\Insitut Teknologi Bandung\ITB STEI-G\SMT 4\Stima\Tucil\Tucil1_13521057> ./bin/run
24 Game Solver with Brute Force
By Hosea N.A.

Pilih cara input
Ketik 1 untuk input manual
Ketik 2 untuk input random
Ketik 3 untuk keluar dari program
2
Kartu yang diperoleh :
A A 6 8

Terdapat 16 solusi
(6/(1+1))*8
6/((1+1)/8)
(6/(1+1))*8
6/((1+1)/8)
6*(8/(1+1))
(6*8)/(1+1)
6*(8/(1+1))
(6*8)/(1+1)
(8/(1+1))*6
8/((1+1)/6)
(8/(1+1))*6
8/((1+1)/6)
8*(6/(1+1))
(8*6)/(1+1)
8*(6/(1+1))
(8*6)/(1+1)

Runtime :0.0002902 Detik

Apakah ingin disimpan dalam file?
(y/Y untuk ya n/N untuk tidak)
y
Masukkan nama file :
test2
Berhasil menuliskan hasil pada file test2.txt
Thank you and goodbye :)
```

test > test2.txt

```
1   Terdapat 16 solusi
2   (6/(1+1))*8
3   6/((1+1)/8)
4   (6/(1+1))*8
5   6/((1+1)/8)
6   6*(8/(1+1))
7   (6*8)/(1+1)
8   6*(8/(1+1))
9   (6*8)/(1+1)
10  (8/(1+1))*6
11  8/((1+1)/6)
12  (8/(1+1))*6
13  8/((1+1)/6)
14  8*(6/(1+1))
15  (8*6)/(1+1)
16  8*(6/(1+1))
17  (8*6)/(1+1)
18
19  Runtime :0.0002902 Detik
20
```

3. Percobaan 3

```
24 Game Solver with Brute Force
By Hosea N.A.

Pilih cara input
Ketik 1 untuk input manual
Ketik 2 untuk input random
Ketik 3 untuk keluar dari program
1
Masukkan 4 angka atau huruf :
(A, 2-10, J, Q, K)
J Q K K
Terdapat 72 solusi
11+(12+(13/13))
(11+12)+(13/13)
11-((12-13)*13)
11+(12+(13/13))
(11+12)+(13/13)
11-((12-13)*13)
11+((13-12)*13)
```

```
((13/13)+11)+12
(13/13)+(11+12)
13+((13-12)*11)
(13*(13-12))+11
((13/13)+12)+11
(13/13)+(12+11)
(13/(13-12))+11

Runtime :0.0004652 Detik

Apakah ingin disimpan dalam file?
(y/Y untuk ya n/N untuk tidak)
n
Ok, bye >:(
```

4. Percobaan 4

```
24 Game Solver with Brute Force
By Hosea N.A.

Pilih cara input
Ketik 1 untuk input manual
Ketik 2 untuk input random
Ketik 3 untuk keluar dari program
1
Masukkan 4 angka atau huruf :
(A, 2-10, J, Q, K)
7 7 7 7
Terdapat 0 solusi

Runtime :0.0002756 Detik

Apakah ingin disimpan dalam file?
(y/Y untuk ya n/N untuk tidak)
y
Masukkan nama file :
test4
Berhasil menuliskan hasil pada file test4.txt
Thank you and goodbye :)
```

test4.txt

```
1   Terdapat 0 solusi
2
3   Runtime :0.0002756 Detik
4
```

5. Percobaan 5

```
24 Game Solver with Brute Force
By Hosea N.A.

Pilih cara input
Ketik 1 untuk input manual
Ketik 2 untuk input random
Ketik 3 untuk keluar dari program
2
Kartu yang diperoleh :
Q 2 3 3

Terdapat 284 solusi
12+(2*(3+3))
((12*2)+3)-3
12*(2+(3-3))
12*((2+3)-3)
```

```
3/(3/(2*12))
3/((3/2)/12)

Runtime :0.0011065 Detik

Apakah ingin disimpan dalam file?
(y/Y untuk ya n/N untuk tidak)
n
Ok, bye >:(
```

6. Percobaan 6

```
24 Game Solver with Brute Force
By Hosea N.A.

Pilih cara input
Ketik 1 untuk input manual
Ketik 2 untuk input random
Ketik 3 untuk keluar dari program
2
Kartu yang diperoleh :
3 5 3 8

Terdapat 0 solusi

Runtime :0.0002163 Detik

Apakah ingin disimpan dalam file?
(y/Y untuk ya n/N untuk tidak)
y
Masukkan nama file :
test6
Berhasil menuliskan hasil pada file test6.txt
Thank you and goodbye :)
```

test > test6.txt

```
1   Terdapat 0 solusi
2
3   Runtime :0.0002163 Detik
4
```

Link Repository

https://github.com/HoseaNA/Tucil1_13521057

Referensi

1. https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2021-2022/Algoritma-Brute-Force-(2022)-Bag1.pdf
2. https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2021-2022/Algoritma-Brute-Force-(2022)-Bag2.pdf
3. https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2022-2023/Tucil1-Stima-2023.pdf