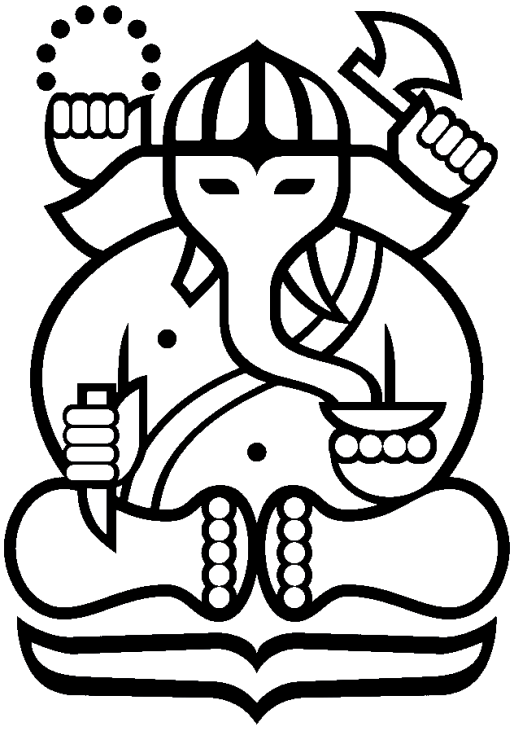
TUGAS KECIL 1 IF2211 STRATEGI ALGORITMA
PENYELESAIAN CYBERPUNK 2077 BREACH PROTOCOL DENGAN ALGORITMA BRUTE FORCE



Disusun oleh:
Enrique Yanuar
13522077

Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung
2024

BAB I
ALGORITMA BRUTE FORCE

Algoritma *brute force* adalah pendekatan yang lempang (straightforward) untuk memecahkan suatu persoalan. Biasanya algoritma *brute force* didasarkan pada pernyataan pada persoalan (problem statement) atau Definisi/konsep yang dilibatkan.. Algoritma *brute force* memecahkan persoalan dengan sangat sederhana, langsung, jelas caranya (*obvious way*), dan dengan model berpikir "just do it".

Cara kerja algoritma brute force pada program yang telah saya buat adalah dengan:

1. Dimulai dari titik koordinat (1, 1), array akan menyimpan koordinat (1,1) untuk pertama kali program akan bergerak pertama secara vertikal dimulai dari baris ke 2 sampai baris ke n.
2. Kemudian pada setiap kemungkinan baris program akan bergerak secara horizontal mengecek semua kolom pada baris tersebut, kecuali kolom yang sudah pernah di lalui (koordinat disimpan dalam array saat itu).
3. Kemudian jika sudah mencapai ujung dari matriks maka rekursif pada bagian tersebut akan dihentikan dan akan mundur atau ketika panjang array sudah mencapai no of buffer maksimum.
4. Pada setiap iterasi rekursif, array pada saat tersebut akan di cek memiliki sub array yang mengandung list sequence atau tidak jika memiliki maka akan dihitung poinnya. Lalu hasil tersebut akan dibandingkan dengan variabel global nilai maks, jika lebih besar maka variabel global nilai maks akan di update dan variabel untuk menyimpan array juga akan di update menjadi array saat itu. Jika tidak lebih besar namun memiliki panjang yang lebih pendek maka akan dilakukan update juga sehingga akan diperoleh hasil akhir yang memiliki nilai reward maksimum namun dengan panjang buffer sependek pendeknya.
5. Begitu juga untuk baris pertama kolom ke 2 sampai n akan dilakukan hal yang sama seperti pada poin 1 sampai 4 sehingga pada akhirnya nanti akan diperoleh hasil akhir yang paling optimal.

BAB II
IMPLEMENTASI ALGORITMA DALAM BAHASA PYTHON

2.1 main.cpp

Fungsi	Deskripsi
`bool areAllUnique()`	Fungsi ini untuk melakukan pengecekan apakah array yang diinputkan melalui cli (token) unik atau tidak. Jika unik maka akan memberi balikan true jika tidak maka false.
`bool isSubArray()`	Fungsi ini untuk melakukan pengecekan apakah array saat ini (temp_result) mengandung variabel list_sequence sebagai subarray nya.
`void solve()`	Fungsi ini merupakan fungsi rekursif untuk melakukan pengecekan semua kemungkinan jalur yang ada kemudian akan memanggil fungsi isSubArray untuk menentukan apakah jalur tersebut merupakan jalur paling optimal atau tidak.
`Int main()`	Fungsi ini menerima inputan user untuk memilih input menggunakan file/cli kemudian menerima inputan lainnya jika user memilih cli. Dan diakhir akan memberi output berupa matriks, list_sequence dan resultnya serta berapa ms waktu eksekusinya. Akhir dari program ini akan menerima input untuk mennetukan apakah result program perlu ditulis dalam file txt atau tidak.

BAB III
SOURCE CODE

3.1 Source Code

```
#include <iostream>
#include <vector>
#include <sstream>
#include <fstream>
#include <chrono>
#include <unordered_set>
using namespace std;
using namespace std::chrono;

int buffer_size, matrix_width, matrix_height, number_of_sequences;
vector<vector<string>> matrix;
vector<vector<string>> list_sequence;
vector<int> list_reward;
vector<pair<int, int>> result_point, temp_result_point;
vector<string> result, temp_result;
int min_buffer;
int max_value, maximax;
int debug;
int len_result;
bool found;

bool areAllUnique(vector<string> &vec)
{
    unordered_set<string> s(vec.begin(), vec.end());

    return s.size() == vec.size();
}
bool isSubArray(vector<string> &arr, vector<string> &subarr)
{
    int arrSize = arr.size();
    int subarrSize = subarr.size();
    if (arrSize < subarrSize)
        return false;

    for (int i = 0; i <= arrSize - subarrSize; ++i)
    {
```

```
        bool match = true;
        for (int j = 0; j < subarrSize; ++j)
        {
            if (arr[i + j] != subarr[j])
            {
                match = false;
                break;
            }
        }
        if (match)
            return true;
    }
    return false;
}

void solve(int x, int y, bool ver, int num)
{
    if (matrix[y][x] == "" || num > buffer_size)
    {
        return;
    }
    int value = 0;

    for (int i = 0; i < number_of_sequences; i++)
    {
        if (isSubArray(temp_result, list_sequence[i]))
        {
            value += list_reward[i];
        }
    }
     if (value > max_value || (value == max_value && found && len_result
> num))
    {
        found = true;
        len_result = num;
        result = temp_result;
        max_value = value;
        result_point = temp_result_point;
    }
```

```
    if (ver)
    {
        // vertical move
        string temp = matrix[y][x];
        matrix[y][x] = "";
        pair<int, int> point(y, x);
        temp_result_point.push_back(point);
        temp_result.push_back(temp);
        for (int i = 1; i < x; i++)
        {
            solve(i, y, !ver, num + 1);
        }
        for (int i = x + 1; i <= matrix_width; i++)
        {
            solve(i, y, !ver, num + 1);
        }
        matrix[y][x] = temp;
        temp_result_point.pop_back();
        temp_result.pop_back();
        // solve(x, y + 1, ver, num);
    }
    else
    {
        // horizontal_move
        string temp = matrix[y][x];
        matrix[y][x] = "";
        pair<int, int> point(y, x);
        temp_result_point.push_back(point);
        temp_result.push_back(temp);
        for (int i = 1; i < y; i++)
        {
            solve(x, i, !ver, num + 1);
        }
        for (int i = y + 1; i <= matrix_height; i++)
        {
            solve(x, i, !ver, num + 1);
        }
        matrix[y][x] = temp;
```

```
        temp_result_point.pop_back();
        temp_result.pop_back();
        // solve(x + 1, y, ver, num);
    }
}

int main()
{
    ios::sync_with_stdio(0);
    maximax = 0;
    len_result = 10000000;
    max_value = 0;
    min_buffer = 1000000;
    found = false;
    string name_file, input_choice;
     cout << "Pilih mau pakai file atau tidak (sebagai input)? y/n (case
sensitive): ";
    cin >> input_choice;
    if (input_choice == "y")
    {
        cout << "input name file: ";
        cin >> name_file;
        // read from file
        ifstream file("./" + name_file + ".txt");
        if (!file.is_open())
        {
            cout << "Error opening file, check the file!!" << endl;
            return 1;
        }
        file >> buffer_size;
        file >> matrix_width >> matrix_height;
        int width = matrix_width + 2;
        vector<string> edge;
        for (int i = 0; i < width; i++)
        {
            edge.push_back("");
        }
        matrix.push_back(edge);
        for (int i = 0; i < matrix_height; i++)
```

```
        {
            vector<string> temp;
            temp.push_back("");
            for (int j = 0; j < matrix_width; j++)
            {
                string a;
                file >> a;
                temp.push_back(a);
            }
            temp.push_back("");
            matrix.push_back(temp);
        }

        matrix.push_back(edge);

        file >> number_of_sequences;

        string sequence_all;
        getline(file, sequence_all);
        for (int i = 0; i < number_of_sequences; ++i)
        {
            vector<string> sequence;
            int reward;
            getline(file, sequence_all);
            stringstream ss(sequence_all);
            string sequence_part;
            int len;
            while (!ss.eof())
            {
                getline(ss, sequence_part, ' ');
                if (sequence_part != "")
                {
                    sequence.push_back(sequence_part);
                    len++;
                }
            }
            if (len < min_buffer)
            {
                min_buffer = len;
```

```
            }
            file >> reward;
            maximax += reward;
            list_sequence.push_back(sequence);
            list_reward.push_back(reward);
            getline(file, sequence_all);
        }
        file.close();
    }
    else
    {
        int token_unik;
        cout << "Masukkan jumlah token unik: ";
        cin >> token_unik;
        vector<string> token;
        bool isNotUnik = true;
        while (isNotUnik)
        {

            cout << "Masukkan token unik: ";
            for (int i = 0; i < token_unik; i++)
            {
                string temp;
                cin >> temp;
                token.push_back(temp);
            }
            if (areAllUnique(token))
            {
                isNotUnik = false;
            }
            else
            {
                cout << "Tidak unik\n";
                token.clear();
            }
        }
        cout << "Masukkan jumlah ukuran buffer: ";
        cin >> buffer_size;
         cout << "Masukkan ukuran matriks (kolom baris): (gunakan spasi)
```

```
";
        cin >> matrix_width >> matrix_height;
        cout << "Masukkan jumlah sekuens: ";
        cin >> number_of_sequences;
        cout << "Masukkan ukuran maksimum sekuens: ";
        int maks_sequnces;
        cin >> maks_sequnces;

        // generate
        int width = matrix_width + 2;
        vector<string> edge;
        for (int i = 0; i < width; i++)
        {
            edge.push_back("");
        }
        matrix.push_back(edge);
        for (int i = 0; i < matrix_height; i++)
        {
            vector<string> temp;
            temp.push_back("");
            for (int j = 0; j < matrix_width; j++)
            {
                string a;
                a = token[(rand() % token_unik)];
                temp.push_back(a);
            }
            temp.push_back("");
            matrix.push_back(temp);
        }
        matrix.push_back(edge);

        for (int i = 0; i < number_of_sequences; i++)
        {
            int len = (rand() % maks_sequnces) + 1;
            if (len < min_buffer)
            {
                min_buffer = len;
            }
            vector<string> sequence;
```

```
            for (int j = 0; j < len; j++)
            {
                sequence.push_back(token[rand() % token_unik]);
            }
            int reward;
            list_sequence.push_back(sequence);

            // range 10 to 50
            reward = (rand() % 41) + 10;
            maximax += reward;
            list_reward.push_back(reward);
        }
    }
    auto start = high_resolution_clock::now();
    for (int i = 1; i <= matrix_width; i++)
    {

        string temp = matrix[1][i];
        matrix[1][i] = "";
        pair<int, int> point(1, i);
        temp_result_point.push_back(point);
        temp_result.push_back(temp);
        for (int j = 2; j <= matrix_height; j++)
            solve(i, j, true, 1);
        matrix[1][i] = temp;
        temp_result_point.pop_back();
        temp_result.pop_back();
    }

    auto stop = high_resolution_clock::now();
    auto duration = duration_cast<microseconds>(stop - start);

    if (found)
    {
        cout << "Result:" << endl
             << endl;
        cout << "Matrix" << endl;
        for (auto &it : matrix)
        {
```

```
            for (auto &it1 : it)
            {
                cout << it1 << " ";
            }
            cout << endl;
        }

        cout << "List sequence and reward" << endl;
        for (int i = 0; i < number_of_sequences; i++)
        {
            int len = list_sequence[i].size();
            for (int j = 0; j < len; j++)
            {
                cout << list_sequence[i][j] << " ";
            }
            cout << endl;
            cout << list_reward[i] << endl
                 << endl;
        }
        cout << "Total Reward" << endl;
        cout << max_value << endl;
        for (auto &it : result)
        {
            // Print the values
            cout << it << " ";
        }
        cout << endl;
        for (auto &it : result_point)
        {
            // Print the values
            cout << it.second << ", " << it.first << endl;
        }
           cout << "Excute in " << duration.count() / 1000 << " ms" <<
endl;
    }
    else
    {
        cout << "Tidak ada hasilnya tu..." << endl;
        cout << "Total Reward \n0" << endl;
```

```
            cout << "Excute in " << duration.count() / 1000 << " ms" <<
endl;
    }

    // saving
    string output_choice, file_name;
    cout << "Mau save? (y/n) case sensitive: ";
    cin >> output_choice;
    if (output_choice == "y")
    {
        cout << "Masukkan nama file (nama saja): ";
        cin >> file_name;
        ofstream outputFile("./../test/" + file_name + ".txt");

        if (outputFile.is_open())
        {

            outputFile << max_value << endl;
            for (auto &it : result)
            {
                // Print the values
                outputFile << it << " ";
            }
            outputFile << endl;
            for (auto &it : result_point)
            {
                // Print the values
                outputFile << it.second << ", " << it.first << endl;
            }
            outputFile << "\n"
                       << duration.count() / 1000 << " ms" << endl;
            outputFile.close();
            cout
                    << "Data was written to " << file_name << ".txt" <<
endl;
        }
        else
        {
            cout << "failed write to file" << endl;
```

```
            }
        }
        return 0;
}
```

BAB IV
HASIL PENGUJIAN

4.1 Test Case 1

Text File:

```
7
6 6
7A 55 E9 E9 1C 55
55 7A 1C 7A E9 55
55 1C 1C 55 E9 BD
BD 1C 7A 1C 55 BD
BD 55 BD 7A 1C 1C
1C 55 55 7A 55 7A
2
7A 55
15
1C BD
30
```

Hasil:

```
Pilih mau pakai file atau tidak (sebagai input)? y/n (case sensitive): y
input name file: input1
Result:

Matrix

 7A 55 E9 E9 1C 55
 55 7A 1C 7A E9 55
 55 1C 1C 55 E9 BD
 BD 1C 7A 1C 55 BD
 BD 55 BD 7A 1C 1C
 1C 55 55 7A 55 7A

List sequence and reward
7A 55
15

1C BD
30

Total Reward
45
7A 55 1C BD
1, 1
1, 2
3, 2
3, 5
Excute in 136 ms
Mau save? (y/n) case sensitive: y
Masukkan nama file (nama saja): output1
Data was written to output1.txt
PS C:\Coding\Tubes\Tucil1_13522077\src>
```

4.2 Test Case 2

Text File:

```
7
6 6
7A 55 E9 E9 1C 55
55 7A 1C 7A E9 55
55 1C 1C 55 E9 BD
BD 1C 7A 1C 55 BD
BD 55 BD 7A 1C 1C
1C 55 55 7A 55 7A
3
7A
1000
BD 7A BD
20
BD 1C BD 1C
30
```

Hasil:

```
Pilih mau pakai file atau tidak (sebagai input)? y/n (case sensitive): y
input name file: input2
Result:

Matrix

 7A 55 E9 E9 1C 55
 55 7A 1C 7A E9 55
 55 1C 1C 55 E9 BD
 BD 1C 7A 1C 55 BD
 BD 55 BD 7A 1C 1C
 1C 55 55 7A 55 7A

List sequence and reward
7A
1000

BD 7A BD
20

BD 1C BD 1C
30

Total Reward
1050
7A BD 7A BD 1C BD 1C
1, 1
1, 4
3, 4
3, 5
6, 5
6, 3
2, 3
Excute in 181 ms
Mau save? (y/n) case sensitive: y
Masukkan nama file (nama saja): output2
Data was written to output2.txt
PS C:\Coding\Tubes\Tucil1_13522077\src>
```

4.3 Test Case 3

Text File:

```
7
6 6
7A 55 E9 E9 1C 55
55 7A 1C 7A E9 55
55 1C 1C 55 E9 BD
BD 1C 7A 1C 55 BD
BD 55 BD 7A 1C 1C
1C 55 55 7A 55 7A
3
1C E9
10
BD 7A BD
20
BD 1C BD 1C
30
```

Hasil:

```
Pilih mau pakai file atau tidak (sebagai input)? y/n (case sensitive): y
input name file: input3
Result:

Matrix

 7A 55 E9 E9 1C 55
 55 7A 1C 7A E9 55
 55 1C 1C 55 E9 BD
 BD 1C 7A 1C 55 BD
 BD 55 BD 7A 1C 1C
 1C 55 55 7A 55 7A

List sequence and reward
1C E9
10

BD 7A BD
20

BD 1C BD 1C
30

Total Reward
50
7A BD 7A BD 1C BD 1C
1, 1
1, 4
3, 4
3, 5
6, 5
6, 3
2, 3
Excute in 154 ms
Mau save? (y/n) case sensitive: y
Masukkan nama file (nama saja): output3
Data was written to output3.txt
PS C:\Coding\Tubes\Tucil1_13522077\src>
```

4.4 Test Case 4

Hasil:

```
Pilih mau pakai file atau tidak (sebagai input)? y/n (case sensitive): n
Masukkan jumlah token unik: 4
Masukkan token unik: AB BC CD DF
Masukkan jumlah ukuran buffer: 7
Masukkan ukuran matriks (kolom baris): (gunakan spasi)  5 5
Masukkan jumlah sekuens: 3
Masukkan ukuran maksimum sekuens: 3
Result:

Matrix

 BC DF CD AB BC
 AB CD CD CD AB
 BC BC BC DF BC
 DF DF CD DF AB
 DF AB CD BC AB

List sequence and reward
BC AB
48

DF CD DF
27

AB
11

Total Reward
59
BC AB
1, 1
1, 2
Excute in 30 ms
Mau save? (y/n) case sensitive: y
Masukkan nama file (nama saja): output4
Data was written to output4.txt
PS C:\Coding\Tubes\Tucil1_13522077\src>
```

4.5 Test Case 5

Hasil:

```
Pilih mau pakai file atau tidak (sebagai input)? y/n (case sensitive): n
Masukkan jumlah token unik: 6
Masukkan token unik: AB BC CD DE FG OK
Masukkan jumlah ukuran buffer: 7
Masukkan ukuran matriks (kolom baris): (gunakan spasi)  8 8
Masukkan jumlah sekuens: 3
Masukkan ukuran maksimum sekuens: 3
Result:

Matrix

 OK OK FG FG OK FG AB AB
 FG CD OK OK BC DE BC OK
 BC CD DE AB DE AB CD DE
 FG FG DE CD CD OK OK AB
 OK AB DE FG OK BC BC AB
 OK DE CD DE DE CD DE BC
 OK FG OK CD FG DE DE BC
 OK DE BC FG FG OK CD AB

List sequence and reward
FG
30

FG FG CD
35

DE FG CD
10

Total Reward
75
FG FG CD DE FG CD
4, 1
4, 8
7, 8
7, 7
2, 7
2, 2
Excute in 2347 ms
Mau save? (y/n) case sensitive: y
Masukkan nama file (nama saja): output5
Data was written to output5.txt
```

4.6 Test Case 6

Hasil:

```
Pilih mau pakai file atau tidak (sebagai input)? y/n (case sensitive): n
Masukkan jumlah token unik: 4
Masukkan token unik: AB JJ KK LO
Masukkan jumlah ukuran buffer: 6
Masukkan ukuran matriks (kolom baris): (gunakan spasi)  5 5
Masukkan jumlah sekuens: 4
Masukkan ukuran maksimum sekuens: 4
Result:

Matrix

 JJ LO KK AB JJ
 AB KK KK KK AB
 JJ JJ JJ LO JJ
 LO LO KK LO AB
 LO AB KK JJ AB

List sequence and reward
JJ AB KK
20

KK LO KK JJ
37

LO LO KK LO
41

JJ JJ AB
22

Total Reward
57
JJ AB KK LO KK JJ
1, 1
1, 2
2, 2
2, 1
3, 1
3, 3
Excute in 9 ms
Mau save? (y/n) case sensitive: y
Masukkan nama file (nama saja): output6
Data was written to output6.txt
```

BAB V
LAMPIRAN

5.1 Repository

https://github.com/mybajwk/Tucil1_13522077

5.2 Tabel Kelengkapan

Poin	Ya	Tidak
1. Program berhasil dikompilasi tanpa kesalahan	☑	☐
2. Program berhasil dijalankan	☑	☐
3. Program dapat membaca masukan berkas .txt	☑	☐
4. Program dapat menghasilkan masukan secara acak	☑	☐
5. Solusi yang diberikan program optimal	☑	☐
6. Program dapat menyimpan solusi dalam berkas .txt	☑	☐
7. Program memiliki GUI	☐	☑