LAPORAN TUGAS KECIL I

IF2211 STRATEGI ALGORITMA

Penyelesaian Cyberpunk 2077 Breach Protocol dengan Algoritma Brute Force

Disusun oleh:

Berto Richardo Togatorop

13522098

Program Studi Teknik Informatika

Sekolah Teknik Elektro dan Informatika

Institut Teknologi Bandung

2024

Daftar Isi

BAGIAN I ALGORITMA BRUTE FORCE **3**
BAB II SOURCE PROGRAM **4**
BAGIAN III SCREENSHOT HASIL TEST **14**
LINK REPOSITORY **19**
CHECKLIST **19**

BAGIAN I
ALGORITMA BRUTE FORCE

Algoritma Brute Force adalah metode pemecahan masalah yang sederhana dan langsung. Dalam konteks algoritma ini, pendekatan yang digunakan adalah dengan mencoba semua kemungkinan secara sistematis untuk mencari solusi. Dengan pendekatan ini, algoritma Brute Force tidak memerlukan strategi yang rumit. Sebaliknya, algoritma ini secara langsung mengevaluasi setiap kemungkinan solusi secara terpisah. Dalam penyelesaian Cyberpunk 2077 Breach Protocol dengan algoritma Brute Force, langkah-langkah yang digunakan adalah

1. Ambil token dari baris teratas dan tambahkan ke dalam buffer. Kemudian, untuk setiap token pada kolom yang sama, iterasikan untuk memilih token berikutnya secara vertikal.

2. Jika buffer belum penuh, lakukan iterasi untuk setiap token yang belum diambil secara horizontal jika langkah sebelumnya dilakukan secara vertikal, atau secara vertikal jika langkah sebelumnya dilakukan secara horizontal. Pastikan bahwa sel yang sudah dilewati tidak ditempati lagi.

3. Setelah itu, periksa jika buffer mengandung sekuens dengan bobot reward positif. Jika ya, dan ini adalah penemuan pertama, inisialisasi solusi dengan buffer tersebut. Jika ditemukan buffer lain dengan bobot reward yang lebih besar, ganti solusi dengan buffer yang baru. Solusi juga diganti dengan buffer yang baru jika ditemukan buffer yang memiliki nilai yang sama dengan buffer maksimal saat ini, tetapi memiliki banyak langkah yang lebih sedikit.

3. Ulangi langkah 2 sampai buffer penuh. Setelah penuh, maka akan dilakukan pemeriksaan dengan kombinasi buffer yang lain hingga setiap kombinasi yang legal(tidak ada sel yang dikunjungi 2 kali) dicoba.

4. Jika point maksimal saat ini tidak 0, hal itu berarti ada solusi yang memungkinkan. Namun point maksimal adalah 0, berarti tidak ada solusi

BAB II
SOURCE PROGRAM

Proyek ini diimplementasikan menggunakan Bahasa C++ dengan compiler g++ versi 11. Fungsi atau prosedur yang diimplementasikan dalam program ini adalah sebagai berikut.

1. inputFromFile
2. getRandomNumber,
3. generateRandomSequencesAndRewards,
4. generateRandomMatrix,
5. inputRandom,
6. getPoint,
7. findToken,
8. printResult,
9. saveToFile,
10. main,

Berikut ini adalah source code program ini.

```
#include <iostream>
#include <vector>
#include <string>
#include <chrono>
#include <fstream>
#include <sstream>
#include <random>
#include <algorithm>

using namespace std;
using namespace std::chrono;

vector<string> sequences;
vector<int> rewards;
int maxPoint = 0;
vector<pair<int, int>> optCoord;
string resToken;

// INPUT
void inputFromFile(int &buffer_size, int &nRow, int &nCol, int &sequences_length,
vector<vector<string>> &matrix)
{
    string dir = "../test/input/";
    string name;
    cout << "Masukkan nama file: ";
```

```
    cin >> name;
    dir += name;
    cin.ignore(); // Ignore newline character

    ifstream file(dir);
    if (!file.is_open())
    {
        cerr << "Unable to open file: " << dir << endl;
        return;
    }

    // Read buffer_size, nCol, nRow from file
    file >> buffer_size;
    file >> nCol >> nRow;
    matrix.resize(nRow, vector<string>(nCol));

    // Read matrix data from file
    for (int i = 0; i < nRow; ++i)
    {
        for (int j = 0; j < nCol; ++j)
        {
            file >> matrix[i][j];
        }
    }

    // Read sequences and rewards from file
    file >> sequences_length;
    file.ignore(); // Ignore the newline character after reading sequences_length
    sequences.resize(sequences_length);
    rewards.resize(sequences_length);
    for (int i = 0; i < sequences_length; ++i)
    {
        getline(file >> std::ws, sequences[i]); // Read sequence line
        file >> rewards[i];                     // Read reward
        file.ignore();                          // Ignore the newline character after
reading reward
    }

    file.close(); // Close the file
}

// random input
// Function to generate a random integer within a range
int getRandomNumber(int min, int max)
{
```

```
    static random_device rd;
    static mt19937 eng(rd());
    uniform_int_distribution<> distribution(min, max);
    return distribution(eng);
}

// Function to generate random sequences and rewards
void generateRandomSequencesAndRewards(int numberOfSequences, int maxSequenceLength,
                                       int tokenNumber, const vector<string> &tokens)
{
    sequences.clear();
    rewards.clear();

    for (int i = 0; i < numberOfSequences; ++i)
    {
        int rdLen = getRandomNumber(2, maxSequenceLength);
        string temp;

        for (int j = 0; j < rdLen; ++j)
        {
            int rdIndex = getRandomNumber(0, tokenNumber - 1);
            temp += tokens[rdIndex];
            if (j != rdLen - 1)
            {
                temp += " ";
            }
        }

        if (find(sequences.begin(), sequences.end(), temp) == sequences.end())
        {
            int reward = getRandomNumber(0, 1000);
            rewards.push_back(reward);
            sequences.push_back(temp);
        }
    }
}

// Function to generate a random matrix
vector<vector<string>> generateRandomMatrix(int row, int col, int tokenNumber, const
vector<string> &uniqueToken)
{
    vector<vector<string>> matrix(row, vector<string>(col));

    for (int i = 0; i < row; ++i)
    {
```

```
        for (int j = 0; j < col; ++j)
        {
            int rdIndex = getRandomNumber(0, tokenNumber - 1);
            matrix[i][j] = uniqueToken[rdIndex];
        }
    }

    return matrix;
}

// Function to get input from user that used to automate the variables
void inputRandom(int &buffer_size, int &nRow, int &nCol, int &sequences_length,
vector<vector<string>> &matrix)
{
    // Token
    int nTokens;
    cin >> nTokens;

    string rawToken;
    getline(cin >> std::ws, rawToken);

    vector<string> uniqueToken;
    for (int i = 0; i < nTokens; i++)
    {
        uniqueToken.push_back(rawToken.substr(3 * i, 2)); // Extract a substring of
length 2 starting from the index 3 * i + 1
    }

    // vector<string> uniqueToken = {"AA", "BB", "CC", "E9"};

    // Input buffer size
    cin >> buffer_size;

    // Matrix dimensions
    cin >> nCol >> nRow;

    // Sequences
    int nSeq;
    cin >> nSeq;

    int lengthSeq;
    cin >> lengthSeq;

    generateRandomSequencesAndRewards(nSeq, lengthSeq, nTokens, uniqueToken);
    sequences_length = sequences.size();
```

```
    // Assume rewards.size() == sequences.size()

    matrix = generateRandomMatrix(nRow, nCol, nTokens, uniqueToken);
}

// UTILS
// getPoint
int getPoint(string token)
{
    int point = 0;
    for (size_t i = 0; i < sequences.size(); i++)
    {
        if (token.find(sequences[i]) != string::npos)
        {
            point += rewards[i];
        }
    }
    return point;
}

// Recursive function to find all Token
void findToken(vector<vector<string>> &matrix, int x, int y, string token, int
buffer_size, vector<vector<bool>> &visited, int steps, vector<pair<int, int>> coord)
{

    // Mark current position as visited and update the coord
    coord.push_back({x, y});

    // check current point
    int currPoint = getPoint(token);
    if (coord.size() < optCoord.size() && currPoint == maxPoint)
    {
        optCoord = coord; // Append new pair (x, y)
        resToken = token;
    }
    else if (currPoint > maxPoint)
    {
        maxPoint = currPoint;
        optCoord = coord; // Append new pair (x, y)
        resToken = token;
    }

    // make sure the steps is still < buffer_size
    if (steps < buffer_size)
    {
```

```
        // Mark visited
        visited[x][y] = true;

        if (steps % 2 == 0)
        {
            // move vertically

            for (int i = 0; i < matrix.size(); i++)
            {
                if (!visited[i][y])
                {
                    findToken(matrix, i, y, token + " " + matrix[i][y], buffer_size,
visited, steps + 1, coord);
                }
            }
        }

        else
        {
            // move horizontally
            for (int j = 0; j < matrix[0].size(); j++)
            {
                {
                    if (!visited[x][j])
                    {
                        findToken(matrix, x, j, token + " " + matrix[x][j],
buffer_size, visited, steps + 1, coord);
                    }
                }
            }
        }

        // Mark current position as unvisited when backtracking
        visited[x][y] = false;

        coord.push_back(make_pair(x, y));
    }
}

// OUTPUT
void printResult(int time, char menu, int nRow, int nCol, vector<vector<string>>
matrix)
{
    if (maxPoint == 0)
    {
```

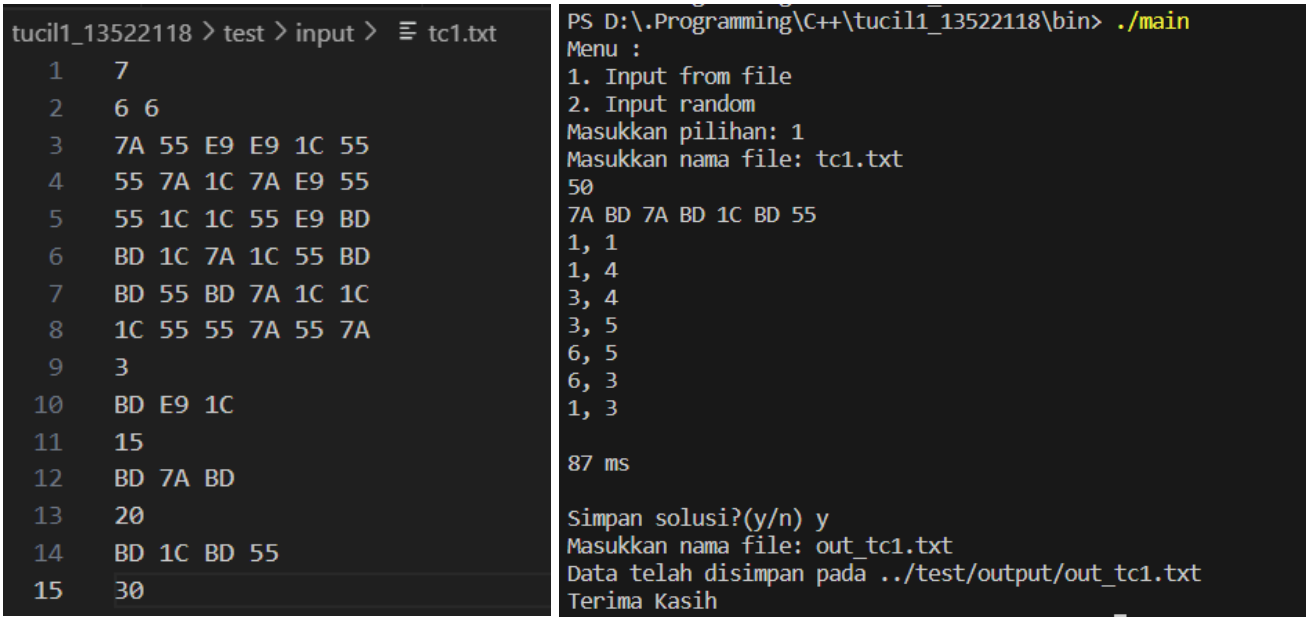
```
        cout << "No solution" << endl;
    }
    else
    {
        if (menu == '2')
        {
            cout << "Matrix" << endl;
            for (int i = 0; i < nRow; ++i)
            {
                for (int j = 0; j < nCol; ++j)
                {
                    cout << matrix[i][j] << " ";
                }
                cout << endl;
            }

            cout << endl
                 << "Sequences" << endl;

            for (int i = 0; i < sequences.size(); ++i)
            {
                cout << "Sequence: " << sequences[i] << ", Reward: " << rewards[i] <<
endl;
            }
        }

        // Print max point, token, coordinats
        cout << maxPoint << endl;
        cout << resToken << endl;
        for (auto resCoord : optCoord)
        {
            cout << resCoord.second + 1 << ", " << resCoord.first + 1 << endl;
        }
        cout << endl;

        // Output the duration
        cout << time << " ms" << endl;
    }
}

// save to file
void saveToFile(int time)
{
    string dir = "../test/output/";
    string name;
```

```
    cout << "Masukkan nama file: ";
    cin >> name;
    dir += name;

    string maxPointStr = to_string(maxPoint);
    string resTokenStr = resToken;
    string timeStr = to_string(time);
    ostringstream oss;
    for (const auto &coord : optCoord)
    {
        oss << coord.second + 1 << ", " << coord.first + 1 << endl;
    }
    string resCoordStr = oss.str();

    ofstream outputFile(dir);
    if (outputFile.is_open())
    {
        if (maxPoint != 0)
        {
            outputFile << maxPointStr << endl;
            outputFile << resTokenStr << endl;
            outputFile << resCoordStr << endl;
            outputFile << timeStr << " ms" << endl;
            outputFile.close();
            cout << "Data telah disimpan pada " + dir << endl;
        }
        else
        {
            outputFile << "No Solution" << endl;
        }
    }
    else
    {
        cout << "Tidak dapat membuat file" << endl;
    }
}

int main()
{
    // Define Variables
    int buffer_size;
    int nRow;
    int nCol;
    int sequences_length;
    vector<vector<string>> matrix;
```

```
    // Initialize token
    string token;

    // Seed the random number generator
    srand(time(nullptr));

    // Get Input
    cout << "Menu :" << endl;
    cout << "1. Input from file" << endl;
    cout << "2. Input random " << endl;
    cout << "Masukkan pilihan: ";
    char menu;
    cin >> menu;

    if (menu == '1')
    {
        inputFromFile(buffer_size, nRow, nCol, sequences_length, matrix);
    }
    else if (menu == '2')
    {
        inputRandom(buffer_size, nRow, nCol, sequences_length, matrix);
    }
    else
    {
        cout << "Input tidak valid." << endl;
        return 0;
    }

    // Initialize the visited array
    vector<vector<bool>> visited(nRow, vector<bool>(nCol, false));

    // Get the current time before executing the code
    auto start = high_resolution_clock::now();

    // findToken starting from the first row
    for (int j = 0; j < matrix[0].size(); j++)
    {
        findToken(matrix, 0, j, matrix[0][j], buffer_size, visited, 0, {});
    }

    // Get the current time after executing the code
    auto stop = high_resolution_clock::now();

    // Calculate the duration by subtracting start time from stop time
```

```
    auto duration = duration_cast<milliseconds>(stop - start);

    int time = duration.count();

    printResult(time, menu, nRow, nCol, matrix);
    cout << endl
         << "Simpan solusi?(y/n) ";

    char save;
    cin >> save;
    if (save == 'y' || save == 'Y')
    {
        saveToFile(time);
    }

    cout << "Terima Kasih" << endl;

    return 0;


}
```

BAGIAN III
SCREENSHOT HASIL TEST

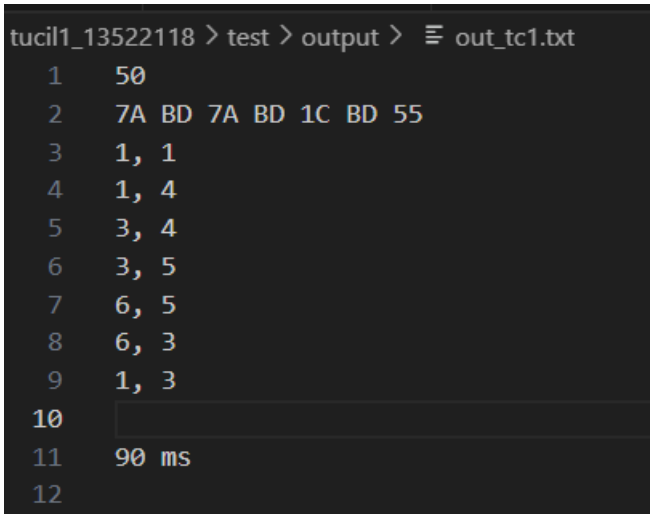


tucil1_13522118 > test > input > tc1.txt

```
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
BD E9 1C
15
BD 7A BD
20
BD 1C BD 55
30
```

```
PS D:\.Programming\C++\tucil1_13522118\bin> ./main
Menu :
1. Input from file
2. Input random
Masukkan pilihan: 1
Masukkan nama file: tc1.txt
50
7A BD 7A BD 1C BD 55
1, 1
1, 4
3, 4
3, 5
6, 5
6, 3
1, 3

87 ms

Simpan solusi?(y/n) y
Masukkan nama file: out_tc1.txt
Data telah disimpan pada ../test/output/out_tc1.txt
Terima Kasih
```

Gambar 1. Input dan output dengan file tc1.txt



tucil1_13522118 > test > output > out_tc1.txt

```
50
7A BD 7A BD 1C BD 55
1, 1
1, 4
3, 4
3, 5
6, 5
6, 3
1, 3

90 ms

```

Gambar 1.b. File output hasil penyelesaian tc2.txt

tucil1_13522118 > test > input > tc2.txt

```
7
6 6
55 1C 55 BD BD 1C
55 1C E9 7A BD 1C
55 55 55 1C 55 7A
7A 1C 55 E9 7A 1C
BD 55 55 7A 7A BD
55 55 E9 E9 55 55
3
BD 1C E9 55
638
E9 BD
47
E9 55 7A 1C
796
```

```
PS D:\.Programming\C++\tucil1_13522118\bin> ./main
Menu :
1. Input from file
2. Input random
Masukkan pilihan: 1
Masukkan nama file: tc2.txt
796
55 55 E9 55 7A 1C 1C
1, 1
1, 2
3, 2
3, 3
6, 3
6, 1
2, 1

90 ms

Simpan solusi?(y/n) y
Masukkan nama file: out_tc2.txt
Data telah disimpan pada ../test/output/out_tc2.txt
Terima Kasih
```

Gambar 2.a. Input dan output dengan file tc2.txt

tucil1_13522118 > test > output > out_tc2.txt

```
796
55 55 E9 55 7A 1C 1C
1, 1
1, 2
3, 2
3, 3
6, 3
6, 1
2, 1

87 ms

```

Gambar 2.b. File output hasil penyelesaian tc2.txt

tucil1_13522118 > test > input > tc3.txt

```
8
8 8
DD DD BB BB EE BB CC EE
DD BB CC DD DD DD AA DD
AA CC AA EE DD EE EE EE
AA EE BB EE CC DD BB EE
BB BB CC AA CC EE BB BB
AA CC DD BB EE AA EE AA
CC DD EE EE BB BB EE AA
DD EE CC CC AA EE AA CC
4
AA AA BB EE
924
BB CC
476
CC AA
894
AA BB BB DD
-276
```

```
PS D:\.Programming\C++\tucil1_13522118\bin> ./main
Menu :
1. Input from file
2. Input random
Masukkan pilihan: 1
Masukkan nama file: tc3.txt
2294
DD DD BB CC AA AA BB EE
1, 1
1, 2
2, 2
2, 3
1, 3
1, 4
3, 4
3, 7

6817 ms

Simpan solusi?(y/n) y
Masukkan nama file: out_tc3.txt
Data telah disimpan pada ../test/output/out_tc3.txt
Terima Kasih
```

Gambar 3a. Input dan output tc3.txt

tucil1_13522118 > test > output > out_tc3.txt

```
2294
DD DD BB CC AA AA BB EE
1, 1
1, 2
2, 2
2, 3
1, 3
1, 4
3, 4
3, 7

6817 ms
```

Gambar 3.b. File output hasil penyelesaian tc2.txt

```
PS D:\.Programming\C++\tucil1_13522118\bin> ./main
Menu :
1. Input from file
2. Input random
Masukkan pilihan: 2
5
BD 1C 7A 55 E9
7
6 6
3
4
Matrix
1C 55 7A 55 E9 1C
BD 55 1C BD E9 7A
7A 1C 1C 1C 55 7A
1C 1C E9 7A 1C 1C
E9 55 BD E9 55 1C
BD 55 1C 55 BD 55

Sequences
Sequence: BD E9, Reward: 297
Sequence: 1C E9, Reward: 200
Sequence: E9 BD 7A 7A, Reward: 340
637
1C BD BD E9 BD 7A 7A
1, 1
1, 6
5, 6
5, 2
1, 2
1, 3
6, 3

91 ms

Simpan solusi?(y/n) y
Masukkan nama file: out_rand1.txt
Data telah disimpan pada ../test/output/out_rand1.txt
Terima Kasih
PS D:\.Programming\C++\tucil1_13522118\bin>
```

tucil1_13522118 > test > output > out_rand1.txt

```
1   637
2   1C BD BD E9 BD 7A 7A
3   1, 1
4   1, 6
5   5, 6
6   5, 2
7   1, 2
8   1, 3
9   6, 3
10
11  91 ms
12
```

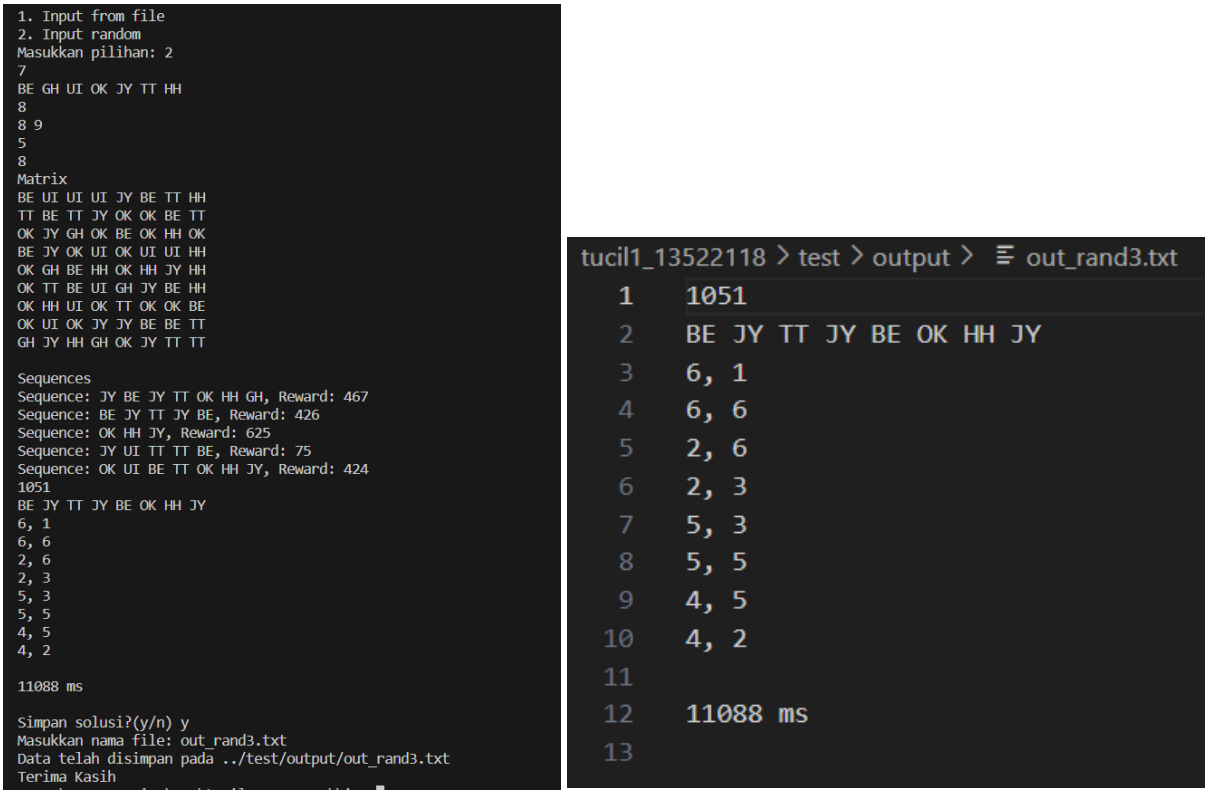
Gambar 4. Input dan output test *generate* random pertama

```
PS D:\.Programming\C++\tucil1_13522118\bin> ./main
Menu :
1. Input from file
2. Input random
Masukkan pilihan: 2
4
BD EF GG 99
6
7 6
3
4
Matrix
99 BD 99 99 BD BD EF
EF GG GG BD BD EF GG
99 EF BD EF BD GG EF
BD 99 99 BD EF GG GG
BD GG BD 99 99 GG GG
BD BD EF EF EF GG BD

Sequences
Sequence: BD EF BD GG, Reward: 247
Sequence: 99 99 EF BD, Reward: 145
Sequence: GG EF, Reward: 506
753
99 BD EF BD GG EF
1, 1
1, 4
5, 4
5, 2
2, 2
2, 3

29 ms

Simpan solusi?(y/n) y
Masukkan nama file: out_rand2.txt
Data telah disimpan pada ../test/output/out_rand2.txt
Terima Kasih
```

tucil1_13522118 > test > output > out_rand2.txt

```
1   753
2   99 BD EF BD GG EF
3   1, 1
4   1, 4
5   5, 4
6   5, 2
7   2, 2
8   2, 3
9
10  29 ms
11
```

Gambar 5. Input dan output test *generate* random kedua.



Gambar 6. Input dan output test *generate* random ketiga.

LINK REPOSITORY

https://github.com/BertoRichardo/Tucil1_13522118

CHECKLIST

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		✓