LAPORAN TUGAS KECIL 1

IF2211 STRATEGI ALGORITMA

Penyelesaian Permainan Cyberpunk 2077 Breach Protocol dengan Algoritma Brute Force

Disusun oleh:

Muhammad Neo Cicero Koda (13522108)

Program Studi Teknik Informatika

Sekolah Teknik Elektro dan Informatika

Institut Teknologi Bandung

2024

DAFTAR ISI

BAB I 3
ALGORITMA BRUTE FORCE 3
BAB II 4
SOURCE CODE PROGRAM 4
BAB III 13
SCREENSHOT HASIL TEST 13
1. **Test Case I** 13
2. **Test Case II** 14
3. **Test Case III** 15
4. **Test Case IV** 16
5. **Test Case V** 18
6. **Test Case VI** 18

LINK KE REPOSITORI 20
LAMPIRAN 20

BAB I

ALGORITMA BRUTE FORCE

Algoritma Brute Force adalah algoritma yang menyelesaikan suatu permasalahan dengan pendekatan yang *straightforward*. Cara kerja algoritma ini adalah menelusuri dan mencoba semua jawaban yang mungkin dalam suatu permasalahan sampai mendapatkan hasil yang terbaik/optimal. Karena metode penyelesaiannya yang mencoba semua kemungkinan, algoritma ini tergolong lambat dan memerlukan waktu komputasi yang banyak.

Algoritma Brute Force dapat diaplikasikan ke permainan Cyberpunk 2077 Breach Protocol dengan pendekatan rekursif untuk mencari solusi yang paling optimal. Fungsi rekursif tersebut akan menerima parameter berupa koordinat saat ini, isi *buffer* saat ini, dan arah gerak. Fungsi rekursif untuk menelusuri semua gerakan yang mungkin akan memiliki beberapa kasus.

Kasus pertama adalah ketika isi *buffer* yang diterima fungsi sudah penuh. Jika *buffer* sudah penuh, fungsi hanya akan menghitung *reward* yang didapatkan dari *buffer* dan mengganti informasi *reward* dan *buffer* maksimal jika mendapat hasil yang lebih optimal.

Kasus kedua adalah ketika fungsi dipanggil untuk pertama kalinya dengan koordinat awal dianggap berada di (0, 0). Pada kasus ini, fungsi akan menelusuri baris pertama matriks. Untuk setiap elemen baris pertama, fungsi akan menambahkan isi *buffer* dengan koordinat yang sedang ditelusuri. Kemudian, fungsi akan menghitung *reward* untuk *buffer* tersebut dan memperbarui nilai *reward* dan *buffer* maksimal jika menemui hasil yang lebih optimal. Lalu, fungsi akan memanggil dirinya sendiri dengan parameter berupa koordinat saat ini dan *buffer* yang sudah diperbarukan. Setelah dipanggil, elemen terakhir *buffer* akan dikeluarkan dan *buffer* akan kembali ke keadaan semula.

Kasus ketiga adalah ketika isi *buffer* belum penuh. Mekanisme penelusuran, penghitungan *reward*, dan pemanggilan diri sendiri yang dilakukan hampir sama dengan kasus kedua. Jika arah gerak adalah vertikal, koordinat yang ditelusuri adalah semua koordinat yang belum terdapat pada *buffer* dan berada pada satu kolom. Jika arah gerak adalah horizontal, koordinat yang ditelusuri adalah semua koordinat yang belum terdapat pada *buffer* dan berada pada satu baris.

Setelah menelusuri semua kemungkinan, program akan menampilkan hasil berupa *reward* maksimal yang bisa didapatkan serta isi *buffer* beserta koordinat yang ditelusurinya.

BAB II

SOURCE CODE PROGRAM

Program ini dibuat dalam bahasa C++ dengan menggunakan library:

1. iostream
2. cstdlib
3. vector
4. filesystem
5. fstream
6. sstream
7. string
8. algorithm
9. chrono
10. cctype

Berikut adalah *source code* program:

```cpp
// Nama      : Muhammad Neo Cicero Koda
// Kelas/NIM : K02/13522108

// Header files
#include <iostream>
#include <cstdlib>
#include <vector>
#include <filesystem>
#include <fstream>
#include <sstream>
#include <string>
#include <algorithm>
#include <chrono>

// Namespace
namespace fs = std::filesystem;

// Custom classes
class Sequence {
    public:
        std::vector<std::string> seqArray;
        int reward;

    Sequence(const std::vector<std::string>& seqArr, int rwd) : seqArray(seqArr), reward(rwd) {}
};

class Coordinate {
    public:
        int col;
        int row;
        std::string token;
        Coordinate(int c, int r, std::string t) : col(c), row(r), token(t) {}
};
```

```cpp
// Variables used
int numTokens;
std::vector<std::string> tokens;

int bufferSize;
std::vector<Coordinate> maxBuffer;
int maxReward;

int matWidth;
int matHeight;
std::vector<std::vector<std::string>> matrix;

int numSeq;
std::vector<Sequence> sequences;
int maxSeqSize;

int timeTaken; // save time taken to get solution
```

```cpp
// Helper functions
Sequence generateSequence() {
    int size = rand() % (maxSeqSize - 1) + 2;
    std::vector<std::string> seqArray;
    for (int i = 0; i < size; i++) {
        seqArray.push_back(tokens[rand() % tokens.size()]);
    }
    int reward = rand() % (101) -50;

    Sequence sequence(seqArray, reward);

    return sequence;
};

bool isUniqueSequence(Sequence seq) {
    for (const auto& sequence : sequences) {
        if (std::equal(sequence.seqArray.begin(), sequence.seqArray.end(), seq.seqArray.begin(), seq.seqArray.end())) {
            return false;
        }
    }
    return true;
}

bool coordinateTravelled(const std::vector<Coordinate>& vec, int targetCol, int targetRow) {
    for (const Coordinate& coord : vec) {
        if (coord.col == targetCol && coord.row == targetRow) {
            return true;
        }
    }
    return false;
}
```

```cpp
bool hasSequence(Sequence seq, std::vector<Coordinate> buffer) {
    if (seq.seqArray.size() > buffer.size()) {
        return false;
    }

    bool has;
    for (int i = 0; i < (buffer.size() - seq.seqArray.size() + 1); i++) {
        has = true;
        for (int j = 0; j < seq.seqArray.size(); j++) {
            if (seq.seqArray[j] != buffer[i + j].token) {
                has = false;
            }
        }
        if (has) {
            return true;
        }
    }

    return has;
}

void countReward(std::vector<Coordinate> buffer) {
    int bufferReward = 0;
    for (Sequence seq : sequences) {
        if (hasSequence(seq, buffer)) {
            bufferReward += seq.reward;
        }
    }
    if (bufferReward > maxReward) {
        maxBuffer = buffer;
        maxReward = bufferReward;
    }
}
```

```cpp
// Brute force algorithm (recursive)
void findPossibleMoves(Coordinate currentCoord, std::vector<Coordinate> currentBuffer, bool vertical) {
    if (currentBuffer.size() == bufferSize) { // recursion basis
        countReward(currentBuffer);
    } else if (currentCoord.col == 0 && currentCoord.row == 0) { // first move
        for (int i = 0; i < matrix[0].size(); i++) {
            Coordinate coord(i + 1, 1, matrix[0][i]);
            currentBuffer.push_back(coord);
            countReward(currentBuffer);
            findPossibleMoves(coord, currentBuffer, true);
            currentBuffer.pop_back();
        }
    } else {
        if (vertical) { // vertical move
            for (int i = 0; i < matrix.size(); i++) {
                if (!coordinateTravelled(currentBuffer, currentCoord.col, i + 1)) {
                    Coordinate coord(currentCoord.col, i + 1, matrix[i][currentCoord.col - 1]);
                    currentBuffer.push_back(coord);
                    countReward(currentBuffer);
                    findPossibleMoves(coord, currentBuffer, false);
                    currentBuffer.pop_back();
                }
            }
        } else { // horizontal move
            for (int i = 0; i < matrix[0].size(); i++) {
                if (!coordinateTravelled(currentBuffer, i + 1, currentCoord.row)) {
                    Coordinate coord(i + 1, currentCoord.row, matrix[currentCoord.row - 1][i]);
                    currentBuffer.push_back(coord);
                    countReward(currentBuffer);
                    findPossibleMoves(coord, currentBuffer, true);
                    currentBuffer.pop_back();
                }
            }
        }
    }
}
```

```cpp
// I/O Functions
void readFromFile() {
    std::string path;
    std::cout << "Mohon masukkan path file: ";
    std::cin >> path;

    while (!(fs::exists(path) && (path.length() >= 4) && (path.substr(path.length() - 4) == ".txt"))) {
        std::cout << "Path yang diberikan tidak ditemukan atau bukan file txt. Mohon masukkan ulang path: ";
        std::cin >> path;
    }

    std::ifstream inputFile(path);

    inputFile >> bufferSize;
    if (inputFile.fail() || bufferSize < 0) {
        throw std::invalid_argument("Ukuran buffer tidak valid. Program akan berhenti.");
    }

    inputFile >> matWidth >> matHeight;
    if (inputFile.fail() || matWidth <= 0 || matHeight <= 0) {
        throw std::invalid_argument("Dimensi matriks tidak valid. Program akan berhenti.");
    }

    matrix.resize(matHeight, std::vector<std::string>(matWidth));
    for (int i = 0; i < matHeight; i++) {
        for (int j = 0; j < matWidth; j++) {
            inputFile >> matrix[i][j];
            if (!(matrix[i][j].size() == 2 && std::isalnum(static_cast<unsigned char>(matrix[i][j][0])) && std::isalnum(static_cast<unsigned char>(matrix[i][j][1])))) {
                throw std::invalid_argument("Token pada matriks tidak valid. Program akan berhenti.");
            }
            if (inputFile.eof() || inputFile.fail()) {
                throw std::invalid_argument("Masukan matriks tidak valid. Program akan berhenti.");
            }
        }
    }

    inputFile >> numSeq;
    if (inputFile.fail() || numSeq < 0) {
        throw std::invalid_argument("Jumlah sekuens tidak valid. Program akan berhenti.");
    }
    inputFile.ignore();

    std::string seqLine;
    int reward;
    for (int i = 0; i < numSeq; i++) {
        std::getline(inputFile, seqLine);
        if (inputFile.eof() || inputFile.fail()) {
                throw std::invalid_argument("Masukan sekuens tidak valid. Program akan berhenti.");
        }

        std::string token;
        std::vector<std::string> seqArray;
        std::istringstream iss(seqLine);
        while (iss >> token) {
            if (!(token.size() == 2 && std::isalnum(static_cast<unsigned char>(token[0])) && std::isalnum(static_cast<unsigned char>(token[1])))) {
                throw std::invalid_argument("Token pada sekuens tidak valid. Program akan berhenti.");
            }
            seqArray.push_back(token);
        }

        inputFile >> reward;
        inputFile.ignore();
        Sequence sequence(seqArray, reward);
        sequences.push_back(sequence);
    }
}
```

```cpp
void readFromInput() {
    std::cout << std::endl << "Masukkan informasi tentang matriks dan sekuens yang digunakan: " << std::endl;

    std::cin >> numTokens;
    if (std::cin.fail() || numTokens < 0) {
        throw std::invalid_argument("Jumlah token tidak valid. Program akan berhenti. ");
    }

    tokens.resize(numTokens);
    std::string token;
    for (int i = 0; i < numTokens; i++) {
        std::cin >> token;
        if (!(token.size() == 2 && std::isalnum(static_cast<unsigned char>(token[0])) && std::isalnum(static_cast<unsigned char>(token[1])))) {
            throw std::invalid_argument("Token yang diberikan tidak berukuran dua/tidak alfanumerik. Program akan berhenti. ");
        } else {
            tokens[i] = token;
        }
    }

    for (int i = 0; i < tokens.size() - 1; i++) {
        for (int j = i + 1; j < tokens.size(); j++) {
            if (tokens[i] == tokens[j]) {
                throw std::invalid_argument("Token yang diberikan tidak unik. Program akan berhenti. ");
            }
        }
    }

    std::cin >> bufferSize;
    if (std::cin.fail() || bufferSize < 0) {
        throw std::invalid_argument("Ukuran buffer tidak valid. Program akan berhenti. ");
    }

    std::cin >> matWidth >> matHeight;
    if (std::cin.fail() || matWidth <= 0 || matHeight <= 0) {
        throw std::invalid_argument("Dimensi matriks tidak valid. Program akan berhenti.");
    }
    matrix.resize(matHeight, std::vector<std::string>(matWidth));
    for (int i = 0; i < matHeight; i++) {
        for (int j = 0; j < matWidth; j++) {
            matrix[i][j] = tokens[rand() % numTokens];
        }
    }

    std::cin >> numSeq;
    if (std::cin.fail() || numSeq < 0) {
        throw std::invalid_argument("Jumlah sekuens tidak valid. Program akan berhenti. ");
    }
    std::cin >> maxSeqSize;
    if (std::cin.fail() || maxSeqSize < 0) {
        throw std::invalid_argument("Ukuran maksimal sekuens tidak valid. Program akan berhenti. ");
    }

    for (int i = 0; i < numSeq; i++) {
        Sequence seq = generateSequence();
        while (!isUniqueSequence(seq)) {
            seq = generateSequence();
        }
        sequences.push_back(seq);
    }

    if (std::cin.fail()) {
        throw std::invalid_argument("Masukan dari keyboard tidak valid. Program akan berhenti. ");
    }
};
```

```cpp
void displayInputInfo() {
    std::cout << std::endl << "Matriks yang digunakan: " << std::endl;
    for (const auto& row : matrix) {
        for (const auto& element : row) {
            std::cout << element << ' ';
        }
        std::cout << std::endl;
    }

    std::cout << std::endl << "Sekuens yang digunakan: " << std::endl;
    for (const auto& seq : sequences) {
        for (const auto& token : seq.seqArray) {
            std::cout << token << ' ';
        }
        std::cout << std::endl << "Reward: " << seq.reward << std::endl;
    }
}
```

```cpp
void displayOutput() {
    std::cout << std::endl;

    if (maxReward != 0) {
        std::cout << maxReward << std::endl;

        for (const auto& coord : maxBuffer) {
            std::cout << coord.token << ' ';
        }
        std::cout << std::endl;

        for (const auto& coord : maxBuffer) {
            std::cout << coord.col << ", " << coord.row << std::endl;
        }
    } else {
        std::cout << maxReward << std::endl;
    }

    std::cout << std::endl;
}
```

```cpp
void saveSolution() {
    std::string path;
    std::cout << "Mohon masukkan path file: ";
    std::cin >> path;

    while (!((path.length() >= 4) && (path.substr(path.length() - 4) == ".txt"))) {
        std::cout << "Path yang diberikan bukan file txt. Mohon masukkan ulang path: ";
        std::cin >> path;
    }

    std::ofstream outputFile(path);

    if (maxReward != 0) {
        outputFile << maxReward << std::endl;

        for (const auto& coord : maxBuffer) {
            outputFile << coord.token << ' ';
        }
        outputFile << std::endl;

        for (const auto& coord : maxBuffer) {
            outputFile << coord.col << ", " << coord.row << std::endl;
        }
    } else {
        outputFile << maxReward << std::endl;
    }

    outputFile << std::endl << timeTaken << " ms";
    outputFile.close();

    std::cout << std::endl << "Solusi berhasil disimpan!" << std::endl;
}
```

```cpp
int main() {
    srand(static_cast<unsigned int>(time(nullptr))); // set seed

    std::cout << R"(
_________        ___.                 _________             __     ________________________    ________      .__
\_   ___ \___.__.\_ |__   ___________\______   \__ __  ____ |  | __ \_____  \   _  \______  \  /  _____/ ____ |  |__ __ ___________
/    \  \<   |  || __ \_/ __ \_  __ \     ___/  |  \/    \|  |/ /  /  ____/  /_\  \   /    /   / \_____  \ /  _ \|  |\  \/ // __ \_  __ \
\     \___\___  || \_\ \  ___/|  | \/    |   |  |  /   |  \    <  /       \  \_/   \ /    /   /   /        (  <_> )  |_\   /\  ___/|  | \/
 \______  / ____||___  /\___  >__|       |____/|____/|___|  /__|_ \ \_______ \_____  /____/   /____/   /_______  /\____/|____/\_/  \___  >__|
)" << std::endl << std::endl;
    std::cout << "Welcome to Cyberpunk 2077 Breach Protocol Solver!" << std::endl << std::endl;

    // get input
    int opt;
    std::cout << "1. File" << std::endl;
    std::cout << "2. Masukan keyboard" << std::endl;
    std::cout << "Pilih metode inputmu: ";
    std::cin >> opt;

    while (opt != 1 && opt != 2) {
        std::cout << std::endl <<  "Mohon pilih antara 1 atau 2." << std::endl;
        std::cout << "1. File" << std::endl;
        std::cout << "2. Masukan keyboard" << std::endl;
        std::cout << "Pilih metode inputmu: ";
        std::cin >> opt;
    }

    if (opt == 1) {
        readFromFile();
    } else {
        readFromInput();
        displayInputInfo();
    }
```

```cpp
    // process input while calculating time taken
    auto startTime = std::chrono::high_resolution_clock::now();

    Coordinate start(0, 0, ""); // initialize starting coordinate
    std::vector<Coordinate> initBuffer;
    findPossibleMoves(start, initBuffer, false);
    displayOutput();

    auto endTime = std::chrono::high_resolution_clock::now();
    auto duration = std::chrono::duration_cast<std::chrono::milliseconds>(endTime - startTime);
    timeTaken = duration.count();
    std::cout << duration.count() << " ms" << std::endl;


    // save solution
    char opt2;
    std::cout << std::endl << "Apakah ingin menyimpan solusi? (y/n) ";
    std::cin >> opt2;

    while (opt2 != 'y' && opt2 != 'n' && opt2 != 'Y' && opt2 != 'N') {
        std::cout << std::endl <<  "Mohon ketik y atau n.";
        std::cout << std::endl << "Apakah ingin menyimpan solusi? (y/n) ";
        std::cin >> opt2;
    }

    if (opt2 == 'y' || opt2 == 'Y') {
        saveSolution();
    }

    std::cout << std::endl << "Terima kasih!" << std::endl << std::endl;

    return 0;
}
```

BAB III

SCREENSHOT HASIL TEST

1. Test Case I

Masukan diterima melalui berkas.

```
tc1.txt
7
6 5
DD CC AA BB CC AA
AA EE BB CC BB CC
DD AA AA DD CC CC
DD CC EE DD EE DD
BB AA EE EE CC AA
4
DD CC AA
40
CC EE
37
DD EE
42
AA CC BB DD CC
38
```

```
PS C:\Users\LENOVO\Koding\Semester 4\Strategi Algoritma\Tucil 1\Tucil1_13522108\src> ../bin/solver.exe

Welcome to Cyberpunk 2077 Breach Protocol Solver!

1. File
2. Masukan keyboard
Pilih metode inputmu: 1
Mohon masukkan path file: tc1.txt

119
CC EE DD CC AA DD EE
5, 1
5, 4
4, 4
4, 2
1, 2
1, 4
3, 4

404 ms

Apakah ingin menyimpan solusi? (y/n) y
Mohon masukkan path file: ../test/out1.txt

Solusi berhasil disimpan!

Terima kasih!
```

test > out1.txt

```
119
CC EE DD CC AA DD EE
5, 1
5, 4
4, 4
4, 2
1, 2
1, 4
3, 4

404 ms
```

2. **Test Case II**

   Masukan diterima melalui *keyboard*.

```
PS C:\Users\LENOVO\Koding\Semester 4\Strategi Algoritma\Tucil 1\Tucil1_13522108\src> ../bin/solver.exe

Welcome to Cyberpunk 2077 Breach Protocol Solver!

1. File
2. Masukan keyboard
Pilih metode inputmu: 2

Masukkan informasi tentang matriks dan sekuens yang digunakan:
6
11 22 33 44 55 66
8
6 7
5
5
```

```
Matriks yang digunakan:
66 55 11 55 66 55
11 55 66 55 33 22
33 33 33 44 44 55
55 44 66 44 22 22
55 55 44 55 22 22
22 55 33 55 66 66
44 11 33 66 44 44

Sekuens yang digunakan:
44 66 11
Reward: 14
55 11 33
Reward: 2
22 33 55 22 22
Reward: 48
66 33 33 22 22
Reward: 17
22 44 44 22
Reward: 39
```

```
48
55 55 22 33 55 22 22
2, 1
2, 6
1, 6
1, 3
6, 3
6, 4
5, 4

10748 ms

Apakah ingin menyimpan solusi? (y/n) y
Mohon masukkan path file: ../test/out2.txt

Solusi berhasil disimpan!

Terima kasih!
```

test > out2.txt

```
1   48
2   55 55 22 33 55 22 22
3   2, 1
4   2, 6
5   1, 6
6   1, 3
7   6, 3
8   6, 4
9   5, 4
10
11  10748 ms
```

3. **Test Case III**

   Masukan memiliki *reward* negatif dan diterima melalui berkas.

tc3.txt

```
5
4 4
7A E9 1C 1C
55 7A 1C E9
E9 BD E9 BD
7A 1C 1C E9
4
7A 55
30
55 E9 BD
25
E9 BD
-10
55 E9 1C
20
```

```
PS C:\Users\LENOVO\Koding\Semester 4\Strategi Algoritma\Tucil 1\Tucil1_13522108\src> ../bin/solver.exe

Welcome to Cyberpunk 2077 Breach Protocol Solver!

1. File
2. Masukan keyboard
Pilih metode inputmu: 1
Mohon masukkan path file: tc3.txt

50
7A 55 E9 1C
1, 1
1, 2
4, 2
4, 1

5 ms

Apakah ingin menyimpan solusi? (y/n) y
Mohon masukkan path file: ../test/out3.txt

Solusi berhasil disimpan!

Terima kasih!
```

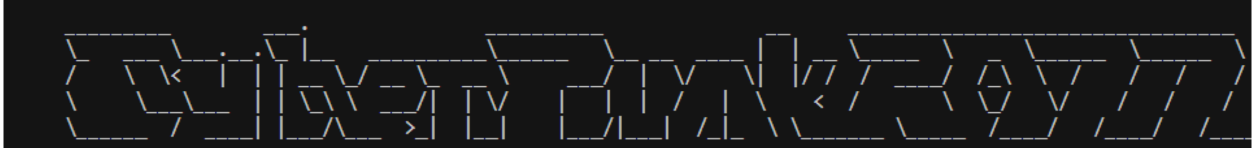
test > out3.txt

```
1   50
2   7A 55 E9 1C
3   1, 1
4   1, 2
5   4, 2
6   4, 1
7
8   5 ms
```

4. **Test Case IV**

   Masukan memiliki *reward* negatif dan diterima melalui *keyboard*.

```
PS C:\Users\LENOVO\Koding\Semester 4\Strategi Algoritma\Tucil 1\Tucil1_13522108\src> ../bin/solver.exe
```



```
Welcome to Cyberpunk 2077 Breach Protocol Solver!

1. File
2. Masukan keyboard
Pilih metode inputmu: 2

Masukkan informasi tentang matriks dan sekuens yang digunakan:
5
AA BB CC DD EE
6
5 5
4
4
```

```
Matriks yang digunakan:
CC DD AA AA BB
EE EE AA AA EE
EE AA EE BB BB
DD CC EE CC EE
DD BB AA EE CC

Sekuens yang digunakan:
DD BB BB BB
Reward: -8
AA BB EE CC
Reward: 1
AA BB AA
Reward: -22
DD DD AA AA
Reward: -32
```

```
1
CC EE AA BB EE CC
1, 1
1, 3
2, 3
2, 5
4, 5
4, 4

49 ms

Apakah ingin menyimpan solusi? (y/n) y
Mohon masukkan path file: ../test/out4.txt

Solusi berhasil disimpan!

Terima kasih!
```

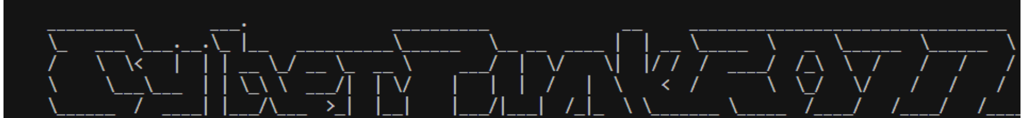
```
test > out4.txt
1
CC EE AA BB EE CC
1, 1
1, 3
2, 3
2, 5
4, 5
4, 4

49 ms
```

5. **Test Case V**

Masukan diterima melalui berkas dan tidak mempunyai solusi.

```
tc5.txt
5
4 4
7A E9 1C 1C
55 7A 1C E9
E9 BD E9 BD
7A 1C 1C E9
1
BD BD BD BD BD
10000
```

```
PS C:\Users\LENOVO\Koding\Semester 4\Strategi Algoritma\Tucil 1\Tucil1_13522108\src> ../bin/solver.exe
```



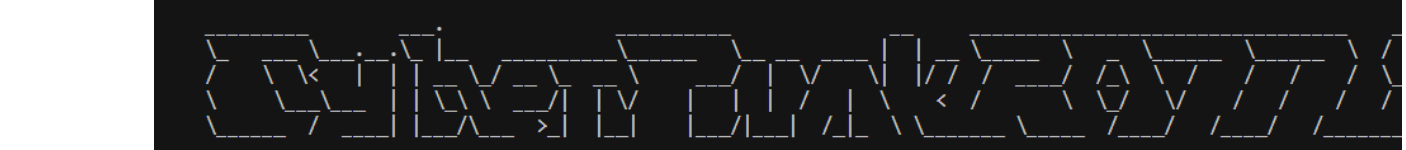
```
Welcome to Cyberpunk 2077 Breach Protocol Solver!

1. File
2. Masukan keyboard
Pilih metode inputmu: 1
Mohon masukkan path file: tc5.txt

0

1 ms

Apakah ingin menyimpan solusi? (y/n) y
Mohon masukkan path file: ../test/out5.txgt
Path yang diberikan bukan file txt. Mohon masukkan ulang path: ../test/out5.txt

Solusi berhasil disimpan!

Terima kasih!
```

```
test > out5.txt
0

1 ms
```

6. **Test Case VI**

Masukan diterima melalui *keyboard* dan tidak mempunyai solusi.

```
PS C:\Users\LENOVO\Koding\Semester 4\Strategi Algoritma\Tucil 1\Tucil1_13522108\src> ../bin/solver.exe
```



```
Welcome to Cyberpunk 2077 Breach Protocol Solver!

1. File
2. Masukan keyboard
Pilih metode inputmu: 2

Masukkan informasi tentang matriks dan sekuens yang digunakan:
5
11 22 33 44 55
5
4 4
1
10
```

```
Matriks yang digunakan:
44 22 33 44
44 33 33 55
33 33 33 44
33 33 11 44

Sekuens yang digunakan:
11 33 11
Reward: 26

0

1 ms

Apakah ingin menyimpan solusi? (y/n) y
Mohon masukkan path file: ../test/out6.txt

Solusi berhasil disimpan!

Terima kasih!
```

```
test > out6.txt
1   0
2
3   1 ms
```

LINK KE REPOSITORI

Pranala ke *repository*: https://github.com/neokoda/Tucil1_13522108

LAMPIRAN

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		✓