LAPORAN TUGAS KECIL 1

IF2211 - STRATEGI ALGORITMA

"Penyelesaian *Cyberpunk* 2077 *Breach Protocol* dengan Algoritma *Brute Force*"

Dosen:

Ir. Rila Mandala, M.Eng., Ph.D.

Monterico Adrian, S.T., M.T.

Mahasiswa:

Farhan Raditya Aji (13522142)

PROGRAM STUDI TEKNIK INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

SEMESTER I TAHUN 2023/2024

DAFTAR ISI

BAB I

Algoritma *Brute Force*

Algoritma brute force adalah sebuah algoritma yang menggunakan pendekatan langsung untuk menyelesaikan masalah dengan memeriksa semua solusi yang mungkin secara menyeluruh. Berikut adalah langkah-langkah implementasi algoritma *brute force* pada *minigames breach protocol* pada *game cyberpunk 2077* :

1. Inisialisasi Matriks dan *Sequence* : Program dimulai dengan inisialisasi matriks yang merepresentasikan token-token dan sebuah *sequence*. Inisialisasi dapat diinput melalui file atau dengan *random generating* .
2. Rute Awal : Algoritma *brute force* dimulai dari baris paling atas namun tidak selalu di paling ujung kiri.
3. Mencari Semua Kemungkinan: Algoritma brute force melakukan pencarian pada semua kemungkinan rute yang mungkin dimulai dari posisi awal. Pencarian ini dilakukan dengan rekursi dan looping yang berulang-ulang untuk mencoba semua kemungkinan rute.
4. Rekursi dan Perulangan: Selama proses pencarian, algoritma dilakukan dengan rekursif dan perulangan untuk mendapat semua kemungkinan rute dari posisi saat ini. Misalnya, jika kita berada di posisi (i, j), kita mungkin ingin mencoba semua kemungkinan rute ke sel-sel yang bersebelahan atau terhubung langsung dengan (i, j) yang sesuai dengan kondisi pergerakan sekarang saat vertikal atau horizontal.
5. Penandaan dan Penyimpanan Solusi: Setiap kali algoritma menemukan rute yang valid yang mencapai tujuan atau memenuhi kriteria tertentu, itu menandai rute tersebut dan mungkin menyimpannya untuk perbandingan lebih lanjut. Ini dapat dilakukan dengan menggunakan struktur data seperti array atau vektor untuk menyimpan rute yang ditemukan.

6. *Backtrack*: Jika algoritma mencapai jalan buntu, maka algoritma akan menggunakan teknik backtracking untuk kembali ke langkah sebelumnya dan mencoba rute atau langkah alternatif lainnya. Ini memastikan bahwa algoritma mengeksplorasi semua kemungkinan dengan benar.
7. Perhitungan *Reward*: Jika algoritma sudah mendapat rute sesuai dengan buffer atau sudah buntu , maka sebelum melakukan *backtrack* atau melanjutkan ke rute lain , algoritma akan menghitung total *reward* yang didapat dari rute sesuai dengan *sequence*-nya. Jika hasilnya lebih besar dari *reward* yang sebelumnya , maka *reward* , rute , dan koordinat dari rute yang dilewati sekarang akan disimpan.
8. Output atau Penyimpanan Hasil: Pada saat selesai maka akan didapatkan hasil *reward* maksimal, rute , dan koordinat dari rute yang dilewati.

BAB 2

Source Code

Source code diimplementasikan menggunakan bahasa C++, berikut adalah *source code*-nya :

1. input.cpp

File ini berisikan program yang digunakan untuk menerima input baik dari file atau CLI.

```
#include <iostream>
#include <string>
#include <fstream>
#include <sstream>
#include <vector>
#include <ctime>
#include <cstdlib>
#include <algorithm>

using namespace std;

struct Tucil
{
    int buffer;
    int row;
    int col;
    int jumlah_sequence;
    int sequence_length;
    vector<vector<string>> matriks;
    vector<vector<string>> sequence;
    vector<int> reward_sequence;
};

int countWords(const string &str)
{
    stringstream ss(str);
    string word;
    int count = 0;

    while (ss >> word)
    {
        count++;
    }
    return count;
```

```
}


vector<vector<string>> generateMatriks(Tucil tucil, vector<string> token, int
jumlah_token_unik)
{
   srand(time(0));
   vector<vector<string>> matriks(tucil.row, vector<string>(tucil.col));

   for (int i = 0; i < tucil.row; i++)
   {
      for (int j = 0; j < tucil.col; j++)
      {
         matriks[i][j] = token[rand() % jumlah_token_unik];
      }
   }
   return matriks;
}

vector<vector<string>> generateSequence(Tucil tucil , vector<string> token, int
jumlah_token_unik)
{
   srand(time(0));
   vector<vector<string>> sequence(tucil.jumlah_sequence,
vector<string>(tucil.sequence_length));
   for (int i = 0; i < tucil.jumlah_sequence; i++)
   {
      int seq_random = rand() % (tucil.sequence_length - 2 + 1) + 2;
      for (int j = 0; j < seq_random; j++)
      {
         sequence[i][j] = token[rand() % jumlah_token_unik];
      }
   }
   return sequence;
}

vector<int> generateRewardSequence(Tucil tucil)
{
   srand(time(0));
   vector<int> reward_sequence(tucil.jumlah_sequence);
   for (int i = 0; i < tucil.jumlah_sequence; i++)
   {
      reward_sequence[i] = rand() % 100;
   }
   return reward_sequence;
}

void checkData(Tucil tucil)
{
```

```
    cout << "\nBerikut adalah data yang dihasilkan: \n\n";
    cout << "Buffer size: " << tucil.buffer << endl;
    cout << "Row matriks: " << tucil.row << endl;
    cout << "Col matriks: " << tucil.col << endl;
    cout << "\nMatriks: " << endl;
    for (int i = 0; i < tucil.row; i++)
    {
        for (int j = 0; j < tucil.col; j++)
        {
            cout << tucil.matriks[i][j] << " ";
        }
        cout << endl;
    }
    cout << "\nJumlah sequence: " << tucil.jumlah_sequence << endl;
    cout << "Sequence length: " << tucil.sequence_length << endl;
    cout << "\nSequences: " << endl;
    for (int i = 0; i < tucil.jumlah_sequence; i++)
    {
        cout << "Sequence " << i + 1 << ": ";
        for (int j = 0; j < tucil.sequence[i].size(); j++)
        {
            cout << tucil.sequence[i][j] << " ";
        }
        cout << endl;
    }
    cout << "\nReward sequences: " << endl;
    for (int i = 0; i < tucil.jumlah_sequence; i++)
    {
        cout << "Sequence " << i + 1 << ": " << tucil.reward_sequence[i] << endl;
    }
    cout << endl;
}

Tucil read_file()
{
    Tucil tucil;
    string filename, line;
    int temp;
    string path = "../test/";
    cout << "Masukkan nama file tanpa (.txt): ";
    cin >> filename;
    filename += ".txt";
    ifstream file(path + filename);

    while (!file.is_open())
    {
        cout << "File tidak ditemukan" << endl;
        cout << "Masukkan nama file tanpa (.txt): ";
        cin >> filename;
```

```
    filename += ".txt";
    file.open(filename);
  }

  file >> tucil.buffer >> tucil.col >> tucil.row;

  tucil.matriks.resize(tucil.row, vector<string>(tucil.col));
  for (int i = 0; i < tucil.row; i++)
  {
    for (int j = 0; j < tucil.col; ++j)
    {
      file >> tucil.matriks[i][j];
    }
  }

  file >> tucil.jumlah_sequence;
  getline(file, line);
  tucil.sequence_length = 0;

  tucil.sequence.resize(tucil.jumlah_sequence, vector<string>(tucil.sequence_length));
  for (int i = 0; i < tucil.jumlah_sequence; i++)
  {
    getline(file, line);
    temp = countWords(line);
    if (temp > tucil.sequence_length)
    {
      tucil.sequence_length = temp;
      for (int i = 0; i < tucil.jumlah_sequence; i++)
      {
        tucil.sequence[i].resize(tucil.sequence_length);
      }
    }
    stringstream ss(line);
    string word;
    vector<string> words;
    int j = 0;
    while (ss >> word)
    {
      tucil.sequence[i][j] = word;
      j++;
    }
    getline(file, line);
    tucil.reward_sequence.push_back(stoi(line));
  }

  file.close();
  return tucil;
}
```

```
Tucil CLI()
{
   Tucil tucil;
   string temp;
   int jumlah_token_unik;
   vector<string> token;

   cout << "Masukkan jumlah token unik: " << endl;
   cin >> jumlah_token_unik;
   token.resize(jumlah_token_unik);
   cout << "Masukkan token unik sekaligus (contoh: 7A 55 E9): " << endl;
   for (int i = 0; i < jumlah_token_unik; ++i)
   {
      cin >> temp;
      token[i] = temp;
   }

   cout << "Masukkan buffer size: " << endl;
   cin >> tucil.buffer;
   cout << "Masukkan row x col matriks (contoh: 6 6): " << endl;
   cin >> tucil.row ;
   cin >> tucil.col;
   cout << "Masukkan jumlah sequence: " << endl;
   cin >> tucil.jumlah_sequence;
   cout << "Masukkan sequence length: " << endl;
   cin >> tucil.sequence_length;
   cout << endl;

   tucil.matriks = generateMatriks(tucil, token, jumlah_token_unik);
   tucil.sequence = generateSequence(tucil, token, jumlah_token_unik);
   tucil.reward_sequence = generateRewardSequence(tucil);

   return tucil;
}
```

2. solver.cpp

File ini berisikan program yang akan mencari semua kemungkinan rute dan *reward*-nya

```
#include <iostream>
#include <vector>
#include <utility>
#include "input.cpp"

using namespace std;

struct Point
```

```
{
    int row;
    int col;
    Point(int r, int c) : row(r), col(c) {}
};

struct RouteInfo
{
    vector<string> route;
    int reward;
    RouteInfo(vector<string> &r, int rw) : route(r), reward(rw) {}
};

int countSeqLength(Tucil tucil, int idx)
{
    int count = 0;
    for (int i = 0; i < tucil.sequence_length; i++)
    {
        if (tucil.sequence[idx][i] != "")
        {
            count++;
        }
    }
    return count;
}

int countScore(Tucil tucil, vector<string> temp)
{
    int score = 0;
    int size = temp.size();
    for (int i = 0; i < size; i++)
    {
        for (int j = 0; j < tucil.jumlah_sequence; j++)
        {
            int length = countSeqLength(tucil, j);
            if (size - i >= length)
            {
                int count = 0;
                for (int k = 0; k < length; k++)
                {
                    if (temp[i + k] == tucil.sequence[j][k])
                    {
                        count++;
                    }
                }
                if (count == length)
                {
                    score += tucil.reward_sequence[j];
                }
```

```
            }
        }
    }

    if (score == 0)
    {
        score = -1;
    }

    return score;
}

int maxReward = 0;
vector<string> maxRoute;
vector<Point> maxRoutePoints;

void searchRoute(Tucil tucil, vector<vector<string>> &matrix, int row, int col,
vector<string> &route, vector<Point> &routePoints, vector<vector<bool>> &visited, int
buffer, bool isVertical)
{
    route.push_back(matrix[row][col]);
    routePoints.push_back(Point(row, col));
    visited[row][col] = true;
    int tempReward = countScore(tucil, route);
    if (tempReward > maxReward)
    {
        maxReward = tempReward;
        maxRoute = route;
        maxRoutePoints = routePoints;
    }

    if (route.size() == buffer)
    {
        route.pop_back();
        routePoints.pop_back();
        visited[row][col] = false;
        return;
    }

    if (isVertical)
    {
        for (int i = 0; i < tucil.row; ++i)
        {
            if (!visited[i][col])
            {
                searchRoute(tucil, matrix, i, col, route, routePoints, visited, buffer, !isVertical);
            }
        }
    }
```

```
    else
    {
        for (int j = 0; j < tucil.col; ++j)
        {
            if (!visited[row][j])
            {
                searchRoute(tucil, matrix, row, j, route, routePoints, visited, buffer, !isVertical);
            }
        }
    }

    route.pop_back();
    routePoints.pop_back();
    visited[row][col] = false;
}
```

3. main.cpp

File ini berisikan main program yang akan dijalankan.

```
#include "solver.cpp"
#include <iostream>
#include <fstream>
#include <chrono>

using namespace std::chrono;
using namespace std;

int main()
{
    string choice,save;
    Tucil tucil;
    cout << "Selamat Datang di Program Cyberpunk 2077 Breach Protocol dengan
Algoritma Brute Force" << endl;
    cout << "Pilih Inputan: " << endl;
    cout << "1. CLI" << endl;
    cout << "2. File.txt" << endl;
    cout << "Pilihan: ";

    cin >> choice;
    while (choice != "1" && choice != "2")
    {
        cout << "Pilihan tidak valid\n"
             << endl;
        cout << "Pilih Inputan: " << endl;
        cout << "1. CLI" << endl;
```

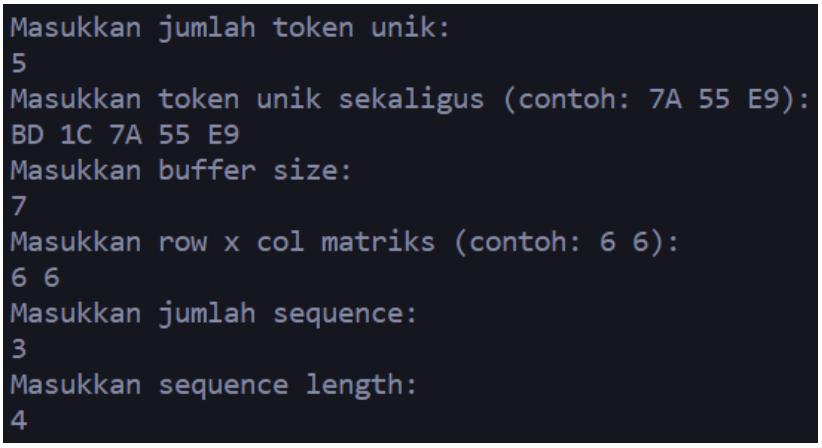
```
    cout << "2. File.txt" << endl;
    cout << "Pilihan: ";
    cin >> choice;
}
cout << endl;

if (choice == "1")
{
    tucil = CLI();
}
else
{
    tucil = read_file();
}

vector<string> route;
vector<Point> routePoints;
vector<vector<bool>> visited(tucil.row, vector<bool>(tucil.col, false));

auto start = high_resolution_clock::now();

for (int j = 0; j < tucil.col; ++j)
{
    searchRoute(tucil, tucil.matriks, 0, j, route, routePoints, visited, tucil.buffer, true);
}
cout << maxReward << endl;
if (maxReward != 0)
{
    for (int i = 0; i < maxRoute.size(); ++i)
    {
        cout << maxRoute[i] << " ";
    }
    cout << endl;
    for (int i = 0; i < maxRoutePoints.size(); ++i)
    {
        cout << maxRoutePoints[i].col +1 << "," << maxRoutePoints[i].row + 1 << endl;
    }
}
auto stop = high_resolution_clock::now();
auto duration = duration_cast<milliseconds>(stop - start);
cout << endl <<duration.count() << " ms" << endl;
cout << "\nApakah ingin menyimpan solusi? (y/n)" << endl;
cin >> save;

while (save != "y" && save != "n")
{
    cout << "Pilihan tidak valid\n"<< endl;
    cout << "Apakah ingin menyimpan solusi? (y/n)" << endl;
    cin >> save;
```

```
    }

    if (save == "y")
    {
        string filename;
        string path = "../test/";
        cout << "Masukkan nama file tanpa (.txt): ";
        cin >> filename;
        filename += ".txt";
        ofstream file(path + filename);

        if (!file.is_open()) {
            cout << "Gagal membuka file untuk penulisan" << endl;
            return 1;
        }

        file << maxReward << endl;
        for (int i = 0; i < maxRoute.size(); ++i)
        {
            file << maxRoute[i] << " ";
        }
        file << endl;
        for (int i = 0; i < maxRoutePoints.size(); ++i)
        {
            file << maxRoutePoints[i].col +1 << ", " << maxRoutePoints[i].row + 1 << endl;
        }
        file << endl << duration.count() << " ms";
        file.close();
    }
    return 0;
}
```

BAB 3

Input dan Output

1. CLI
    a. Input :



```
Berikut adalah data yang dihasilkan:

Buffer size: 7
Row matriks: 6
Col matriks: 6

Matriks:
55 55 7A 7A 7A E9
1C 1C 7A BD 7A 7A
E9 55 BD 55 E9 1C
BD E9 BD 55 1C E9
E9 55 BD BD 1C 55
E9 E9 BD E9 7A BD

Jumlah sequence: 3
Sequence length: 4

Sequences:
Sequence 1: 55 7A
Sequence 2: 7A E9 1C 1C
Sequence 3: BD 7A 7A

Reward sequences:
Sequence 1: 58
Sequence 2: 23
Sequence 3: 27
```

Output :

```
116
55 E9 55 7A 1C 55 7A
1, 1
1, 5
6, 5
6, 2
2, 2
2, 1
3, 1

12902 ms

Apakah ingin menyimpan solusi? (y/n)
```

b. Input :

```
Masukkan jumlah token unik:
5
Masukkan token unik sekaligus (contoh: 7A 55 E9):
BD 1C 7A 55 E9
Masukkan buffer size:
4
Masukkan row x col matriks (contoh: 6 6):
5 5
Masukkan jumlah sequence:
3
Masukkan sequence length:
4
```

```
Berikut adalah data yang dihasilkan:

Buffer size: 4
Row matriks: 5
Col matriks: 5

Matriks:
55 55 E9 55 7A
BD BD 55 BD 1C
55 55 E9 BD E9
1C 55 1C 1C 1C
55 1C 55 E9 1C

Jumlah sequence: 3
Sequence length: 4

Sequences:
Sequence 1: 55 E9 55
Sequence 2: BD BD 55 BD
Sequence 3: 55 55 E9 BD

Reward sequences:
Sequence 1: 73
Sequence 2: 28
Sequence 3: 89
```

Output :

```
89
55 55 E9 BD
1, 1
1, 5
4, 5
4, 2

36 ms

Apakah ingin menyimpan solusi? (y/n)
y
```

c. Input :

```
Masukkan jumlah token unik:
5
Masukkan token unik sekaligus (contoh: 7A 55 E9):
BD 1C 7A 55 E9
Masukkan buffer size:
6
Masukkan row x col matriks (contoh: 6 6):
7 7
Masukkan jumlah sequence:
3
Masukkan sequence length:
4
```

```
Berikut adalah data yang dihasilkan:

Buffer size: 6
Row matriks: 7
Col matriks: 7

Matriks:
E9 55 1C 7A E9 E9 7A
1C 7A E9 E9 BD 55 BD
E9 E9 55 7A 7A 7A BD
E9 E9 7A 7A 55 55 55
1C 1C BD E9 E9 E9 55
55 1C 55 7A 1C BD 55
7A E9 E9 55 E9 BD 1C

Jumlah sequence: 3
Sequence length: 4

Sequences:
Sequence 1: 55 1C
Sequence 2: E9 E9 7A 1C
Sequence 3: E9 E9 BD 55

Reward sequences:
Sequence 1: 24
Sequence 2: 28
Sequence 3: 66
```

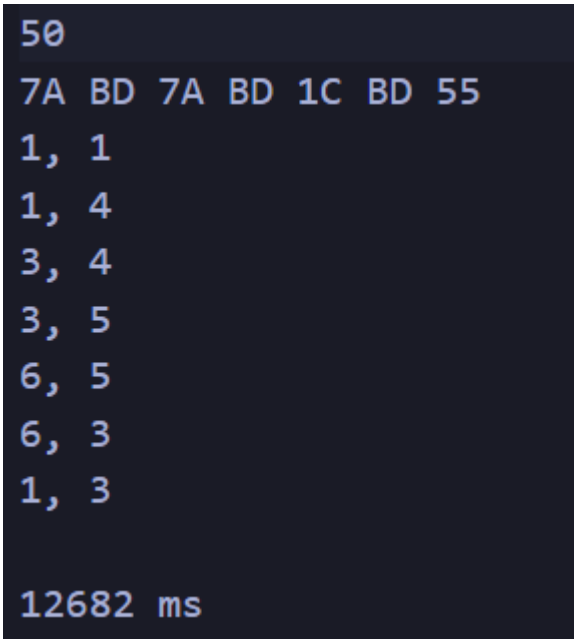
Output :

```
90
E9 E9 BD 55 55 1C
1, 1
1, 3
7, 3
7, 4
5, 4
5, 6

8822 ms

Apakah ingin menyimpan solusi? (y/n)
```

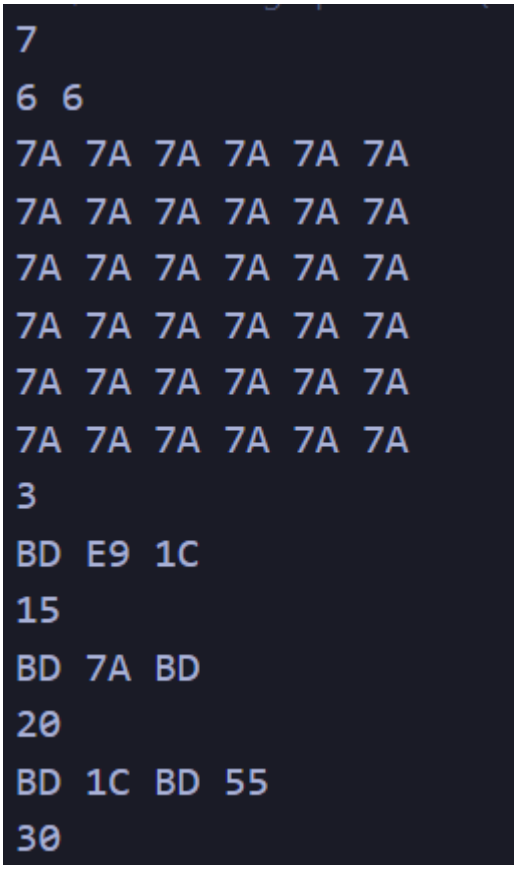
2. File.txt
    a. Input :

```
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
```

Output :



b. Input :



Output :

c. Input :

```
5
7 6
7A BD BD 7A 1C E9 1C
55 55 55 55 E9 BD 7A
1C 55 55 7A 55 7A 1C
BD BD 1C 1C 55 E9 7A
1C 55 7A 55 E9 7A E9
1C 1C BD E9 55 1C 7A
3
BD BD
12
1C E9
20
55 55 55
40
```

Output :

```
80
7A 55 55 55 55
1, 1
1, 2
2, 2
2, 3
3, 3

845 ms
```

BAB 4

Pranala Repository Github

Link Github :

https://github.com/sibobbbbbb/Tucil1_13522142.git

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		✔