

**LAPORAN TUGAS KECIL I**  
**IF2211 STRATEGI ALGORITMA**

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

William Glory Henderson 13522113



SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA  
INSTITUT TEKNOLOGI BANDUNG

2023

## **Daftar Isi**

Daftar Isi.....	2
Algoritma Brute Force.....	3
Source Program.....	5
Tangkapan Layar Input dan Output.....	11

## **ALGORITMA BRUTE FORCE**

Algoritma brute force merupakan algoritma yang menggunakan pendekatan yang dibuat secara langsung hanya untuk memecahkan persoalan yang diminta. Oleh karena itu, algoritma brute force bukan merupakan algoritma yang optimal karena algoritma tersebut seringkali menyelesaikan masalah dengan memeriksa semua kemungkinan solusi yang ada dan pada akhirnya baru mengambil solusi yang terbaik.

Pada program Cyberpunk 2077 breach protocol ini diminta menggunakan algoritma brute force untuk menentukan kode yang dipakai untuk menyelesaikan mini game simulasi peretasan. Langkah-langkah menyelesaikan permasalahan dengan algoritma brute force adalah sebagai berikut :

1. Proses menerima input

Pada proses ini, program akan menerima input/masukan dalam bentuk file .txt atau masukan dari CLI (Command Line Interface). Masukan yang diterima berupa ukuran buffer, matriks, elemen matriks, jumlah sekuens, isi sekuens, dan reward masing-masing sekuens. Program ini akan menerima ukuran buffer dalam bentuk integer dan menyimpannya ke variabel `buffer_size`. Untuk matriks dan elemennya akan disimpan ke variabel matriks. Untuk isi dan reward sekuens dibuat sebuah tipe data baru yaitu Sequence yang isinya terdiri dari isi sekuens (token) dan reward dari sekuens tersebut. Pada program ini juga dibuat tipe data baru yaitu Coordinate yang menyatakan posisi untuk token dan nama token tersebut pada matriks. Contohnya token 7A berada pada baris 1 kolom 1 sehingga token 7A memiliki koordinat 1,1.

2. Proses brute force

Pada proses ini, program akan mencari semua kemungkinan sekuens yang dapat dibentuk dari beberapa token dan menghitung reward yang didapat dari setiap kemungkinan tersebut. Proses pencarian solusi ini dimulai dengan memilih salah satu token pada baris pertama. Pemain hanya boleh bergerak vertikal terlebih dahulu kemudian dilanjutkan dengan horizontal, lalu vertikal, lalu horizontal, dan diulang kembali secara selang-seling sampai jumlah token yang dipakai mencapai batas maksimal yaitu sesuai dengan ukuran buffer. Sekuens yang dapat diambil minimal berupa 2 token. Oleh karena itu, setiap kemungkinan sekuens yang akan didapatkan dimulai dari 2 token lalu 3 token sampai ukuran buffer maksimal. Maka brute force

ini akan mencari semua kemungkinan sekuens dari jumlah token terkecil terlebih dahulu dan akan dilanjutkan ke jumlah token yang lebih besar.

Pada saat ditemukannya sebuah sekuens, akan dilakukan juga pengecekan reward secara langsung dengan membandingkan sekuens tersebut apakah memiliki sekuens reward atau tidak. Jika sekuens tersebut memiliki reward maka sekuens tersebut akan disimpan ke variabel `max_buffer` dan reward dari sekuens tersebut juga disimpan ke dalam variabel `max_reward`. Kedua variabel ini akan terus berubah jika ada sekuens yang memiliki reward yang lebih besar tetapi jika tidak ada atau sama maka variabel tersebut tidak berubah. Pencarian sekuens ini dilakukan secara rekursif untuk mencari semua kemungkinan yang dapat dibentuk dari batas minimal sekuens sampai batas maksimal dan menghindari kemungkinan sekuens yang dapat terlewat.

### 3. Proses menghasilkan output

Setelah selesai mencari semua sekuens, program ini akan menampilkan satu sekuens terbaik yaitu yang berisi reward maksimal dari algoritma brute force tersebut. Terdapat informasi mengenai jumlah reward yang didapat, token-token yang terdapat pada sekuens tersebut, koordinat token-token, dan waktu eksekusi kode. Untuk waktu eksekusi juga bergantung dengan pemakaian bahasa dan kecepatan pc.

## Source Program

1. Program menggunakan library time untuk menghitung waktu. Library sys digunakan untuk menghentikan kode ketika terjadi kesalahan. Library random digunakan untuk membuat matriks dan sekuens secara random. Library numpy digunakan untuk penggunaan array. Tipe data yang dibuat adalah Coordinate untuk menyimpan informasi mengenai alfanumerik token dan posisi token dalam kolom dan baris. Tipe data sequence digunakan untuk menyimpan informasi mengenai isi sekuens yaitu token-token dan juga reward dari sekuens tersebut.

```
import time
import sys
import random
import numpy as np

class Coordinate:
    def __init__(self, col, row, value):
        self.col = col
        self.row = row
        self.value = value

class Sequence:
    def __init__(self, pattern, reward):
        self.pattern = pattern
        self.reward = reward

buffer = []
max_buffer = []
sequences = []
buffer_size = 0
max_reward = 0
```

2. readInputFile digunakan untuk membaca masukan dari file .txt dengan format buffer\_size, matrix\_width (col), matrix\_height (row), matrix number\_of\_sequences, sequences\_1, sequences\_1\_reward, sequences\_2, sequences\_2\_reward, sequences\_n, sequences\_n\_reward

```
def readInputFile(file_path):
    while True:
        try:
            with open(file_path, 'r') as file:
                lines = file.readlines()
                buffer_size = int(lines[0])
                matrix_col, matrix_row = map(int, lines[1].split())
                matrix = [line.split() for line in lines[2:2+matrix_row]]

                check1 = True
                check2 = True
                for i in range(len(matrix)):
                    for j in range(len(matrix[0])):
                        if (len(matrix[i][j])) != 2 :
                            check1 = False
                        if (not(matrix[i][j].isalnum())) :
                            check2 = False

                if(check1 == False) :
                    print("Silahkan run ulang program kembali karena token tidak terdiri dari 2 karakter")
                    sys.exit()
                if(check2 == False) :
                    print("Silahkan run ulang program kembali karena token tidak terdiri dari alfanumerik")
                    sys.exit()

                num_sequences = int(lines[2 + matrix_row])
                current_line = 3 + matrix_row
                for _ in range(num_sequences):
                    sequence = lines[current_line].split()
                    current_line += 1

                    sequence_reward = int(lines[current_line])
                    current_line += 1

                    sequence_instance = Sequence(sequence, sequence_reward)
                    sequences.append(sequence_instance)

            return buffer_size, matrix, sequences

        except FileNotFoundError:
            file_path = input("File tidak ditemukan. Silahkan input ulang nama file: ")
```

3. `readInputTerminal` digunakan untuk membaca masukan dari terminal dengan format `jumlah_token_unik`, `token`, `ukuran_buffer`, `ukuran_matriks`, `jumlah_sekuens`, dan `ukuran_maksimal_sekuens`. Untuk matriks dan sekuens akan dihasilkan secara random dengan menggunakan library bawaan python.

```
def readInputTerminal():
    check1 = True
    check2 = True
    total_token = int(input(""))
    token = input("")
    token_list = token.split()

    for i in range(len(token_list)):
        if (len(token_list[i]) != 2):
            check1 = False
        if (not(token_list[i].isalnum())):
            check2 = False

    if(check1 == False):
        print("Silahkan run ulang program kembali karena token tidak terdiri dari 2 karakter")
        sys.exit()

    if(check2 == False):
        print("Silahkan run ulang program kembali karena token tidak terdiri dari alfanumerik")
        sys.exit()

    buffer_size = int(input(""))
    matrix_size = input("")
    matrix_col, matrix_row = map(int, matrix_size.split())
    matrix = np.array([[random.choice(token_list) for _ in range(matrix_col)] for _ in range(matrix_row)], dtype=object)

    total_sequence = int(input(""))
    sequence_size = int(input(""))

    for _ in range(total_sequence):
        seq = random.randint(2, sequence_size)
        sequence_token = [random.choice(token_list) for _ in range(seq)]
        reward = random.randint(1, 100)
        sequence_instance = Sequence(sequence_token, reward)
        sequences.append(sequence_instance)

    return buffer_size, matrix, sequences
```

4. `outputToFile` digunakan untuk menuliskan atau mengcopy hasil dari program ke dalam file baru atau file yang sudah tercantum.

```
def outputToFile(file_path, max_reward, max_buffer, start_time, end_time):
    try:
        with open(file_path, 'w') as file:
            if max_reward == 0:
                file.write(str(max_reward) + "\n")
                file.write("No Solution\n")
            else:
                file.write(str(max_reward) + "\n")
                for item in max_buffer:
                    file.write(str(item.value) + " ")
                file.write('\n')
                for item in max_buffer:
                    file.write(f'{item.col}, {item.row}\n')
            execution_time = round((end_time - start_time) * 1000)
            file.write(f'{execution_time} ms\n')

        print("Output telah disimpan ke", file_path)
    except IOError:
        print("Error: Unable to write to file", file_path)
```

5. hasSequence digunakan untuk memeriksa sekuens yang didapatkan melalui algoritma brute force memiliki sekuens reward atau tidak.

```
def hasSequence(seq, buff):
    has_sequence = True
    for j in range(len(buff) - len(seq.pattern) + 1):
        has_sequence = True
        for i in range(len(seq.pattern)):
            if seq.pattern[i] != buff[i+j].value:
                has_sequence = False
        if has_sequence:
            return True
    return False
```

6. countReward digunakan untuk menghitung reward yang didapatkan dari sekuens yang dihasilkan.

```
def countReward(buff):
    rewards = 0
    for i in range(len(sequences)):
        if hasSequence(sequences[i], buff):
            rewards += sequences[i].reward
    return rewards
```

7. hasPass digunakan untuk memeriksa token tersebut sudah pernah dilewati atau belum.

```
def hasPass(coord, buff):
    for i in range(len(buff)):
        if coord.row == buff[i].row and coord.col == buff[i].col:
            return True
    return False
```



8. findRoute digunakan untuk mencari semua kemungkinan sekuens yang ada dengan menggunakan rekursif dan akan membandingkan reward dari setiap sekuens.

```
def findRoute(coord, buff, vertical):
    global max_reward
    global max_buffer
    global buffer_size
    if coord.col == 0 and coord.row == 0:
        for i in range(len(matrix[0])):
            newCoord = Coordinate(i + 1, 1, matrix[0][i])
            if not hasPass(newCoord, buff):
                newBuffer = buff[:]
                newBuffer.append(newCoord)
                reward = countReward(newBuffer)
                if reward > max_reward:
                    max_reward = reward
                    max_buffer = newBuffer
                findRoute(newCoord, newBuffer, True)

    elif len(buff) == buffer_size:
        return

    elif vertical:
        for i in range(len(matrix)):
            newCoord = Coordinate(coord.col, i + 1, matrix[i][coord.col - 1])
            if not hasPass(newCoord, buff):
                newBuffer = buff[:]
                newBuffer.append(newCoord)
                reward = countReward(newBuffer)
                if reward > max_reward:
                    max_reward = reward
                    max_buffer = newBuffer
                findRoute(newCoord, newBuffer, False)

    elif not vertical:
        for i in range(len(matrix[0])):
            newCoord = Coordinate(i + 1, coord.row, matrix[coord.row - 1][i])
            if not hasPass(newCoord, buff):
                newBuffer = buff[:]
                newBuffer.append(newCoord)
                reward = countReward(newBuffer)
                if reward > max_reward:
                    max_reward = reward
                    max_buffer = newBuffer
                findRoute(newCoord, newBuffer, True)
```

## 9. Main program

[illegible]

```
print(f'Ukuran Buffer: {buffer_size}')
```

```
print(f'Ukuran Kolom Matrix: {len(matrix[0])}')
```

```
print(f'Ukuran Baris Matrix: {len(matrix)}')
```

```
print("Matriks: ")
```

```
for row in matrix:
```

```
    for element in row:
```

```
        print(element, end=" ")
```

```
    print()
```

```
print(f'Jumlah sekuens: {len(sequences)}')
```

```
print("Sekuens: ")
```

```
for i in range (len(sequences)):
```

```
    print(f'sekuens {sequences[i].pattern} memiliki reward {sequences[i].reward}')
```

```
for i in range (len(sequences)) :
```

```
    if (len(sequences[i].pattern)) < 2 :
```

```
        print("Silahkan run ulang program kembali karena sekuens reward kurang dari 2 token")
```

```
        sys.exit()
```

```
start_time = time.time()
```

```
start_coord = Coordinate(0, 0, "")
```

```
test = findRoute(start_coord, buffer, False)
```

```
end_time = time.time()
```

```

if max_reward == 0:
    print(max_reward)
    print("No Solution")
else:
    print(max_reward)
    for i in range(len(max_buffer)):
        print(max_buffer[i].value, end=" ")
    print('')
    for i in range(len(max_buffer)):
        print(f'{max_buffer[i].col}, {max_buffer[i].row}')
print(round((end_time - start_time) * 1000), 'ms')

copy = input("Apakah ingin menyimpan solusi? (y/n) : ")
while copy != "y" and copy != "n":
    copy = input("Silahkan pilih ulang untuk simpan (y/n) : ")

if copy == "y":
    file_name = input("Masukkan nama file untuk menyimpan output: ")
    new_path = f'test/{file_name}'
    outputToFile(new_path, max_reward, max_buffer, start_time, end_time)

elif copy == "n":
    sys.exit()

```

## Tangkapan Layar Input dan Output

Tangkapan Layar Input dan Output 1 dari file

```
WELCOME TO  
CYBERPUNK 2077  
BREACH PROTOCOL
```

```
1. Masukkan melalui file .txt  
2. Masukkan melalui CLI  
Pilih jenis masukkan (1/2): 1  
Masukkan nama file: test1.txt  
Ukuran Buffer: 7  
Ukuran Kolom Matrix: 6  
Ukuran Baris Matrix: 6  
Matriks:  
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  
Jumlah sekuens: 3  
Sekuens:  
sekuen ['BD', 'E9', '1C'] memiliki reward 15  
sekuen ['BD', '7A', 'BD'] memiliki reward 20  
sekuen ['BD', '1C', 'BD', '55'] memiliki reward 30  
50  
7A BD 7A BD 1C BD 55  
1, 1  
1, 4  
3, 4  
3, 5  
6, 5  
6, 3  
1, 3  
1075 ms  
Apakah ingin menyimpan solusi? (y/n) : y  
Masukkan nama file untuk menyimpan output: jawaban1.txt  
Output telah disimpan ke jawaban1.txt
```

```
test1.txt × ... jawaban1.txt ×  
test1.txt jawaban1.txt  
1 7 1 50  
2 6 6 2 7A BD 7A BD 1C BD 55  
3 7A 55 E9 E9 1C 55 3 1, 1  
4 55 7A 1C 7A E9 55 4 1, 4  
5 55 1C 1C 55 E9 BD 5 3, 4  
6 BD 1C 7A 1C 55 BD 6 3, 5  
7 BD 55 BD 7A 1C 1C 7 6, 5  
8 1C 55 55 7A 55 7A 8 6, 3  
9 3 9 1, 3  
10 BD E9 1C 10 1075 ms  
11 15 11  
12 BD 7A BD  
13 20  
14 BD 1C BD 55  
15 30
```

## Tangkapan Layar Input dan Output 2 dari file

```
WELCOME TO  
CYBERPUNK 2077  
BREACH PROTOCOL
```

```
1. Masukkan melalui file .txt  
2. Masukkan melalui CLI  
Pilih jenis masukkan (1/2): 1  
Masukkan nama file: test2.txt  
Ukuran Buffer: 6  
Ukuran Kolom Matrix: 6  
Ukuran Baris Matrix: 7  
Matriks:  
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  
BD 55 BD 7A 1C 1C  
Jumlah sekuens: 3  
Sekuens:  
sekuen ['BD', 'E9', '1C'] memiliki reward -10  
sekuen ['BD', '7A', 'BD'] memiliki reward -20  
sekuen ['BD', '1C', 'BD', '55'] memiliki reward 30  
30  
7A 55 BD 1C BD 55  
1, 1  
1, 3  
6, 3  
6, 5  
1, 5  
1, 2  
323 ms  
Apakah ingin menyimpan solusi? (y/n) : y  
Masukkan nama file untuk menyimpan output: jawaban2.txt  
Output telah disimpan ke jawaban2.txt
```

```
test2.txt  x  ...  jawaban2.txt  x  
test2.txt  
1 6  
2 6 7  
3 7A 55 E9 E9 1C 55  
4 55 7A 1C 7A E9 55  
5 55 1C 1C 55 E9 BD  
6 BD 1C 7A 1C 55 BD  
7 BD 55 BD 7A 1C 1C  
8 1C 55 55 7A 55 7A  
9 BD 55 BD 7A 1C 1C  
10 3  
11 BD E9 1C  
12 -10  
13 BD 7A BD  
14 -20  
15 BD 1C BD 55  
16 30  
jawaban2.txt  
1 30  
2 7A 55 BD 1C BD 55  
3 1, 1  
4 1, 3  
5 6, 3  
6 6, 5  
7 1, 5  
8 1, 2  
9 323 ms  
10
```

## Tangkapan Layar Input dan Output 3 dari file

```
WELCOME TO
CYBERPUNK 2077
BREACH PROTOCOL

1. Masukkan melalui file .txt
2. Masukkan melalui CLI
Pilih jenis masukkan (1/2): 1
Masukkan nama file: test3.txt
Ukuran Buffer: 7
Ukuran Kolom Matrix: 7
Ukuran Baris Matrix: 7
Matriks:
7A 55 E9 E9 1C 55 BD
55 7A 1C 7A E9 55 7A
55 1C 1C 55 E9 BD E9
BD 1C 7A 1C 55 BD 1C
BD 55 BD 7A 1C 1C 7A
1C 55 55 7A 55 7A 7A
BD 55 BD 7A 1C 1C 1C
Jumlah sekuens: 2
Sekuens:
sekuen ['7A', '1C', 'BD'] memiliki reward 20
sekuen ['1C', 'BD', '7A'] memiliki reward 30
50
7A 55 7A 1C BD 7A
1, 1
1, 2
2, 2
2, 3
6, 3
6, 6
2536 ms
Apakah ingin menyimpan solusi? (y/n) : y
Masukkan nama file untuk menyimpan output: jawaban3.txt
Output telah disimpan ke jawaban3.txt
```

test3.txt	jawaban3.txt
1 7	1 50
2 7 7	2 7A 55 7A 1C BD 7A
3 7A 55 E9 E9 1C 55 BD	3 1, 1
4 55 7A 1C 7A E9 55 7A	4 1, 2
5 55 1C 1C 55 E9 BD E9	5 2, 2
6 BD 1C 7A 1C 55 BD 1C	6 2, 3
7 BD 55 BD 7A 1C 1C 7A	7 6, 3
8 1C 55 55 7A 55 7A 7A	8 6, 6
9 BD 55 BD 7A 1C 1C 1C	9 2536 ms
10 2	10
11 7A 1C BD	
12 20	
13 1C BD 7A	
14 30	

## Tangkapan Layar Input dan Output 4 dari file

```
WELCOME TO  
CYBERPUNK 2077  
BREACH PROTOCOL  
  
1. Masukkan melalui file .txt  
2. Masukkan melalui CLI  
Pilih jenis masukkan (1/2): 2  
5  
BD 1C 7A 55 E9  
7  
6 6  
3  
4  
Ukuran Buffer: 7  
Ukuran Kolom Matrix: 6  
Ukuran Baris Matrix: 6  
Matriks:  
E9 55 1C BD BD 1C  
55 1C E9 7A 1C 55  
1C 7A BD BD 55 7A  
E9 7A BD 1C 7A 1C  
E9 BD BD BD 55 55  
7A 55 E9 1C E9 1C  
Jumlah sekuens: 3  
Sekuens:  
sekuen ['E9', '55', 'E9'] memiliki reward 60  
sekuen ['BD', 'E9', '55', 'E9'] memiliki reward 3  
sekuen ['1C', 'E9'] memiliki reward 62  
125  
55 BD E9 55 E9 1C E9  
2, 1  
2, 5  
1, 5  
1, 2  
3, 2  
3, 1  
1, 1  
1534 ms  
Apakah ingin menyimpan solusi? (y/n) : y  
Masukkan nama file untuk menyimpan output: jawaban4.txt  
Output telah disimpan ke jawaban4.txt
```

```
≡ jawaban4.txt ×  
  
≡ jawaban4.txt  
1 125  
2 55 BD E9 55 E9 1C E9  
3 2, 1  
4 2, 5  
5 1, 5  
6 1, 2  
7 3, 2  
8 3, 1  
9 1, 1  
10 1534 ms  
11
```

## Tangkapan Layar Input dan Output 5 dari file

```
WELCOME TO
CYBERPUNK 2077
BREACH PROTOCOL

1. Masukkan melalui file .txt
2. Masukkan melalui CLI
Pilih jenis masukkan (1/2): 2
3
BD 1C 7A
7
5 5
4
4
Ukuran Buffer: 7
Ukuran Kolom Matrix: 5
Ukuran Baris Matrix: 5
Matriks:
7A BD 7A 1C 1C
BD 1C 1C BD 7A
BD BD BD 7A 7A
1C BD BD 7A BD
1C BD 7A 1C 7A
Jumlah sekuens: 4
Sekuens:
sekuens ['7A', 'BD', '1C'] memiliki reward 75
sekuens ['BD', 'BD', 'BD', '7A'] memiliki reward 46
sekuens ['1C', '7A', '7A'] memiliki reward 81
sekuens ['1C', 'BD', '7A', 'BD'] memiliki reward 88
244
1C BD 7A BD 1C 7A 7A
4, 1
4, 2
5, 2
5, 4
1, 4
1, 1
3, 1
418 ms
Apakah ingin menyimpan solusi? (y/n) : y
Masukkan nama file untuk menyimpan output: jawaban5.txt
Output telah disimpan ke jawaban5.txt
```

```
≡ jawaban5.txt ×
≡ jawaban5.txt
1    244
2    1C BD 7A BD 1C 7A 7A
3    4, 1
4    4, 2
5    5, 2
6    5, 4
7    1, 4
8    1, 1
9    3, 1
10   418 ms
11   |
```

## Tangkapan Layar Input dan Output 6 dari file

```
WELCOME TO  
CYBERPUNK 2077  
BREACH PROTOCOL  
  
1. Masukkan melalui file .txt  
2. Masukkan melalui CLI  
Pilih jenis masukkan (1/2): 2  
4  
7A BD 1C E9  
5  
3 4  
3  
3  
Ukuran Buffer: 5  
Ukuran Kolom Matrix: 3  
Ukuran Baris Matrix: 4  
Matriks:  
E9 BD 1C  
E9 1C E9  
E9 BD 1C  
1C BD 7A  
Jumlah sekuens: 3  
Sekuens:  
sekuen ['E9', '7A'] memiliki reward 74  
sekuen ['7A', 'E9', 'BD'] memiliki reward 59  
sekuen ['1C', '1C', '1C'] memiliki reward 8  
74  
E9 E9 E9 7A  
1, 1  
1, 2  
3, 2  
3, 4  
3 ms  
Apakah ingin menyimpan solusi? (y/n) : y  
Masukkan nama file untuk menyimpan output: jawaban6.txt  
Output telah disimpan ke jawaban6.txt
```

```
≡ jawaban6.txt ×  
  
≡ jawaban6.txt  
  
1      74  
2      E9 E9 E9 7A  
3      1, 1  
4      1, 2  
5      3, 2  
6      3, 4  
7      3 ms  
8      
```



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		✓

link github : <https://github.com/BocilBlunder/Tucil1-13522113.git>