

LAPORAN TUGAS KECIL 1

Semester II tahun 2023/2024

Penyelesaian *Cyberpunk 2077 Breach Protocol* dengan Algoritma
Brute Force



NIM : 13522112

Nama : Dimas Bagoes Hendrianto

Kelas : K02

**PROGRAM STUDI TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
2024**

DAFTAR ISI

DAFTAR ISI.....	1
BAB I Deskripsi Masalah	2
BAB II Teori Singkat.....	4
BAB III Implementasi.....	5
BAB IV Eksperimen	8
Lampiran	11

BAB I

Dekripsi Masalah



Gambar 1.1 Permainan Breach Protocol

Cyberpunk 2077 Breach Protocol adalah minigame meretas pada permainan video Cyberpunk 2077. Minigame ini merupakan simulasi peretasan jaringan local dari ICE (Intrusion Countermeasures Electronics) pada permainan Cyberpunk 2077. Komponen pada permainan ini antara lain adalah:

1. Token – terdiri dari dua karakter alfanumerik seperti E9, BD, dan 55.
2. Matriks – terdiri atas token-token yang akan dipilih untuk menyusun urutan kode.
3. Sekuens – sebuah rangkaian token (dua atau lebih) yang harus dicocokkan.
4. Buffer – jumlah maksimal token yang dapat disusun secara sekuensial.

Aturan permainan Breach Protocol antara lain:

1. Pemain bergerak dengan pola horizontal, vertikal, horizontal, vertikal (bergantian) hingga semua sekuens berhasil dicocokkan atau buffer penuh.
2. Pemain memulai dengan memilih satu token pada posisi baris paling atas dari matriks.
3. Sekuens dicocokkan pada token-token yang berada di buffer.
4. Satu token pada buffer dapat digunakan pada lebih dari satu sekuens.
5. Setiap sekuens memiliki bobot hadiah atau reward yang variatif.
6. Sekuens memiliki panjang minimal berupa dua token.

IF2211 Strategi Algoritma
Tugas Kecil 1

Ilustrasi kasus :

Diberikan matriks sebagai berikut dan ukuran buffernya adalah tujuh

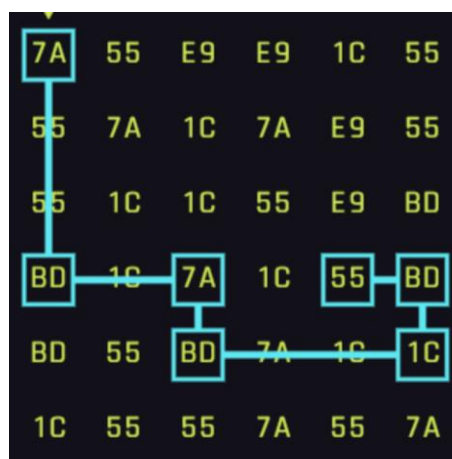
7A	55	E9	E9	1C	55
55	7A	1C	7A	E9	55
55	1C	1C	55	E9	BD
BD	1C	7A	1C	55	BD
BD	55	BD	7A	1C	1C
1C	55	55	7A	55	7A

Dengan sekuens sebagai berikut:

1. BD E9 1C dengan hadiah berbobot 15.
2. BD 7A BD dengan hadiah berbobot 20.
3. BD 1C BD 55 dengan hadiah berbobot 30.

Maka solusi yang optimal untuk matriks dan sekuens yang diberikan adalah sebagai berikut:

- Total bobot hadiah : 50 poin
- Total langkah : 6 langkah



Gambar 1.2 Contoh Solusi

BAB II

Teori Singkat

Algoritma Brute Force

Brute force adalah sebuah pendekatan yang lempang (straightforward) untuk memecahkan suatu masalah, biasanya didasarkan pada pernyataan masalah (problem statement) dan definisi konsep yang dilibatkan. Algoritma brute force memecahkan masalah dengan sangat sederhana, langsung dan dengan cara yang jelas (obvious way).

Algoritma brute force umumnya tidak “cerdas” dan tidak mangkus, karena ia membutuhkan jumlah langkah yang besar dalam penyelesaiannya. Kadang-kadang algoritma brute force disebut juga algoritma naif (naïve algorithm). Algoritma Algoritma brute force seringkali merupakan merupakan pilihan pilihan yang kurang disukai karena ketidakmangkusannya itu, tetapi dengan mencari pola-pola yang mendasar, keteraturan, atau trik-trik khusus, biasanya akan membantu kita menemukan algoritma yang lebih cerdas dan lebih mangkus. Untuk masalah yang ukurannya kecil, kesederhanaan brute force biasanya lebih diperhitungkan daripada ketidakmangkusannya. Algoritma brute force sering digunakan sebagai basis bila membandingkan beberapa alternatif algoritma yang mangkus.

BAB III

Implementasi

```
def find(nbuff, mWidth, mHeight, matrix, nSeq, arrSeq, arrPoint):
    start = time.time()

    buff = nbuff

    coorAll = []

    coorRoot = []

    for i in range (0, mWidth):
        coorRoot += [[[0, i]]]

    coorAll += coorRoot

    for i in range (1, buff):
        coorNew = []
        if i % 2 == 1:
            for coor in coorRoot :
                path = []
                path += coor
                abs, oor = coor[len(coor)-1][0], coor[len(coor)-1][1]
                for j in range (0, mHeight):
                    path += ([[j, oor]])
                    coorNew.append(path)
                    path = path[:-1]

            if i % 2 == 0:
                for coor in coorRoot :
                    path = []
                    path += coor
                    abs, oor = coor[len(coor)-1][0], coor[len(coor)-1][1]
                    for j in range (0, mWidth):
                        path += ([[abs, j]])
                        coorNew.append(path)
                        path = path[:-1]

        coorAll += coorNew
        coorRoot = coorNew

    remove = []

    toRemove = 0

    for seq in coorAll:
```

IF2211 Strategi Algoritma

Tugas Kecil 1

```
total = 0
for coor in seq:
    for i in range (0, len(seq)):
        if coor == seq[i]:
            total += 1
if total > len(seq):
    remove.append(toRemove)
toRemove += 1

for j in range (len(remove)-1, -1, -1):
    coorAll.pop(remove[j])

pointAll = [0] * len(coorAll)

for i in range (0, len(coorAll)):
    for j in range (0, nSeq):
        if (len(coorAll[i]) >= len(arrSeq[j])):
            for k in range (0, len(coorAll[i]) - len(arrSeq[j]) + 1):
                status = 0
                for l in range (0, len(arrSeq[j])):
                    if (matrix[coorAll[i][k+l][0]][coorAll[i][k+l][1]] ==
arrSeq[j][l]):
                        status += 1
                if (status == len(arrSeq[j])):
                    pointAll[i] += arrPoint[j]

max = 0
loc = 0
tokenRes = []
coorRes = []

for i in range (0, len(pointAll)):
    if pointAll[i] > max:
        max = pointAll[i]
        loc = i

step = len(coorAll[loc])

for i in range (0, step):
    tokenRes.append(matrix[coorAll[loc][i][0]][coorAll[loc][i][1]])
    coorRes.append(coorAll[loc][i])
    coorRes[i][0], coorRes[i][1] = coorRes[i][1]+1, coorRes[i][0]+1

end = time.time()
elapsedTime = end - start

return max, tokenRes, coorRes, elapsedTime
```

IF2211 Strategi Algoritma
Tugas Kecil 1

Algoritma Brute Force yang saya gunakan :

1. Mencari semua kemungkinan “jalur” yang bisa dilewati dengan aturan vertikal, horizontal bergantian mulai dari 1 token, 2 token, 3 token, ... sampai buffer token
2. Menghapus “jalur” yang memakai token pada tempat yang sama lebih dari sekali
3. Menghitung poin dari setiap kemungkinan jalur yang valid sesuai sekuen yang ada dan mencari yang terbesar serta yang paling optimal dengan definisi jumlah token yang seminimal mungkin diantaranya.
4. Mengembalikan nilai maksimal, urutan token, urutan koordinat dalam matriks, serta waktu yang digunakan

BAB IV

Eksperimen

```
1.txt M X readCom.py > < ... powershell + - < > x
test> 1.txt
1 7
2 6 6
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 3
10 BD E9 1C
11 15
12 BD 7A BD
13 20
14 BD 1C BD 55
15 30

PS C:\Users\Dimas\Documents\GitHub\Tucill_13522112\src> python main.py
-- BREACH PROTOCOL --
Select input (1/2) =
1. File input
2. Manual input
=> 1
File = 1.txt

50
7A BD 7A BD 1C BD 55

1, 1
1, 4
3, 4
3, 5
6, 5
6, 3
1, 3

22.76264524459839s

Apakah ingin menyimpan solusi? (y/n)
y
File = 1ans.txt

Answer saved
PS C:\Users\Dimas\Documents\GitHub\Tucill_13522112\src> >
```

Gambar 4.1 Testcase 1

```
PS C:\Users\Dimas\Documents\GitHub\Tucill1_13522112\src> python main.py
-- BREACH PROTOCOL --
Select input (1/2) =
1. File input
2. Manual input
==> 2
5
BD 1C 7A 55 E9
7
6 6
3
4

MATRIX
55 E9 1C 1C E9 1C

1C E9 55 BD 55 1C

E9 55 7A 7A BD

55 E9 55 55 BD E9

BD BD 1C 55 1C 1C

7A 1C E9 E9 55 BD

SEKUEN
BD 1C
20
1C BD
25
55 E9 1C BD
5

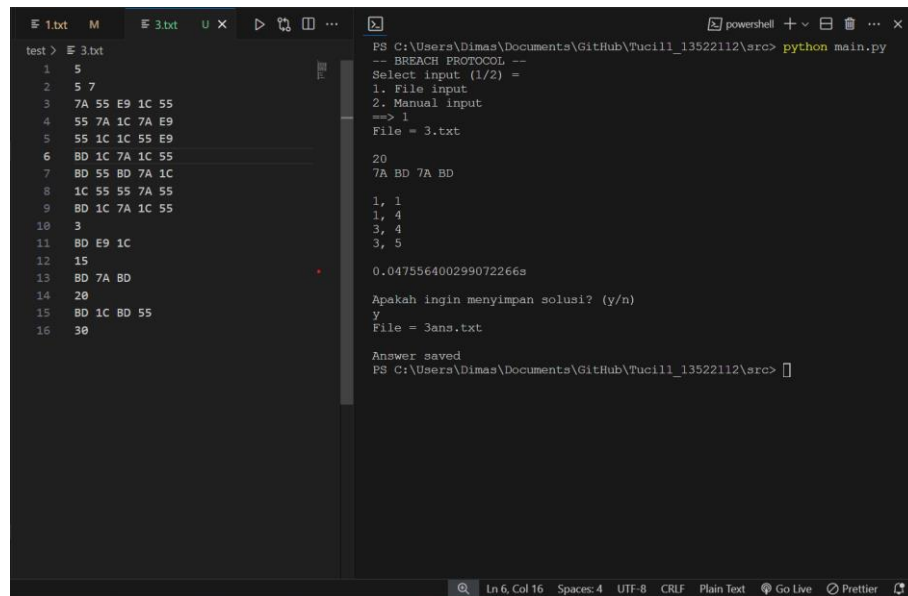
135
1C BD 1C BD 1C BD 1C

4, 1
4, 2
1, 2
1, 5
6, 5
6, 6
2, 6
```

Gambar 4.2 Testcase 2

IF2211 Strategi Algoritma

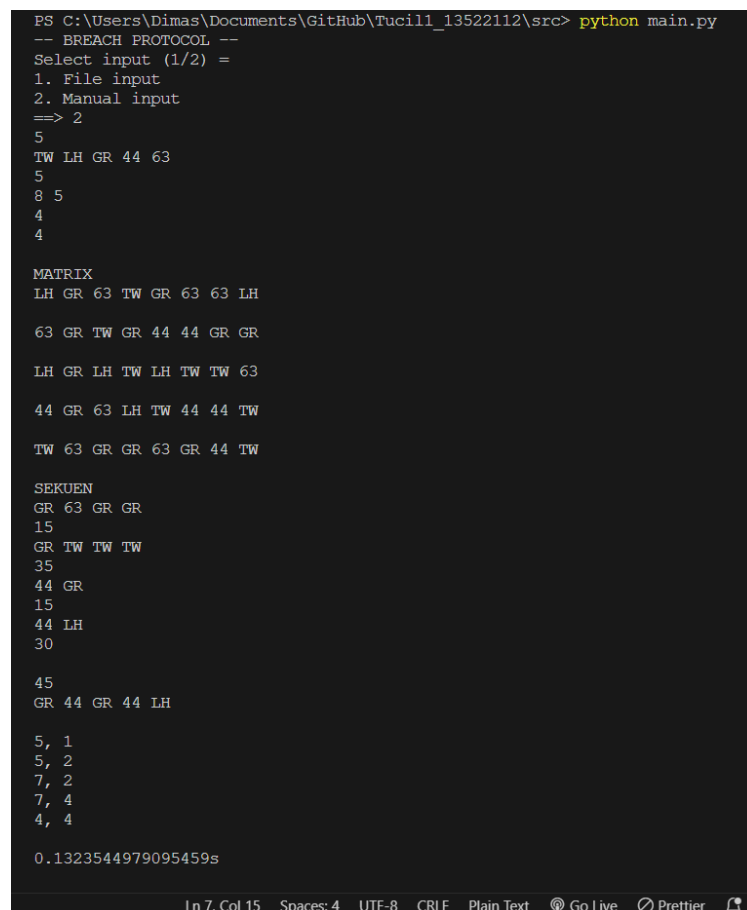
Tugas Kecil 1



```
test > 1.txt M 3.txt U X
1 5
2 5 7
3 7A 55 E9 1C 55
4 55 7A 1C 7A E9
5 55 1C 1C 55 E9
6 BD 1C 7A 1C 55
7 BD 55 BD 7A 1C
8 1C 55 55 7A 55
9 BD 1C 7A 1C 55
10 3
11 BD E9 1C
12 15
13 BD 7A BD
14 20
15 BD 1C BD 55
16 30

PS C:\Users\Dimas\Documents\GitHub\Tucil1_13522112\src> python main.py
-- BREACH PROTOCOL --
Select input (1/2) =
1. File input
2. Manual input
==> 1
File = 3.txt
20
7A BD 7A BD
1, 1
1, 4
3, 4
3, 5
0.047556400299072266s
Apakah ingin menyimpan solusi? (y/n)
y
File = 3ans.txt
Answer saved
PS C:\Users\Dimas\Documents\GitHub\Tucil1_13522112\src>
```

Gambar 4.3 Testcase 3

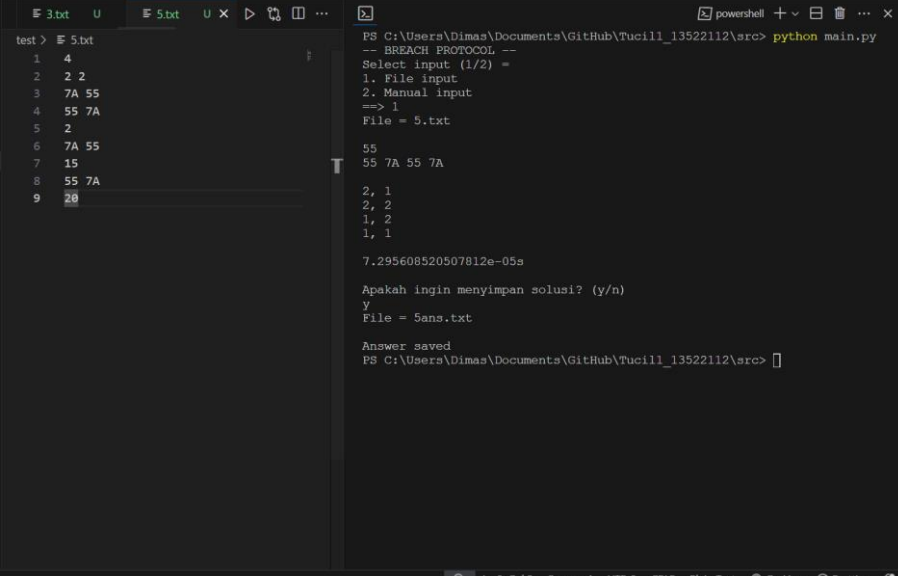


```
PS C:\Users\Dimas\Documents\GitHub\Tucil1_13522112\src> python main.py
-- BREACH PROTOCOL --
Select input (1/2) =
1. File input
2. Manual input
==> 2
5
TW LH GR 44 63
5
8 5
4
4
MATRIX
LH GR 63 TW GR 63 63 LH
63 GR TW GR 44 44 GR GR
LH GR LH TW LH TW TW 63
44 GR 63 LH TW 44 44 TW
TW 63 GR GR 63 GR 44 TW
SEKUEN
GR 63 GR GR
15
GR TW TW TW
35
44 GR
15
44 LH
30
45
GR 44 GR 44 LH
5, 1
5, 2
7, 2
7, 4
4, 4
0.1323544979095459s
```

Gambar 4.4 Testcase 4

IF2211 Strategi Algoritma

Tugas Kecil 1



```
test > 5.txt
1 4
2 2 2
3 7A 55
4 55 7A
5 2
6 7A 55
7 15
8 55 7A
9 20

PS C:\Users\Dimas\Documents\GitHub\Tucil1_13522112\src> python main.py
-- BREACH PROTOCOL --
Select input (1/2) =
1. File input
2. Manual input
==> 1
File = 5.txt

55
55 7A 55 7A

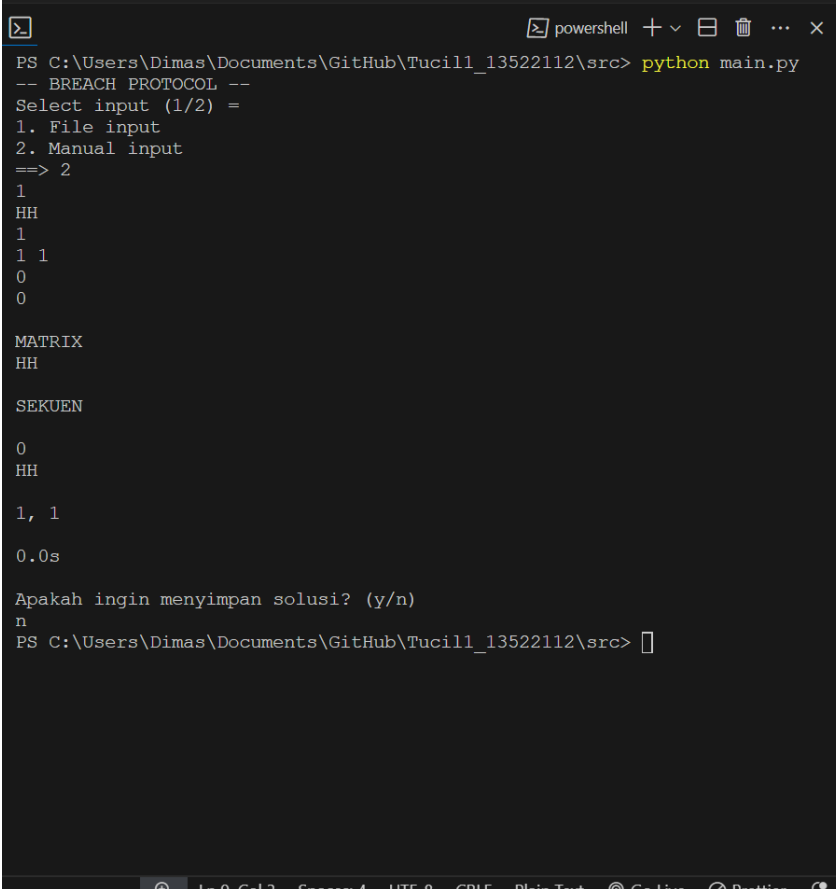
2, 1
2, 2
1, 2
1, 1

7.295608520507812e-05s

Apakah ingin menyimpan solusi? (y/n)
y
File = 5ans.txt

Answer saved
PS C:\Users\Dimas\Documents\GitHub\Tucil1_13522112\src>
```

Gambar 4.5 Testcase 5



```
PS C:\Users\Dimas\Documents\GitHub\Tucil1_13522112\src> python main.py
-- BREACH PROTOCOL --
Select input (1/2) =
1. File input
2. Manual input
==> 2
1
HH
1
1 1
0
0
0

MATRIX
HH

SEKUEN

0
HH

1, 1

0.0s

Apakah ingin menyimpan solusi? (y/n)
n
PS C:\Users\Dimas\Documents\GitHub\Tucil1_13522112\src>
```

Gambar 4.6 Testcase 6

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		✓

Link

https://github.com/dimasb1954/Tucil1_13522112.git