

**Laporan Tugas Kecil 1 IF2211 Strategi Algoritma
Penyelesaian Cyberpunk 2077 Breach Protocol Dengan
Algoritma Brute Force**



Disusun oleh:

Keanu Amadius Gonza Wrahatno - 13522082

**PROGRAM STUDI TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
2023/2024**

DAFTAR ISI

DAFTAR ISI	1
BAB I Deskripsi	2
BAB II Algoritma Brute Force	3
BAB III Source Code.....	6
BAB IV Test Case	10
BAB V Git Hub.....	16
BAB VI Tabel poin	17

BAB I

Deskripsi

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.

Penulis menggunakan bahasa python untuk mencari solusi dari permainan Breach Protocol pada Cyberpunk 2077 menggunakan algoritma Brute Force.

BAB II

Algoritma Brute Force

Algoritma yang digunakan penulis untuk menyelesaikan masalah ini adalah brute force dengan hasil yang optimal (**path token terpendek dari reward maksimal**) sehingga merelakan waktu lebih lama sedikit. Algoritma Brute Force adalah algoritma yang melakukan pencocokan “karakter” dari sebelah kiri ke sebelah kanan dan jika antara pattern dan sequence terdapat kecocokan, maka algoritma akan menghasilkan nilai true. Untuk kasus ini, yang di outputkan tidak hanya true false melainkan bobot maksimal sequence, sequence, koordinat sequence pada matrix, dan waktu. Berikut ini langkah-langkah brute force yang penulis buat atau terapkan:

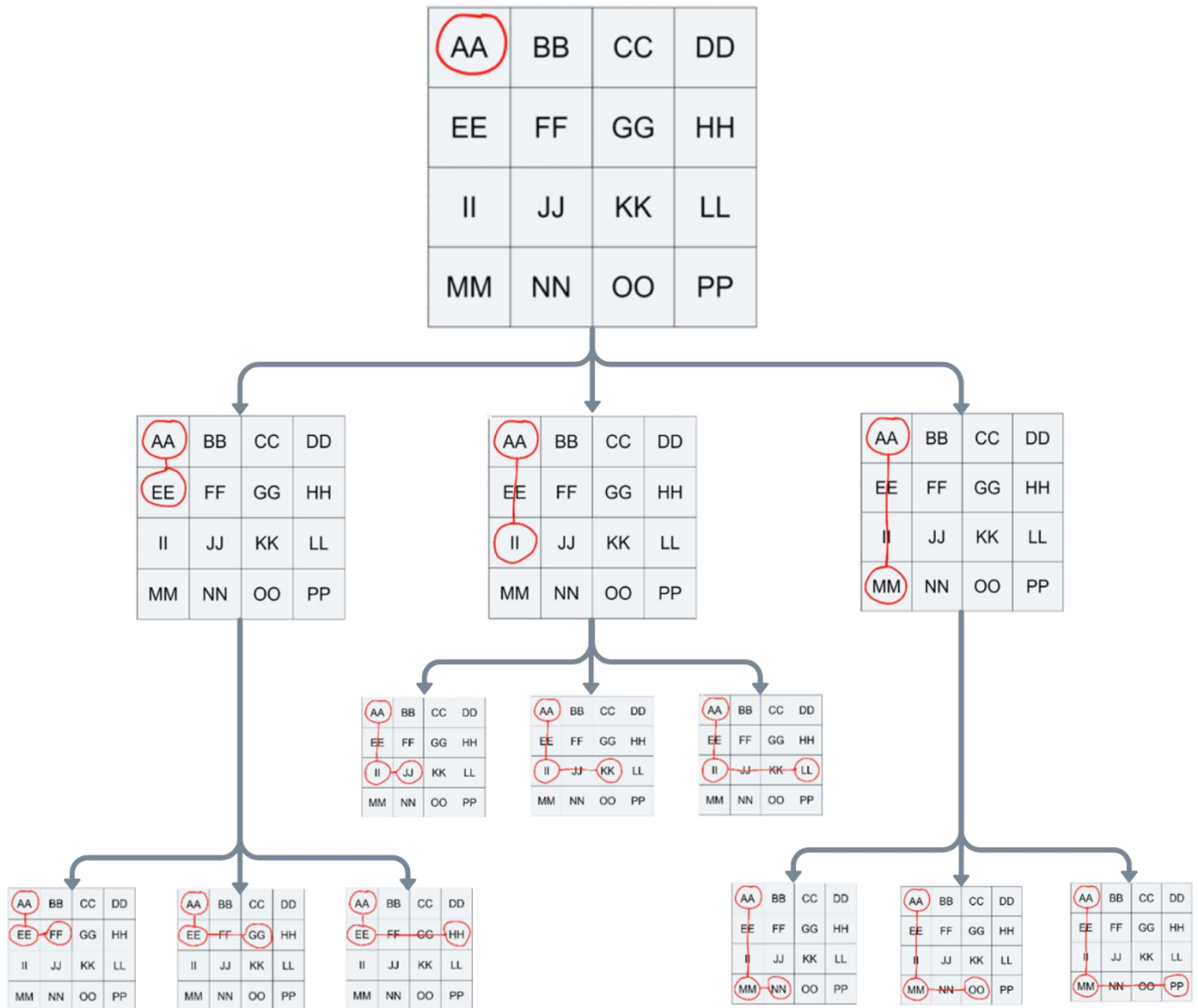
1. Membuat fungsi rekursif bruteForce dimana fungsi ini akan mengenerate seluruh path hanya sepanjang buffer.
2. Misal buffer max yaitu n, maka program akan memanggil fungsi bruteForce sebanyak n kali untuk tiap tiap panjang path dari 1 – n.
3. Terdapat kondisi false dan true secara bergantian saat rekursif dimana saat true melakukan pembacaan secara vertikal, sedangkan saat false melakukan pembacaan secara horizontal.
4. Program diimplementasikan secara rekursif dan memulai pembacaan baris pertama yaitu dari koordinat (a,1) dengan a dari 1 sampai jumlah kolom. Lalu memulai pembacaan token kedua yaitu untuk setiap a, dilakukan pembacaan (a,b) dengan b dari 1 sampai jumlah baris. Dilakukan secara rekursif sampai jumlah token mencapai buffer dengan syarat cell belum pernah dikunjungi sebelumnya.

(1,1)	(2,1)	(3,1)	(4,1)
(1,2)	(2,2)	(3,2)	(4,2)
(1,3)	(2,3)	(3,3)	(4,3)
(1,4)	(2,4)	(3,4)	(4,4)

Gambar 2.1. Matrix 4 x 4

(Sumber: Arsip Pribadi)

5. Misal mencari path dengan buffer 3 pada matrix 4x4, maka program akan berjalan seperti ini:



Gambar 2.2. Proses Pembentukan Pola

(Sumber: Arsip Pribadi)

Maka akan terbentuk pola [AA,EE,FF] , [AA,EE,GG] , [AA,EE,HH] , [AA,II,JJ] , [AA,II,KK] , [AA,II,LL] , [AA,MM,NN] , [AA,MM,OO] , [AA,MM,PP]

6. Program masih akan terus rekrusif dengan memulai dari koordinat (2,1) yaitu cell BB sampai koordinat paling kanan atas yaitu (4,1) yaitu cell DD. Tentunya dengan syarat cell belum pernah dikunjungi sebelumnya.
7. Path token terakhir yang didapatkan dari buffer sebesar 3 adalah [DD,PP,OO]

AA	BB	CC	DD
EE	FF	GG	HH
II	JJ	KK	LL
MM	NN	OO	PP

8. Lalu hasil dari pola token ini disimpan dalam array.
9. Kemudian dilakukan pengecekan apakah ada sequence pada tiap tiap substring dari semua pola path dalam array.
10. Lalu dilakukan pencatatan pola token yang memiliki reward maximum sekaligus menyimpan pola token dan koordinat.

BAB III

Source Code

Struktur folder yang penulis buat yaitu:

```
Tucil1_13522082
├── bin
├── doc
│   └── Tucil1_13522082_Keanu Amadius Gonza Wrahatno.pdf
├── src
│   ├── bruteForce.py
│   ├── main.py
│   └── readFile.py
├── test
│   ├── solusi.txt
│   └── test.txt
└── README.md
```

bruteForce.py

```
1 import numpy as np
2 import time
3 import os
4
5 def bruteForce (row, col, depth, condition, hasil, koordinat, mJejak, MATRIX, M_ROW, M_COL, BUFFER, ALL_PATH, ALL_KOORDINAT):
6     if (BUFFER < depth): #depth melebihi buffer
7         return
8
9     elif (depth == BUFFER): #simpan jawaban di ALL_PATH dan ALL_KOORDINAT
10         ALL_PATH.append(hasil)
11         ALL_KOORDINAT.append(koordinat)
12         return
13
14     if condition: #proses vertikal
15         for j in range(M_COL):
16             curJejak = np.copy(mJejak)
17             if (curJejak[row][j] == 0):
18                 curJejak[row][j] = 1
19                 curHasil = hasil + MATRIX[row][j] + " "
20                 bruteForce(row, j, depth+1, False, curHasil, [*koordinat, (j+1, row+1)], curJejak, MATRIX, M_ROW, M_COL, BUFFER, ALL_PATH, ALL_KOORDINAT)
21     else: #proses horizontal
22         for i in range(M_ROW):
23             curJejak = np.copy(mJejak)
24             if (curJejak[i][col] == 0):
25                 curJejak[i][col] = 1
26                 curHasil = hasil + MATRIX[i][col] + " "
27                 bruteForce(i, col, depth+1, True, curHasil, [*koordinat, (col+1, i+1)], curJejak, MATRIX, M_ROW, M_COL, BUFFER, ALL_PATH, ALL_KOORDINAT)
```

Tugas Kecil 1 IF2211 Strategi Algoritma
Penyelesaian Cyberpunk 2077 Breach Protocol Dengan Algoritma Brute Force

```
def matchedSequence(ALL_PATH, sequence, bobot_sequence, ALL_KOORDINAT, hasilKoordinat, hasilBobot, hasilPath): #cek apakah ada sequence di dalam semua path
    bobotMax = 0
    cocok = False
    j = -1
    for path in ALL_PATH:
        j += 1
        bobotNow = 0
        for i in range(len(sequence)):

            if sequence[i] in path:
                bobotNow = bobotNow + bobot_sequence[i]
                cocok = True

        if bobotMax < bobotNow and cocok:
            path_final = path
            bobotMax = bobotNow
            koordinat = ALL_KOORDINAT[j]

    hasilBobot.append(bobotMax)
    if bobotMax != 0:
        hasilKoordinat.append(koordinat)
        hasilPath.append(path_final)

def solusi(mJejak, MATRIX, M_ROW, M_COL, BUFFER, sequence, bobot_sequence):
    start = time.time()
    ALL_PATH = []
    ALL_KOORDINAT = []
    PATH = []
    KOORDINAT = []
    for i in range(2, BUFFER+1): #mencari semua path dengan panjang sequence 1 sampai max buffer
        bruteForce(0, 0, 0, True, "", [], mJejak, MATRIX, M_ROW, M_COL, i, ALL_PATH, ALL_KOORDINAT)
        PATH += ALL_PATH
        KOORDINAT += ALL_KOORDINAT
    koordinatHasil = []
    bobotMax = []
    hasilPath = []
    matchedSequence(ALL_PATH, sequence, bobot_sequence, ALL_KOORDINAT, koordinatHasil, bobotMax, hasilPath)
    end = time.time()

    #tampilkan hasil
    for bobot in bobotMax:
        print(f"Reward maksimal : {bobot}")
    if bobot != 0:
        print(f"Path token      : {hasilPath[0]}")
        print("Koordinat      : ")
        for koordinat in koordinatHasil:
            for koordinat2 in koordinat:
                print(koordinat2)

    waktu_ms = "{:.3f}".format((end - start) * 1000)
    waktu_detik = "{:.3f}".format(end - start)
    print(f'\nWaktu dibutuhkan: {waktu_ms} ms / {waktu_detik} detik.')

    #save ke file .txt
    save = input('\nApakah anda ingin menyimpan jawaban? (y/n)\n')
    while save.lower() not in ['y', 'n']:
        save = input('Masukkan pilihan dengan benar (y/n): ')

    if (save == 'y' or save == 'Y'):
        namaSave = input('\nMasukan / buat nama file untuk menyimpan: ')
        currentDir = os.path.dirname(os.path.realpath(__file__))
        with open(os.path.join(currentDir, "..", "test", f"{namaSave}.txt"), 'w') as f:
            f.write(f"Reward maksimal : {bobot}\n")
            if bobot != 0:
                f.write(f"Path token      : {hasilPath[0]}\n")
                f.write("Koordinat      :\n")
                for koordinat in koordinatHasil:
                    for koordinat2 in koordinat:
                        f.write(str(koordinat2) + '\n')
                f.write(f'\nWaktu dibutuhkan: {waktu_ms} ms / {waktu_detik} detik.')
            print(f'\nTelah tersimpan dengan nama file "{namaSave}.txt" pada folder "test"')
    else:
        print('\nJawaban tidak tersimpan!!!')
```


readFile.py

```
import os
import numpy as np

class FileReader:
    def __init__(self):
        self.matrix_size = []
        self.matrix = []
        self.sequence_size = 0
        self.sequence = []
        self.bobot_sequence = []
        self.buffer = 0
        self.condition = True

    def getDirectory(self):
        # Mengembalikan directory saat ini
        return os.path.dirname(os.path.realpath(__file__))

    def read_file(self, namaFile):
        currentDir = self.getDirectory()
        try:
            with open(os.path.join(currentDir, "..", "test", namaFile), 'r') as f:
                self.condition = True
                content = f.read()
                content = content.split('\n')
                content = [x for x in content if x.strip()]

                self.buffer = int(content[0])

                #membuat matrix size
                matrix_size = content[1].split(' ')
                self.matrix_size.append(int(matrix_size[0]))
                self.matrix_size.append(int(matrix_size[1]))

                #membuat matrix
                for i in range(int(matrix_size[1])):
                    matrix = []
                    temp = content[2+i].split(' ')
                    for j in range(int(matrix_size[0])):
                        matrix.append(temp[j])
                    self.matrix.append(matrix)

                #membuat sequence_size
                sequence_size = content[2+int(matrix_size[1])]
                self.sequence_size = sequence_size

                #membuat sequence dan bobot_sequence
                for i in range(0, int(sequence_size)*2, 2):
                    self.sequence.append(content[3+int(matrix_size[1])+i])
                    self.bobot_sequence.append(int(content[4+int(matrix_size[1])+i]))
        except FileNotFoundError:
            self.condition = False
            print(f"\nFile {namaFile} tidak ditemukan (pastikan ada .txt).")

    def auto(self):
        print("\n====MODE AUTO====")
        print("Silahkan isi data\n")
        token = []
        matrix_size = [[], []]
        jumlah_token = input("Jumlah token unik:\n")
        token = input(f"Masukkan token sebanyak {jumlah_token}: Contoh (BD 1C 7A 55 E9)\n")
        token = token.split(' ')
        self.buffer = int(input('Ukuran Buffer: \n'))
        matrix_size = input('Ukuran Matrix: Contoh (6 6) \n')
        matrix_size = matrix_size.split(' ')
        matrix_size = list(map(int, matrix_size))
        sequence_size = int(input('Jumlah Sequence: \n'))
        ukuran_maks_sequences = int(input('Ukuran Maksimal Sequence: \n'))
        self.matrix_size = matrix_size
        self.sequence_size = sequence_size

        self.matrix = np.random.choice(token, size=(matrix_size[1], matrix_size[0]))
        for i in range(sequence_size):
            cur_sequence = ' '.join(np.random.choice(token, size= np.random.randint(1, ukuran_maks_sequences)))
            while cur_sequence in self.sequence:
                cur_sequence = ' '.join(np.random.choice(token, size= np.random.randint(1, ukuran_maks_sequences)))
            self.sequence.append(cur_sequence)
        self.bobot_sequence = np.random.randint(1, 50, size=sequence_size)
```

main.py

```
1 import readFile
2 import bruteforce
3 import numpy as np
4 import time
5
6 print('Apakah anda akan menggunakan file atau auto?')
7 print('1. File')
8 print('2. Auto')
9
10
11 pilihan = int(input('Pilihan: '))
12 while pilihan <1 or pilihan>2:
13     pilihan = int(input('Masukan pilihan dengan benar: '))
14
15 if pilihan == 1:
16     namaFile = input('\nPastikan file ada dalam folder "test".\nContoh nama file (soal.txt).\nMasukan nama file anda: ')
17     reader = readFile.FileReader()
18     reader.read_file(namaFile)
19     while not (reader.condition):
20         namaFile= input('Masukan nama file dengan benar: ')
21         reader.read_file(namaFile)
22     print('\n\n=====FILE DITEMUKAN=====')
23     mJepak = np.zeros((reader.matrix_size[1],reader.matrix_size[0]))
24     bruteforce.solusi(mJepak, reader.matrix, reader.matrix_size[1], reader.matrix_size[0], reader.buffer, reader.sequence, reader.bobot_sequence)
25
26
27 elif pilihan == 2:
28     reader = readFile.FileReader()
29     reader.auto()
30     print("\n=====")
31     print('\nMatrix: ')
32     for i in range(len(reader.matrix)):
33         print(reader.matrix[i])
34     print('\nSequences (unik): ')
35     for i in range(len(reader.sequence)):
36         print(str(reader.sequence[i]) + '      (bobotnya: ' + str(reader.bobot_sequence[i]) + ')')
37     print("\n=====")
38     mJepak = np.zeros((reader.matrix_size[1],reader.matrix_size[0]))
39     bruteforce.solusi(mJepak, reader.matrix, reader.matrix_size[1], reader.matrix_size[0], reader.buffer, reader.sequence, reader.bobot_sequence)
```

BAB IV

Test Case

4.1 Test Case 1

Input (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
```

```
Apakah anda akan menggunakan file atau auto?
1. File
2. Auto
Pilihan: 1

Pastikan file ada dalam folder "test".
Contoh nama file (soal.txt).
Masukan nama file anda: test_1.txt

=====FILE DITEMUKAN=====

Reward maksimal : 50
Path token      : 7A BD 7A BD 1C BD 55
Koordinat       :
(1, 1)
(1, 4)
(3, 4)
(3, 5)
(6, 5)
(6, 3)
(1, 3)

Waktu dibutuhkan: 273.420 ms / 0.273 detik.

Apakah anda ingin menyimpan jawaban? (y/n)
y

Masukan / buat nama file untuk menyimpan: solusi_1

Telah tersimpan dengan nama file "solusi_1.txt" pada folder "test"
PS C:\Users\Keanu\Documents\GitHub\Tucil-1\src>
```

Result (disimpan di solusi_1.txt)

```
1 Reward maksimal : 50
2 Path token      : 7A BD 7A BD 1C BD 55
3 Koordinat       :
4 (1, 1)
5 (1, 4)
6 (3, 4)
7 (3, 5)
8 (6, 5)
9 (6, 3)
10 (1, 3)
11
12 Waktu dibutuhkan: 273.420 ms / 0.273 detik.
```

4.2 Test Case 2

Input (test_2.txt)

Test jika input salah

```
1 7
2 6 9
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 7A 55 E9 E9 1C 55
9 55 7A 1C 7A E9 55
10 55 1C 1C 55 E9 BD
11 1C 55 55 7A 55 7A
12 3
13 BD E9 1C
14 -15
15 BD 7A BD
16 -20
17 BD 1C BD 55
18 -30
```

```
Apakah anda akan menggunakan file atau auto?
1. File
2. Auto
Pilihan: 3
Masukan pilihan dengan benar: 1

Pastikan file ada dalam folder "test".
Contoh nama file (soal.txt).
Masukan nama file anda: test

File test tidak ditemukan (pastikan ada .txt).
Masukan nama file dengan benar: test_2.txt

=====FILE DITEMUKAN=====

Reward maksimal : 0

Waktu dibutuhkan: 1286.887 ms / 1.287 detik.

Apakah anda ingin menyimpan jawaban? (y/n)
1
Masukkan pilihan dengan benar (y/n): y

Masukan / buat nama file untuk menyimpan: solusi_3

Telah tersimpan dengan nama file "solusi_3.txt" pada folder "test"
```

Result (disimpan di solusi_2.txt)

```
1 Reward maksimal : 0
2
3 Waktu dibutuhkan: 1286.887 ms / 1.287 detik.
```

4.3 Test Case 3

Input (test_9.txt)

```
1 7
2 1 5
3 AA
4 BB
5 CC
6 DD
7 EE
8 3
9 AA CC
10 15
11 AA BB
12 20
13 CC DD
14 30
```

```
Apakah anda akan menggunakan file atau auto?
1. File
2. Auto
Pilihan: 1

Pastikan file ada dalam folder "test".
Contoh nama file (soal.txt).
Masukan nama file anda: test_9.txt

=====FILE DITEMUKAN=====

Reward maksimal : 20
Path token      : AA BB
Koordinat       :
(1, 1)
(1, 2)

Waktu dibutuhkan: 0.000 ms / 0.000 detik.

Apakah anda ingin menyimpan jawaban? (y/n)
y

Masukan / buat nama file untuk menyimpan: solusi_9

Telah tersimpan dengan nama file "solusi_9.txt" pada folder "test"
```

Result (disimpan di solusi_9.txt)

```
1 Reward maksimal : 20
2 Path token      : AA BB
3 Koordinat       :
4 (1, 1)
5 (1, 2)
6
7 Waktu dibutuhkan: 0.000 ms / 0.000 detik.
```

4.4 Test Case 4

Input Auto

```
Apakah anda akan menggunakan file atau auto?
1. File
2. Auto
Pilihan: 2

====MODE AUTO====
Silahkan isi data

Jumlah token unik:
5
masukan token sebanyak 5: Contoh (BD 1C 7A 55 E9)
BD 1C 7A 55 E9
Ukuran Buffer:
7
Ukuran Matrix: Contoh (6 6)
6 6
Jumlah Sequence:
3
Ukuran Maksimal Sequence:
4
```

Result

```
=====
Matrix:
['55' '55' '7A' '55' '55' 'BD']
['1C' 'E9' 'E9' 'BD' '55' '55']
['7A' '1C' '1C' 'BD' '1C' 'BD']
['BD' 'E9' '7A' 'E9' '7A' '55']
['7A' 'E9' '7A' '55' '7A' '55']
['55' '7A' 'E9' '1C' '1C' 'BD']

Sequences (unik):
55 1C BD 1C      (bobotnya:20)
E9 E9 BD        (bobotnya:45)
7A              (bobotnya:20)

=====

Reward maksimal : 65
Path token      : 55 7A E9 E9 BD
Koordinat      :
(1, 1)
(1, 5)
(2, 5)
(2, 2)
(4, 2)

Waktu dibutuhkan: 334.304 ms / 0.334 detik.

Apakah anda ingin menyimpan jawaban? (y/n)
y

Masukan / buat nama file untuk menyimpan: solusiAuto_1
Telah tersimpan dengan nama file "solusiAuto_1.txt" pada folder "test"
```

(disimpan di solusiAuto_1.txt)

```
test > solusiAuto_1.txt
1  Reward maksimal : 65
2  Path token      : 55 7A E9 E9 BD
3  Koordinat      :
4  (1, 1)
5  (1, 5)
6  (2, 5)
7  (2, 2)
8  (4, 2)
9
10 Waktu dibutuhkan: 334.304 ms / 0.334 detik.
```

4.5 Test Case 5

Input Auto

```
Apakah anda akan menggunakan file atau auto?
1. File
2. Auto
Pilihan: 2

===MODE AUTO===
Silahkan isi data

Jumlah token unik:
5
masukan token sebanyak 5: Contoh (BD 1C 7A 55 E9)
BD 1C 7A 55 E9
Ukuran Buffer:
5
Ukuran Matrix: Contoh (6 6)
5 10
Jumlah Sequence:
20
Ukuran Maksimal Sequence:
10

=====
```

Result

```
=====
Matrix:
['BD' '1C' '55' '1C' '55']
['1C' '1C' '1C' '1C' '7A']
['1C' '55' '1C' '55' '7A']
['7A' '7A' '1C' 'E9' 'E9']
['BD' '55' '1C' 'E9' '7A']
['55' 'E9' 'E9' '55' '55']
['1C' '7A' '55' '7A' '55']
['7A' 'E9' 'BD' 'E9' '55']
['1C' '7A' 'BD' 'E9' 'E9']
['1C' '55' '1C' '1C' 'BD']

Sequences (unik):
55 55 1C BD 55 BD BD BD (bobotnya:25)
1C 1C (bobotnya:45)
55 7A (bobotnya:3)
E9 E9 E9 (bobotnya:44)
7A 1C 1C 55 BD 7A 7A 1C 1C 55 (bobotnya:10)
7A 7A 7A (bobotnya:32)
BD E9 7A 55 7A 1C E9 55 BD BD (bobotnya:34)
1C BD BD (bobotnya:49)
55 E9 BD E9 7A BD 55 BD BD 55 (bobotnya:37)
BD 1C E9 (bobotnya:46)
55 1C 7A (bobotnya:22)
1C 1C BD E9 7A E9 BD BD BD (bobotnya:35)
E9 E9 E9 7A 7A E9 (bobotnya:34)
7A (bobotnya:26)
E9 E9 BD 1C E9 1C 1C 1C (bobotnya:49)
E9 1C BD BD (bobotnya:8)
E9 E9 1C 1C BD (bobotnya:10)
55 55 1C E9 E9 7A 55 55 55 (bobotnya:44)
BD 7A 55 BD BD E9 BD 55 E9 (bobotnya:1)
1C (bobotnya:45)

=====

Reward maksimal : 138
Path token      : 55 1C 7A 1C 1C
Koordinat       :
(3, 1)
(3, 4)
(1, 4)
(1, 2)
(2, 2)

Waktu dibutuhkan: 30.621 ms / 0.031 detik.

Apakah anda ingin menyimpan jawaban? (y/n)
Y

Masukan / buat nama file untuk menyimpan: solusiAuto_2

Telah tersimpan dengan nama file "solusiAuto_2.txt" pada folder "test"
PS C:\Users\Keanu\Documents\GitHub\Tucil-1>
```

Disimpan di file (solusiAuto_2.txt)

```
test > solusiAuto_2.txt

1 Reward maksimal : 138
2 Path token      : 55 1C 7A 1C 1C
3 Koordinat       :
4 (3, 1)
5 (3, 4)
6 (1, 4)
7 (1, 2)
8 (2, 2)
9
10 Waktu dibutuhkan: 30.621 ms / 0.031 detik.
```

4.6 Test Case 6

Input Auto

```
Apakah anda akan menggunakan file atau auto?
1. File
2. Auto
Pilihan: 2

====MODE AUTO====
Silahkan isi data

Jumlah token unik:
5
masukan token sebanyak 5: Contoh (BD 1C 7A 55 E9)
AA SS DD FF GG
Ukuran Buffer:
2
Ukuran Matrix: Contoh (6 6)
6 12
Jumlah Sequence:
1
Ukuran Maksimal Sequence:
10
=====
```

Result

```
=====
Matrix:
['GG' 'GG' 'FF' 'SS' 'DD' 'FF']
['FF' 'DD' 'DD' 'DD' 'AA' 'GG']
['DD' 'DD' 'AA' 'SS' 'FF' 'GG']
['DD' 'FF' 'DD' 'DD' 'DD' 'DD']
['AA' 'FF' 'GG' 'FF' 'SS' 'SS']
['SS' 'SS' 'AA' 'SS' 'SS' 'SS']
['GG' 'DD' 'SS' 'GG' 'SS' 'GG']
['DD' 'DD' 'FF' 'GG' 'GG' 'AA']
['DD' 'FF' 'DD' 'FF' 'FF' 'SS']
['DD' 'GG' 'GG' 'AA' 'FF' 'AA']
['GG' 'DD' 'GG' 'GG' 'GG' 'AA']
['DD' 'FF' 'GG' 'SS' 'GG' 'GG']

Sequences (unik):
AA GG DD GG DD AA GG SS DD AA      (bobotnya:26)
=====

Reward maksimal : 0


Waktu dibutuhkan: 0.000 ms / 0.000 detik.

Apakah anda ingin menyimpan jawaban? (y/n)
y

Masukan / buat nama file untuk menyimpan: solusiAuto_3

Telah tersimpan dengan nama file "solusiAuto_3.txt" pada folder "test"
```

Disimpan di file (solusiAuto_3.txt)

```
test >  solusiAuto_3.txt
1   Reward maksimal : 0
2
3   Waktu dibutuhkan: 0.000 ms / 0.000 detik.
```


BAB IV

Git Hub

Kode program dapat diakses pada link berikut

https://github.com/keanugonza/Tucil1_13522082.git

How to Use

- Masuk ke folder src (cd src)
- python main.py

Mohon maaf, apabila link tidak dapat diakses, dapat kontak ke email (13522082@std.stei.itb.ac.id)

BAB V

Tabel Poin

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		✓