

Tugas Kecil I IF2211 Strategi Algoritma

Penyelesaian Cyberpunk 2077 Breach Protocol dengan Algoritma Brute Force

Diajukan sebagai pemenuhan tugas kecil I



Disusun Oleh:

Wilson Yusda (13522019)

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

DAFTAR ISI

DAFTAR ISI.....	2
DAFTAR GAMBAR	3
BAB 1 DESKRIPSI MASALAH	5
BAB II LANDASAN TEORI.....	7
1. Brute-Force Algorithm.....	7
BAB III IMPLEMENTASI PROGRAM	8
1. Alur Program	8
2. Algoritma Brute-Force.....	13
Source Code Program	14
3. Struktur File	20
Front-End	20
Back-End.....	23
BAB IV PENGUJIAN	26
1. Test-Case 1 (File Input)	26
2. Test Case 2 (Random Input)	27
3. Test Case 3 (File Input).....	29
4. Test Case 4 (Random Input)	30
5. Test Case 5 (Manual Input).....	32
6. Test Case 6 (Random Input)	33
7. Test Case 7 (File Input).....	34
8. Test Case 8 (File Input).....	36
BAB V PENUTUP.....	38
1. Kesimpulan	38
2. Saran	38
BAB VI LAMPIRAN	39
1. Link Repository	39
2. Program Checkpoint	39

DAFTAR GAMBAR

Gambar 1 Contoh Permainan Breach Protocol	5
Gambar 2 Terminal Input from File	8
Gambar 3 Terminal Random Input	9
Gambar 4 Tampilan Manual Input 1	9
Gambar 5 Tampilan Manual Input 2	10
Gambar 6 Tampilan Random Input.....	10
Gambar 7 Tampilan File Input (Success).....	11
Gambar 8 Tampilan File Input (Failed)	11
Gambar 9 Tampilan ResultBox 1	12
Gambar 10 Tampilan ResultBox 2.....	12
Gambar 11 Tampilan Save File 1	13
Gambar 12 Hasil Unduhan File	13
Gambar 13 Source Code helper.py	15
Gambar 14 Souce Code readfile	16
Gambar 15 Source Code untuk Generate Matrix and Sequences	17
Gambar 16 Source Code untuk Save Result to File.....	18
Gambar 17 Source Code find_seq	19
Gambar 18 Source Code handleCalculate	21
Gambar 19 Source Code handleSubmit	22
Gambar 20 Router /calculate	23
Gambar 21 Router /random	24
Gambar 22 Router /upload.....	25
Gambar 23 File Test-Case 1 (File input).....	26
Gambar 24 Hasil Test-Case 1 (1)	26
Gambar 25 Hasil Test-Case 1 (2).....	27
Gambar 26 Test-Case 2 (Random Input)	27
Gambar 27 Hasil Test-Case 2 (1).....	28
Gambar 28 Hasil Test-Case 2 (2).....	28
Gambar 29 File Test-Case 3.....	29
Gambar 30 Hasil Test-Case 3 (1)	29
Gambar 31 Hasil Test-Case 3 (2).....	30
Gambar 32 Test-Case 4 (Random Input)	30
Gambar 33 Hasil Test-Case 4 (1).....	31
Gambar 34 Hasil Test-Case 4 (2).....	31
Gambar 35 Hasil Test-Case 5 (1).....	32
Gambar 36 Test-Case 6 (Random Input)	33
Gambar 37 Hasil Test-Case 6 (1)	33
Gambar 38 Hasil Test-Case 6 (2).....	34
Gambar 39 File Test-Case 7.....	34
Gambar 40 Hasil Test-Case 7 (1)	35
Gambar 41 Hasil Test-Case 7 (2).....	35
Gambar 42 File Test-Case 8.....	36
Gambar 43 Hasil Test-Case 8 (1).....	36

Gambar 44 Hasil Test-Case 8 (2).....	37
--------------------------------------	----

BAB 1

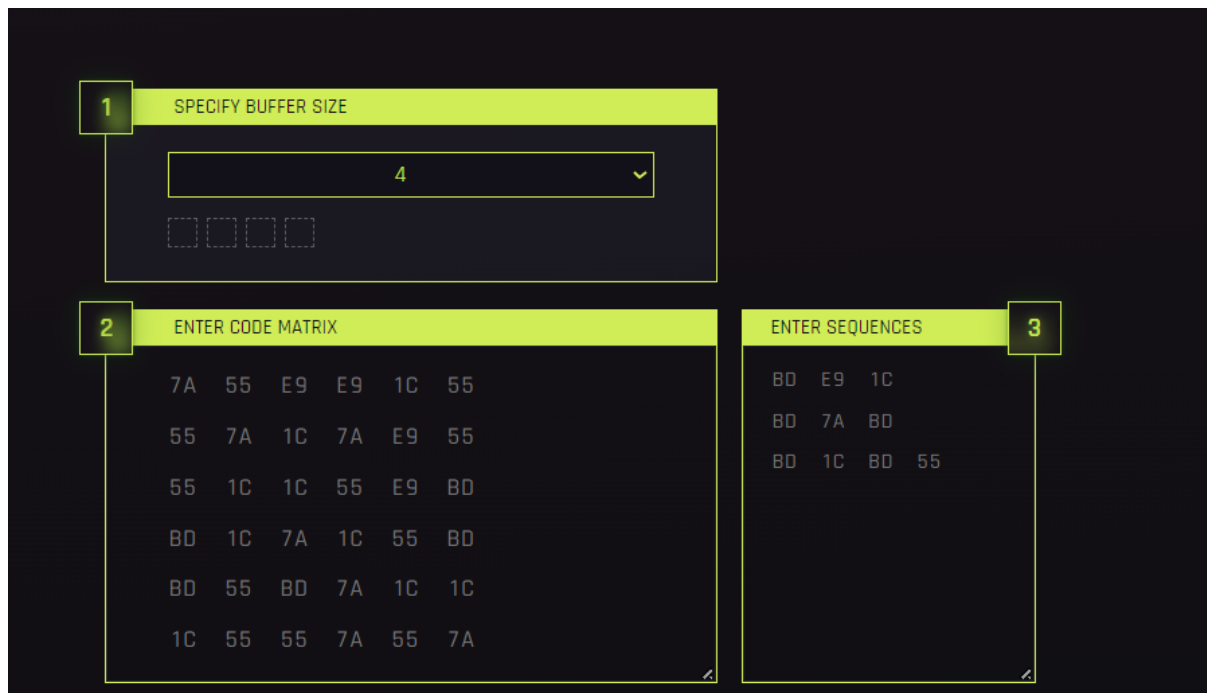
DESKRIPSI MASALAH

Cyberpunk 2077 adalah video game role-playing yang dikembangkan dan diterbitkan oleh CD Projekt. Game ini menampilkan dunia terbuka yang dinamis, dipenuhi dengan narasi yang kaya dan beragam aktivitas sampingan, salah satunya adalah "Breach Protocol."

Breach Protocol adalah mini-game hacking yang terdapat dalam Cyberpunk 2077, yang memungkinkan pemain untuk mengakses sistem keamanan, mengendalikan perangkat elektronik, mengumpulkan informasi, atau mengganggu operasi musuh dengan cara yang tidak terdeteksi. Ini merupakan salah satu elemen kunci gameplay yang memberikan kedalaman taktis dan strategis, memperkaya pengalaman pemain di dunia cyberpunk yang futuristik.

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.



Gambar 1 Contoh Permainan Breach Protocol

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.

Dalam program ini, dilakukan algoritma brute force untuk menyelesaikan mini-game Breach Protocol dalam Cyberpunk 2077. Algoritma brute force merupakan pendekatan komputasi yang mencoba semua kemungkinan solusi untuk menemukan jawaban yang benar. Dalam konteks Breach Protocol, ini berarti secara sistematis mengecek setiap kombinasi urutan kode yang mungkin dari grid untuk mencocokkan dengan urutan target yang diberikan.

BAB II

LANDASAN TEORI

1. Brute-Force Algorithm

Algoritma brute force, juga dikenal sebagai pendekatan "kekuatan kasar" atau "pencarian lengkap", adalah metode pemecahan masalah komputasi yang sangat dasar dan langsung. Algoritma ini bekerja dengan mencoba semua kemungkinan solusi untuk suatu masalah hingga solusi yang benar ditemukan. Pendekatan ini dianggap sederhana karena tidak memerlukan pengetahuan atau asumsi khusus tentang masalah yang sedang dipecahkan, menjadikannya metode yang universal dan mudah diterapkan pada berbagai jenis masalah.

Beberapa fitur brute force dapat dijelaskan sebagai berikut:

1. **Eksplorasi Semua Kemungkinan:** Algoritma brute force akan menelusuri semua kemungkinan solusi yang ada tanpa menghilangkan satu pun, hingga menemukan solusi yang benar atau terbaik.
2. **Penerapan Sederhana:** Algoritma ini biasanya mudah diimplementasikan karena hanya memerlukan iterasi atas semua kemungkinan solusi tanpa membutuhkan strategi khusus atau heuristik.
3. **Kekurangan:** Algoritma brute force sering kali tidak efisien terutama untuk masalah dengan ruang solusi yang besar, karena waktu yang dibutuhkan untuk mencari melalui semua kemungkinan solusi bisa sangat lama.
4. **Kompleksitas Waktu:** Kompleksitas waktu untuk algoritma brute force biasanya eksponensial terhadap ukuran masalah, yang berarti peningkatan kecil dalam ukuran masalah dapat menyebabkan peningkatan besar dalam waktu yang diperlukan untuk menemukan solusi.
5. **Optimalitas:** Brute force menjamin penemuan solusi yang optimal karena mengevaluasi semua kemungkinan solusi, asalkan cukup waktu dan sumber daya komputasi diberikan.

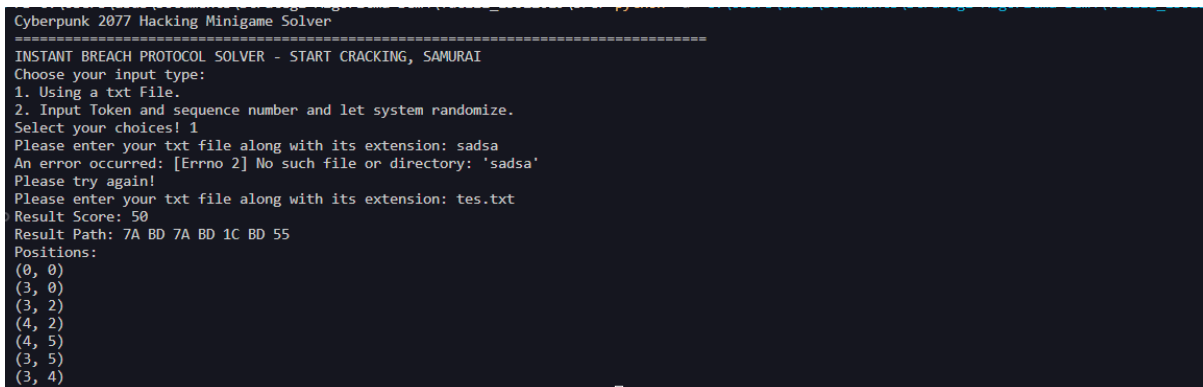
BAB III

IMPLEMENTASI PROGRAM

1. Alur Program

Dalam pengerjaan tugas kecil ini, terdapat beberapa metode masukkan informasi yakni dalam bentuk input file dalam bentuk .txt , masukan dalam bentuk CLI (Command Line Interface) dan terakhir yakni bersikap bonus dalam pengerjaan tugas kecil ini yakni berupa masukkan informasi dengan GUI (Graphical User Interface). Dalam pengerjaan tugas kecil ini, jenis GUI yang digunakan yakni berupa website, lebih tepatnya dengan menggunakan React JS sebagai frontend dan Flask sebagai backend dan pengerjaan program utama dilakukan dengan bahasa pemrograman Python.

Untuk input dari terminal, dapat dilakukan dengan menjalankan program main.py. Kemudian akan terdapat *prompt* atau tampilan mengenai metode yang diinginkan. Untuk metode pertama yakni metode input file, user cukup memasukkan nama file dari file txt berisi format inputan. Untuk metode CLI random, cukup memasukkan informasi seperti jumlah_token_unik, token, ukuran_buffer, ukuran_matriks, jumlah_sekuens, dan ukuran_maksimal_sekuens dan akan digenerate matriks dan sequence yang memenuhi inputan tersebut.



```
Cyberpunk 2077 Hacking Minigame Solver
=====
INSTANT BREACH PROTOCOL SOLVER - START CRACKING, SAMURAI
Choose your input type:
1. Using a txt File.
2. Input Token and sequence number and let system randomize.
Select your choices! 1
Please enter your txt file along with its extension: sadsa
An error occurred: [Errno 2] No such file or directory: 'sadsa'
Please try again!
Please enter your txt file along with its extension: tes.txt
Result Score: 50
Result Path: 7A BD 7A BD 1C BD 55
Positions:
(0, 0)
(3, 0)
(3, 2)
(4, 2)
(4, 5)
(4, 5)
(3, 5)
(3, 4)
```

Gambar 2 Terminal Input from File


```
Cyberpunk 2077 Hacking Minigame Solver
=====
INSTANT BREACH PROTOCOL SOLVER - START CRACKING, SAMURAI
Choose your input type:
1. Using a txt File.
2. Input Token and sequence number and let system randomize.
Select your choices! 2
How many tokens would you like to have: 3
Enter token #1 (max 2 characters): AAA
Invalid token. It must be unique and up to 2 characters long.
Enter token #1 (max 2 characters): AA
Enter token #2 (max 2 characters): AA
Invalid token. It must be unique and up to 2 characters long.
Enter token #2 (max 2 characters): BB
Enter token #3 (max 2 characters): CC
Enter max buffer size:4
Input Matrix width:5
Input Matrix height:5
How many sequence would you like to have:3
Input Sequence Max Size:2
Result Score: 142
Result Path: AA CC BB CC
Positions:
(0, 0)
(4, 0)
(4, 3)
(2, 3)
```

Gambar 3 Terminal Random Input

Input GUI dimulai dengan menjalankan frontend dan backend. Hal pertama yang dilakukan yaitu untuk menjalankan frontend yaitu dengan merujuk ke directory interface dan menjalankan npm start di terminal. Jika tidak dialihkan ke laman browser, dapat dibuka terlebih dahulu browser sebelum menjalankan npm start. Untuk backend, cukup merujuk ke directory flask-app pada terminal berbeda dan menjalankan 'python app.py' atau menekan tombol run code pada vscode.

Untuk input GUI , dibedakan menjadi 3 jenis input. Jenis pertama yakni berupa input manual dimana pada bagian ini user dapat menginput 1 per 1 informasi pada textbox yang sudah tersedia. Setelah itu, barulah user dapat menekan tombol "Find Solution" dan program akan mencari dan menampilkan solusi atas permasalahan tersebut.

Gambar 4 Tampilan Manual Input 1

Sequence	Score
YYQQ	32
WWQQ	6
QQQQ	13
QQYY	46
QQWW	84
YYYY	21

Gambar 5 Tampilan Manual Input 2

Untuk Input Manual, penulisan sequence tidak menggunakan koma dengan asumsi bahwa setiap token bersifat 2 karakter sehingga nantinya akan diparsing secara mandiri oleh program.

Untuk jenis input kedua yaitu metode Random Input dimana user juga akan memasukkan informasi yang diperlukan pada field inputan. Namun perbedaanya yaitu pada bagian ini jenis input yang diberikan yakni berupa jumlah_token_unik, token, ukuran_buffer, ukuran_matriks, jumlah_sekuens, dan ukuran_maksimal_sekuens.

Cyberpunk 2077 Hacking Minigame Solver
INSTANT BREACH PROTOCOL SOLVER - START CRACKING, SAMURAI

Manual Input Random Input File Input

Number of Tokens: 3

WW
QQ
YY

Buffer Size: 7
Matrix Width: 3
Matrix Height: 3
Number of Sequences: 6
Max Sequence Size: 2

Find Solution

Gambar 6 Tampilan Random Input

Jenis input ketiga yakni berupa input dari file dimana pada bagian ini user cukup memberikan file txt yang diformat selayaknya sesuai dengan spesifikasi input txt. Untuk hasil

input yang sesuai yang formatting yang sesuai, akan memberikan penanda kesuksesan dalam pengumpulan file txt.



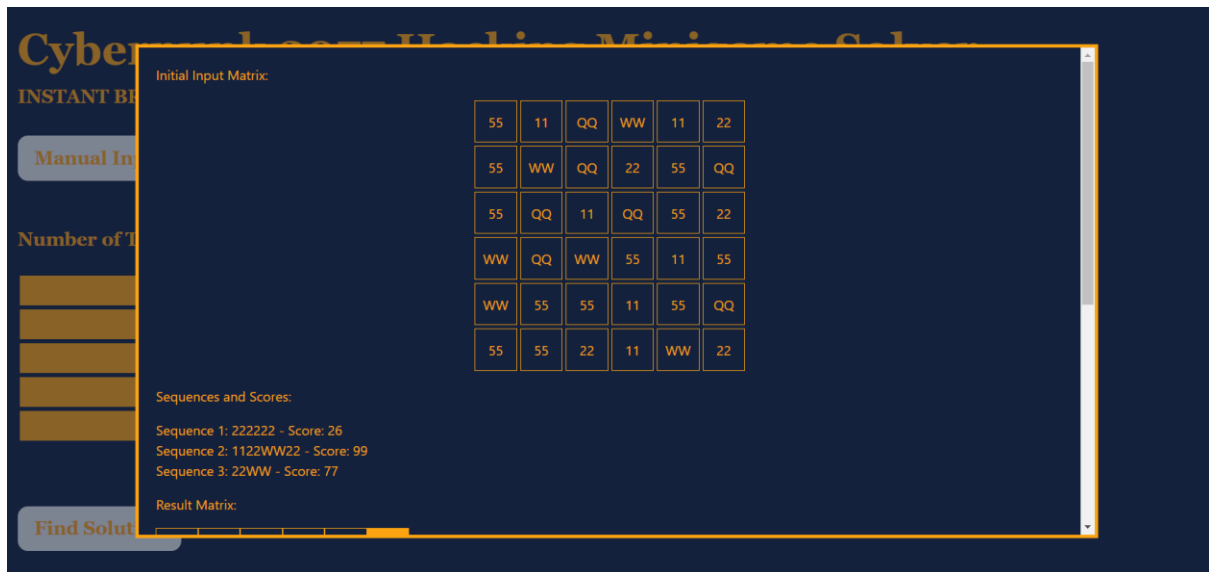
Gambar 7 Tampilan File Input (Success)

Namun untuk file yang memiliki kesalahan dalam hal formatting akan diberikan error dalam hal kesalahan di sisi formatting sehingga informasi yang didapat akan berupa kosong dan hasil dipastikan akan didapat 'No Solution'



Gambar 8 Tampilan File Input (Failed)

Ketiga bagian ini nantinya akan dicari solusi dan solusi akan diperlihatkan dalam sebuah resultbox. Dalam resultbox, terdapat beberapa informasi yang dapat user lihat yakni berupa Matrix awal inputan, Sequence beserta score inputan, Grid jalan dari solusi yang didapat, urutan path yang didapat, total score, dan run time dalam milisecond.



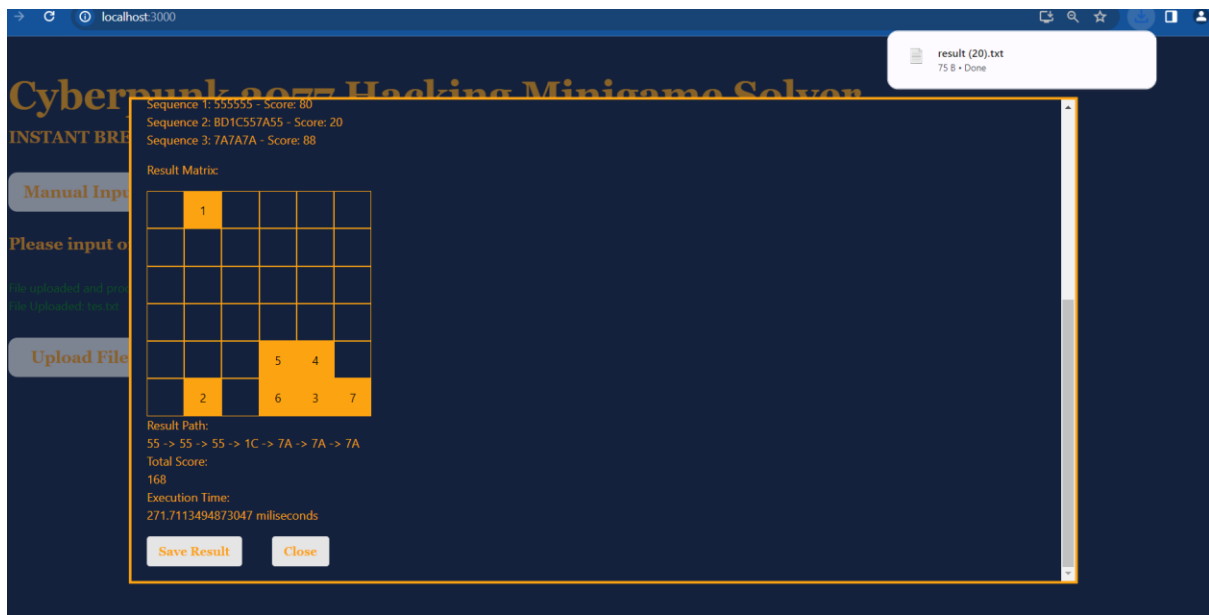
Gambar 9 Tampilan ResultBox 1



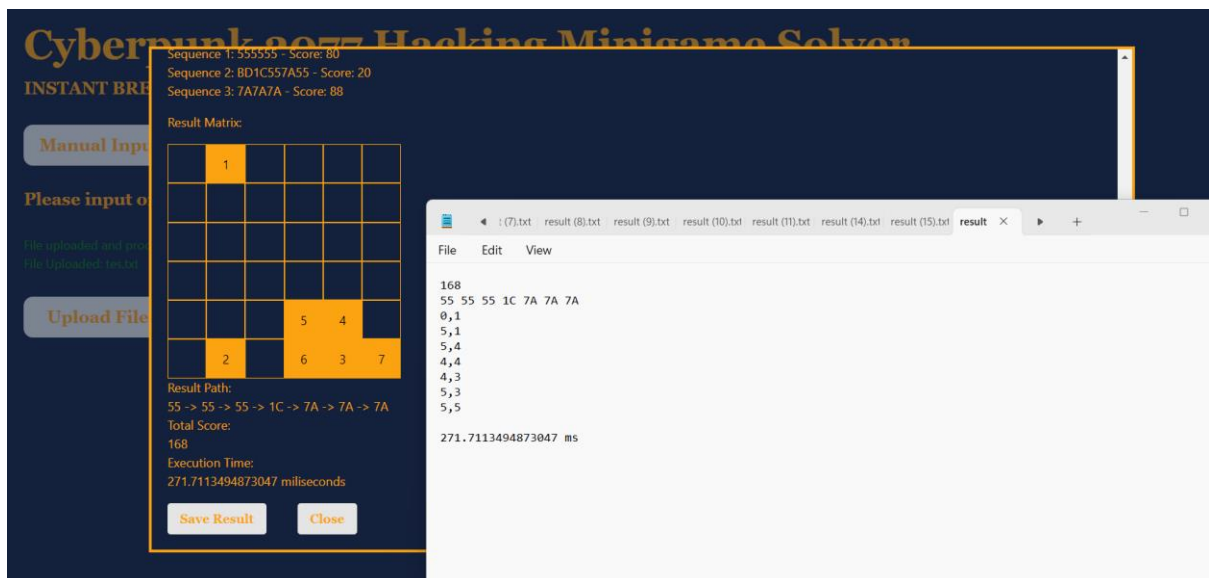
Gambar 10 Tampilan ResultBox 2

Perlu diperhatikan bahwa untuk segala jenis kesalahan input akan mempengaruhi hasil, tepatnya untuk setiap spesifikasi komponen dalam tugas kecil ini yang tidak sesuai, maka hasil yang didapat tidak akurat. Hal ini mencakup seperti banyaknya karakter pada token, sequence yang teratur dan lain sebagainya. Untuk kesalahan yang fatal, dimana formatting bersifat salah atau bahkan kosong, sistem akan mengembalikan kosong untuk segala jenis informasi sehingga solusi 'No Solution' akan diperlihatkan.

User juga dapat menyimpan hasil solusi dalam bentuk file txt dengan menekan tombol Save Result.



Gambar 11 Tampilan Save File 1



Gambar 12 Hasil Unduhan File

2. Algoritma Brute-Force

Secara garis besar, algoritma brute force yang diimplementasikan dalam tugas kecil ini dimulai dengan tahap pencarian path secara brute force. Program akan terus mencari segala jenis kemungkinan dengan memastikan jalan yang tetap berganti antara siklus horizontal dan vertikal sampai mencapai ukuran buffer maksimal. Setelah itu akan ditentukan apakah untuk setiap kombinasi ditemukan kecocokan atas sequence yang ada dan ditentukan apakah dapat

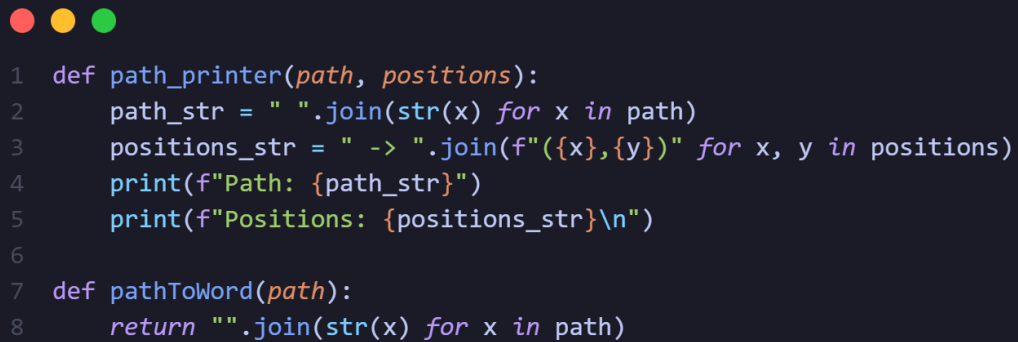
dilakukan trimming sehingga solusi yang didapatkan akan selalu menjadi solusi yang paling optimal.

Berikut tahapan yang dijalankan:

1. Inisialisasi: Program dimulai dengan menginisialisasi variabel-variabel seperti skor hasil (resultscore), jalur hasil (resultspath), dan posisi yang disimpan (savedposition). Panjang matriks juga ditentukan. Perlu juga diperhatikan bahwa segala informasi terkait pencarian solusi sudah ada pada parameter.
2. Program melakukan iterasi melalui kolom pertama matriks untuk menentukan titik awal dari setiap jalur yang mungkin.
3. Setiap kemungkinan jalur baru ditambahkan ke dalam stack.
4. Dari stack, program mengambil setiap kombinasi jalur dan posisi untuk diperiksa. Jika panjang jalur sudah sama dengan ukuran buffer, maka jalur tersebut akan dievaluasi.
5. Setiap jalur yang telah mencapai panjang buffer akan diubah menjadi string dan dibandingkan dengan setiap sequence yang dicari. Jika sebuah sequence ditemukan dalam jalur, skor akan ditambahkan
6. Program membandingkan skor total yang diperoleh dengan skor tertinggi yang sudah ada. Jika skor yang diperoleh lebih tinggi, maka program akan memperbarui jalur hasil dan skor hasil dengan nilai baru tersebut.
7. Jika terdapat beberapa jalur dengan skor yang sama, program akan memilih jalur dengan panjang terpendek yang masih mengandung semua sequence yang telah ditemukan.
8. Jika panjang jalur belum mencapai buffer, program akan menghitung posisi berikutnya yang mungkin untuk dieksplorasi dan menambahkan keadaan baru ke dalam stack untuk dieksplorasi selanjutnya.
9. Setelah semua kemungkinan telah dieksplorasi, program mengembalikan jalur dengan skor tertinggi, skor itu sendiri, dan posisi-posisi yang menjadi bagian dari jalur tersebut sebagai hasil.

Source Code Program

Sebelum melanjutkan ke program utama atas algoritma brute force , terdapat beberapa fungsi tambahan serta pembantu yang digunakan dalam pengerjaan program ini.



```
1 def path_printer(path, positions):
2     path_str = " ".join(str(x) for x in path)
3     positions_str = " -> ".join(f"({x},{y})" for x, y in positions)
4     print(f"Path: {path_str}")
5     print(f"Positions: {positions_str}\n")
6
7 def pathToWord(path):
8     return "".join(str(x) for x in path)
```

Gambar 13 Source Code helper.py

Fungsi `path_printer` berfungsi untuk menampilkan jalur dalam bentuk string dan posisi dalam bentuk tuple koordinat. Fungsi `pathToWord` berfungsi untuk mengubah path yang awalnya ada dalam list menjadi string.

Kemudian, terdapat juga fungsi yang termasuk konfigurasi yakni dalam hal handle input text, input CLI random, dan bahkan save result.

```

1  def readfile(file_path):
2      matrixInput = False
3      count = 0
4      Matrix = []
5      seqlist = []
6      scorelist = []
7      with open(file_path, 'r') as file:
8          lines = file.readlines()
9          buffer_size = int(lines[0].strip())
10         matrix_width = int((lines[1].strip().split())[0])
11         matrix_height = int((lines[1].strip().split())[1])
12         for i in range(2,2+matrix_height):
13             rows = lines[i].strip().split()
14             if (len(rows) == matrix_width):
15                 Matrix.append(rows)
16             else:
17                 print("error in reading file")
18                 return 0,[],[],[]
19         seqnum = int(lines[2+matrix_height].strip())
20         for x in range(3+matrix_height,len(lines),2):
21             seqword = pathToWord(lines[x].strip())
22             seqword = ''.join(seqword.split())
23             score = int(lines[x+1].strip())
24             seqlist.append(seqword)
25             scorelist.append(score)
26         return buffer_size,Matrix,seqlist,scorelist

```

Gambar 14 Source Code readfile

Fungsi readfile berfungsi untuk menerima input file dari user dan mengirimkan informasi yang diperlukan ke variable secara terurut. Jika terdapat kesalahan format, maka akan memberikan output error , atau bahkan error keseluruhan fungsi yang nantinya akan dihandle di Flask backend.


```

1 def generate_unique_sequences(tokens, sequenceCount, sequenceMaxSize):
2     sequences = []
3
4     while len(sequences) < sequenceCount:
5         sequence_length = random.randint(2, sequenceMaxSize)
6         sequence = [random.choice(tokens) for _ in range(sequence_length)]
7         sequence_tuple = tuple(sequence)
8         existing_sequences = {tuple(seq) for seq in sequences}
9         if sequence_tuple not in existing_sequences:
10             sequences.append(sequence)
11     return sequences
12
13 def generate_scores(sequence_count):
14     scores = set()
15     while len(scores) < sequence_count:
16         scores.add(random.randint(0, 100))
17     return list(scores)
18
19 def generate_matrix_and_sequences(tokens, sequenceCount, matrix_width, matrix_height, sequenceMaxSize):
20     matrix = [[random.choice(tokens) for _ in range(matrix_width)] for _ in range(matrix_height)]
21     sequences = generate_unique_sequences(tokens, sequenceCount, sequenceMaxSize)
22     scores = generate_scores(sequenceCount)
23     return matrix, sequences, scores

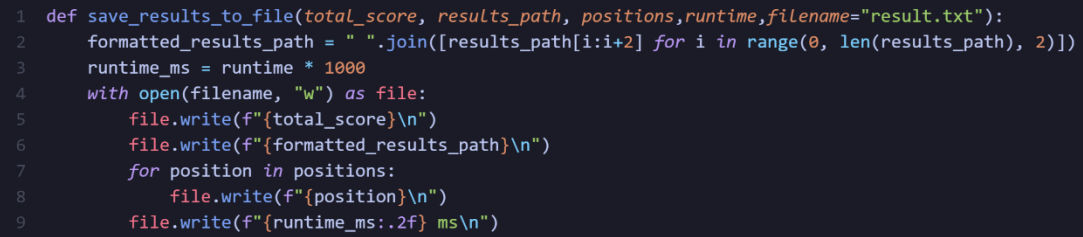
```

Gambar 15 Source Code untuk Generate Matrix and Sequences

Fungsi `generate_matrix_and_sequences` digunakan untuk secara otomatis menghasilkan sebuah matriks dan sekumpulan sequence berserta skor masing-masing untuk digunakan dalam program utama. Matriks dihasilkan dengan dimensi yang ditentukan oleh lebar (`matrix_width`) dan tinggi (`matrix_height`), diisi dengan elemen yang dipilih secara acak dari daftar token yang diberikan.

Selanjutnya, fungsi `generate_unique_sequences` membangkitkan sekumpulan sequence yang unik berdasarkan token yang tersedia, jumlah sequence yang diinginkan (`sequenceCount`), dan ukuran maksimal dari setiap sequence (`sequenceMaxSize`). Ini memastikan bahwa setiap sequence adalah kombinasi unik dari token yang mungkin, dengan panjang antara 2 hingga `sequenceMaxSize` token. Keunikan dijamin dengan membandingkan setiap sequence baru yang dihasilkan dengan sequence yang telah ada sebelumnya.

Fungsi `generate_scores` menghasilkan skor untuk setiap sequence yang dibangkitkan, dengan nilai acak antara 0 hingga 100. Ini meniru mekanisme skor dalam permainan, di mana setiap sequence memiliki nilai tertentu yang menambah skor pemain ketika berhasil ditemukan dan diikuti dengan benar.

A code editor window with a dark background and three colored window control buttons (red, yellow, green) in the top-left corner. The code is written in Python and defines a function named `save_results_to_file`. The function takes five arguments: `total_score`, `results_path`, `positions`, `runtime`, and `filename` (with a default value of "result.txt"). The code calculates a formatted path by joining elements of `results_path` with a space character. It then converts the runtime to milliseconds by multiplying by 1000. Finally, it opens the file in write mode and writes the total score, formatted path, each position, and the runtime in milliseconds to the file, each on a new line.

```
1 def save_results_to_file(total_score, results_path, positions, runtime, filename="result.txt"):
2     formatted_results_path = " ".join([results_path[i:i+2] for i in range(0, len(results_path), 2)])
3     runtime_ms = runtime * 1000
4     with open(filename, "w") as file:
5         file.write(f"{total_score}\n")
6         file.write(f"{formatted_results_path}\n")
7         for position in positions:
8             file.write(f"{position}\n")
9         file.write(f"{runtime_ms:.2f} ms\n")
```

Gambar 16 Source Code untuk Save Result to File

Fungsi `save_result_to file` berfungsi untuk menyimpan hasil pencarian solusi ke dalam file `txt`. Untuk tujuan folder tidak dispesifikasikan mengingat bahwa pengerjaan tugas kecil ini menggunakan GUI sehingga juga memiliki metode berbeda. Metode ini hanya untuk jika ingin menggunakan input terminal.

```

1 def find_seq(buffer, matrix, start_x, start_y, movement, sequence, score):
2     resultscore = 0
3     resultspath = ''
4     savedposition = []
5     length = len(matrix[0])
6     for x in range(length):
7         stack = [(start_x, x, movement, [matrix[start_x][x]], [(start_x, x)])]
8         while stack:
9             current_x, current_y, current_movement, path, positions = stack.pop()
10            if len(path) == buffer and len(path):
11                path_str = pathToWord(path)
12                # print(path_str)
13                totalscore = 0
14                endpoint = -1
15                FinalEndpoint = -1
16                for count, datamine in enumerate(sequence):
17                    initialsore = score[count]
18                    pos = path_str.find(datamine)
19                    if pos != -1:
20                        totalscore += initialsore
21                        endpoint = pos + len(datamine) - 1
22                        if (endpoint > FinalEndpoint):
23                            FinalEndpoint = endpoint
24                    if totalscore > resultscore:
25                        resultscore = totalscore
26                        resultspath = path_str
27                        savedposition = positions
28                    if totalscore == resultscore:
29                        if len(pathToWord(resultspath)) > len(path_str[:FinalEndpoint + 1]):
30                            resultspath = path_str[:FinalEndpoint+1]
31                            coordinatecut = (FinalEndpoint+1)/2
32                            savedposition = positions[:int(coordinatecut)]
33
34            else:
35                next_movements = range(len(matrix)) if current_movement == "vertical" else range(len(matrix[0]))
36                for next_pos in next_movements:
37                    new_x, new_y = (next_pos, current_y) if current_movement == "vertical" else (current_x, next_pos)
38                    if (new_x, new_y) not in positions:
39                        new_path = path.copy()
40                        new_positions = positions.copy()
41                        new_path.append(matrix[new_x][new_y])
42                        new_positions.append((new_x, new_y))
43                        next_movement = "horizontal" if current_movement == "vertical" else "vertical"
44                        stack.append((new_x, new_y, next_movement, new_path, new_positions))
45        return resultspath, resultscore, savedposition

```

Gambar 17 Source Code find_seq

Fungsi find_seq merupakan fungsi utama serta merupakan algoritma brute force sendiri dalam tugas kecil ini. Fungsi ini menerima buffer maksimal, matriks, posisi x dan y awal, sequence (dalam list) dan score (dalam list). Program ini dimulai dengan mengiterasi sebanyak jumlah kolom dimana hal ini dilakukan karena pencarian path dimulai dengan baris pertama.

Fungsi find_seq ini meneruskan proses iterasinya dengan menggunakan struktur data stack untuk menyimpan setiap kemungkinan path yang bisa diambil. Untuk setiap elemen dalam stack, program mengecek apakah panjang path yang terbentuk sama dengan buffer maksimal dan apakah path tersebut sesuai dengan salah satu sequence yang dicari. Jika ya, maka akan dihitung total skor berdasarkan sequence yang ditemukan dalam path tersebut.

Jika total skor yang diperoleh lebih besar dari skor terbaik yang sudah ada, maka skor tersebut akan menjadi skor terbaik baru, dan path serta posisi yang menghasilkan skor tersebut akan disimpan sebagai hasil terbaik sementara. Jika total skor yang diperoleh sama dengan skor terbaik yang sudah ada, program kemudian akan membandingkan panjang dari path terbaik yang sudah ada dengan path baru yang menghasilkan skor sama. Jika path baru lebih pendek, maka path baru akan menjadi path terbaik sementara. Ini dilakukan untuk memastikan bahwa path yang dipilih adalah yang paling efisien.

Dalam setiap iterasi, program juga menentukan arah gerakan selanjutnya, yang bergantian antara vertikal dan horizontal, untuk mengeksplorasi semua kemungkinan path dalam matriks. Setiap kali mengubah arah, program mengecek posisi baru yang akan ditambahkan ke path untuk memastikan bahwa posisi tersebut belum pernah dikunjungi sebelumnya dalam path yang sama, mencegah pembentukan loop.

Setelah mengeksplorasi semua kemungkinan path, fungsi `find_seq` akan mengembalikan path dengan skor tertinggi, total skor dari path tersebut, dan posisi-posisi yang dilalui dalam path.

3. Struktur File

Secara garis besar, didalam folder `src`, hanya terdapat 4 folder yakni `__pycache__`, `flask-app`, `interface`, dan `Uploaded`. `Uploaded` merupakan tempat dari file yang dikumpulkan dari web, sedangkan `pycache` berupa cache dari program yang dijalankan. `Flask-app` mencakup keseluruhan program yang menggunakan python, mulai dr interface terminal, program utama brute force, sampai `app.py`, yang merupakan backend utama dari program ini. Interface berisi segala jenis file yang saling mendukung untuk menampilkan GUI dalam program ini

Front-End

Perancangan dan pembuatan front end dalam tugas kecil ini menggunakan Framework `React JS` dan di *styling* dengan menggunakan `tailwind CSS`. Untuk keseluruhan program dapat dilihat di pranala github dibawah namun terdapat beberapa fitur yang akan dijelaskan disini.



```

1  const handleCalculate = async () => {
2    setIsLoading(true);
3    setIsResultBoxVisible(false);
4    try {
5      const response = await fetch('http://localhost:5000/calculate', {
6        method: 'POST',
7        headers: {
8          'Content-Type': 'application/json',
9        },
10       body: JSON.stringify({ matrix, bufferSize, sequences, scores }),
11     });
12     if (!response.ok) {
13       throw new Error('Network response was not ok');
14     }
15     const result = await response.json();
16     setPath(result.result_path);
17     setResultMatrix(result.result_matrix);
18     setInitialMatrix(result.initial_matrix);
19     setResultScore(result.result_scores);
20     setTime(result.time)
21
22     console.log(result.result_matrix);
23   } catch (error) {
24     console.error('Error during operation:', error);
25   }
26   setIsLoading(false);
27   setIsResultBoxVisible(true);
28 };

```

Gambar 18 Source Code handleCalculate

handleCalculate merupakan salah satu metode untuk menghubungkan frontend dengan backend serta membuat adanya interaksi dalam website yang dibuat. handleCalculate dihubungkan dengan calculate dalam local host 5000 dan mengirimkan informasi seperti matrix, bufferSize, sequences, dan scores. handleCalculate juga menerima data dari hasil perhitungan seperti result_path, result_matrix, initial_matrix, result_scores, dan time (lama jalan program). Setelah program menerima hasil barulah ditampilkan dalam result box.



```

1  const handleSubmit = async () => {
2    setIsLoading(true);
3    setIsResultBoxVisible(false);
4    const payload = {
5      tokens,
6      bufferSize: parseInt(bufferSize, 10),
7      matrixWidth: parseInt(cols, 10),
8      matrixHeight: parseInt(rows, 10),
9      numSeq: parseInt(numSeq, 10),
10     maxSequenceSize: parseInt(maxSequenceSize, 10)
11   };
12   try {
13     const response = await fetch('http://localhost:5000/random', {
14       method: 'POST',
15       headers: {
16         'Content-Type': 'application/json',
17       },
18       body: JSON.stringify(payload),
19     });
20
21     if (!response.ok) throw new Error('Network response was not ok');
22     const result = await response.json();
23     setPath(result.result_path);
24     setResultMatrix(result.result_matrix);
25     setInitialMatrix(result.initial_matrix);
26     setMatrix(result.initial_matrix)
27     setResultScore(result.result_scores);
28     setScores(result.scores)
29     setSequences(result.sequences)
30     setTime(result.time)
31   } catch (error) {
32     console.error('Error:', error);
33   }
34   setIsLoading(false);
35   setIsResultBoxVisible(true);
36 };

```

Gambar 19 Source Code handleSubmit

handleSubmit bekerja hampir serupa dengan handleCalculate dimana perbedaannya ada pada sebelum dilakukan kalkulasi, terlebih dahulu dijalankan pada program random generator yang pada folder flask-app, sehingga terdapat beberapa return tambahan pada result seperti score, sequences serta matrix awal.

Back-End

Untuk Backend, digunakan Flask dalam penghubungan dengan frontend React JS. Dalam laporan ini, hanya akan ditampilkan source code untuk tiap route. Source code lengkap dapat dilihat dari pranala github pada lampiran.

A screenshot of a code editor with a dark background and light-colored text. The code is written in Python and defines a Flask route for the endpoint '/calculate'. The route is decorated with '@app.route' and the method 'POST'. The function 'calculate()' starts by printing 'tes', then retrieves data from the request using 'request.get_json()'. It extracts 'matrix', 'bufferSize' (default 1), 'sequences' (default []), and 'scores' (default []). It then records the start time with 'time.time()'. A call to 'find_seq()' is made with parameters: 'buffer_size', 'matrix', '0', '0', 'vertical', 'sequences', and 'scores'. The end time is recorded with 'time.time()', and the execution time 'gap' is calculated as '(end-start) * 1000'. Finally, a JSON response is returned containing 'result_path', 'result_scores', 'initial_matrix', 'result_matrix', 'buffer_size', 'sequences', 'scores', and 'time'.

```
1 @app.route('/calculate', methods=['POST'])
2 def calculate():
3     print("tes")
4     data = request.get_json()
5     matrix = data['matrix']
6     buffer_size = data.get('bufferSize', 1)
7     sequences = data.get('sequences', [])
8     scores = data.get('scores', [])
9     start = time.time()
10    resultspath, resultscore, position = find_seq(buffer_size, matrix, 0, 0, "vertical", sequences, scores)
11    end = time.time()
12    gap = (end-start) * 1000
13    return jsonify({
14        'result_path':resultspath,
15        'result_scores': resultscore,
16        'initial_matrix': matrix,
17        'result_matrix': position,
18        'buffer_size': buffer_size,
19        'sequences': sequences,
20        'scores': scores,
21        'time':gap
22    })
```

Gambar 20 Router /calculate

Route /calculate merupakan endpoint dalam aplikasi Flask yang berfungsi untuk melakukan kalkulasi pada data yang dikirim oleh frontend. Data ini meliputi matrix, ukuran buffer, sequences, dan scores. Fungsi calculate() diawali dengan menerima data yang dikirim dalam format JSON melalui request.get_json(). Kemudian, data ini dipecah menjadi variabel-variabel yang sesuai.

Setelah menerima semua data, fungsi ini mencatat waktu mulai eksekusi dengan time.time(), menjalankan fungsi find_seq() dengan parameter yang diterima untuk mendapatkan resultspath, resultscore, dan position. Setelah eksekusi selesai, fungsi ini mencatat waktu selesai dan menghitung selisih waktu eksekusi dalam milidetik (gap), yang merupakan waktu eksekusi dari algoritma find_seq().

Akhirnya, fungsi ini mengembalikan response dalam format JSON yang berisi jalur hasil (result_path), skor hasil (result_scores), matriks awal (initial_matrix), matriks hasil (result_matrix), ukuran buffer (buffer_size), sequence (sequences), skor (scores), dan waktu

eksekusi (time). Ini memungkinkan frontend untuk menampilkan hasil dari kalkulasi kepada pengguna.



```
1 @app.route('/random', methods=['POST'])
2 def generate_random():
3     data = request.get_json()
4     tokens = data['tokens']
5     buffer_size = data['bufferSize']
6     matrix_width = data['matrixWidth']
7     matrix_height = data['matrixHeight']
8     num_seq = data['numSeq']
9     max_sequence_size = data['maxSequenceSize']
10    matrix, sequences, scores = generate_matrix_and_sequences(tokens, num_seq, matrix_width, matrix_height, max_sequence_size)
11    sequencescombined = [pathToWord(x) for x in sequences]
12    start = time.time()
13    resultspath, resultscore, position = find_seq(buffer_size, matrix, 0, 0, "vertical", sequencescombined, scores)
14    end = time.time()
15    gap = (end-start)*1000
16    return jsonify({
17        'result_path':resultspath,
18        'initial_matrix': matrix,
19        'result_matrix': position,
20        'buffer_size': buffer_size,
21        'sequences': sequencescombined,
22        'result_scores': resultscore,
23        'scores':scores,
24        'time':gap
25    })
```

Gambar 21 Router /random

Endpoint /random pada aplikasi Flask ini digunakan untuk menghasilkan data secara acak dan melakukan kalkulasi berdasarkan data tersebut. Fungsi generate_random() dijalankan ketika ada permintaan POST ke route ini. Fungsi ini memulai dengan mengambil data dari permintaan dengan request.get_json() dan menyimpannya dalam variabel yang sesuai. Data tersebut meliputi token-token yang digunakan, ukuran buffer, lebar dan tinggi matriks, jumlah sequence, dan ukuran maksimal sequence.

Setelah itu, fungsi generate_matrix_and_sequences() dipanggil dengan parameter yang diperoleh dari data permintaan untuk menghasilkan matriks acak dan sequence yang unik beserta skornya. Setiap sequence yang dihasilkan kemudian diubah menjadi string dengan fungsi pathToWord().

Selanjutnya, fungsi find_seq() dipanggil untuk mencari path terbaik yang mungkin dari matriks berdasarkan sequence dan skor yang dihasilkan. Pencarian ini dimulai dari sudut kiri atas matriks dan bergerak secara vertikal. Waktu mulai dan selesai pencarian dicatat untuk menghitung durasi pencarian dalam milidetik.

Akhirnya, fungsi mengembalikan respons JSON yang berisi path hasil (result_path), skor hasil (result_scores), matriks awal (initial_matrix), matriks hasil (result_matrix), ukuran

buffer (buffer_size), sequence (sequences), skor (scores), dan waktu yang diperlukan untuk eksekusi (time). Ini memungkinkan frontend untuk menampilkan hasil yang telah dihitung berdasarkan data yang dihasilkan secara acak.

```
1 @app.route('/upload', methods=['POST'])
2 def upload_file():
3     if 'file' not in request.files:
4         return jsonify({'message': 'No file part', 'status': 'failed'}), 400
5     file = request.files['file']
6     if file.filename == '':
7         return jsonify({'message': 'No selected file', 'status': 'failed'}), 400
8     if file and file.filename.endswith('.txt'):
9         filename = secure_filename(file.filename)
10        filepath = os.path.join(app.config['UPLOAD_FOLDER'], filename)
11        file.save(filepath)
12        print("upload done")
13        try:
14            buffer_size, matrix, sequences, scores = readfile(filepath)
15            width = len(matrix[0])
16            height = len(matrix)
17            return jsonify({
18                'message': 'File uploaded and processed successfully',
19                'filepath': filepath,
20                'buffer_size': buffer_size,
21                'matrix': matrix,
22                'sequences': sequences,
23                'scores': scores,
24                'status': 'success',
25                'width': width,
26                'height': height
27            }), 200
28        except Exception as e:
29            print(f"Error processing file: {e}")
30            return jsonify({
31                'message': 'Error processing the uploaded file',
32                'status': 'failed'
33            }), 500
34    else:
35        return jsonify({'message': 'Invalid file type, only .txt files are accepted', 'status': 'failed'}), 400
```

Gambar 22 Router /upload

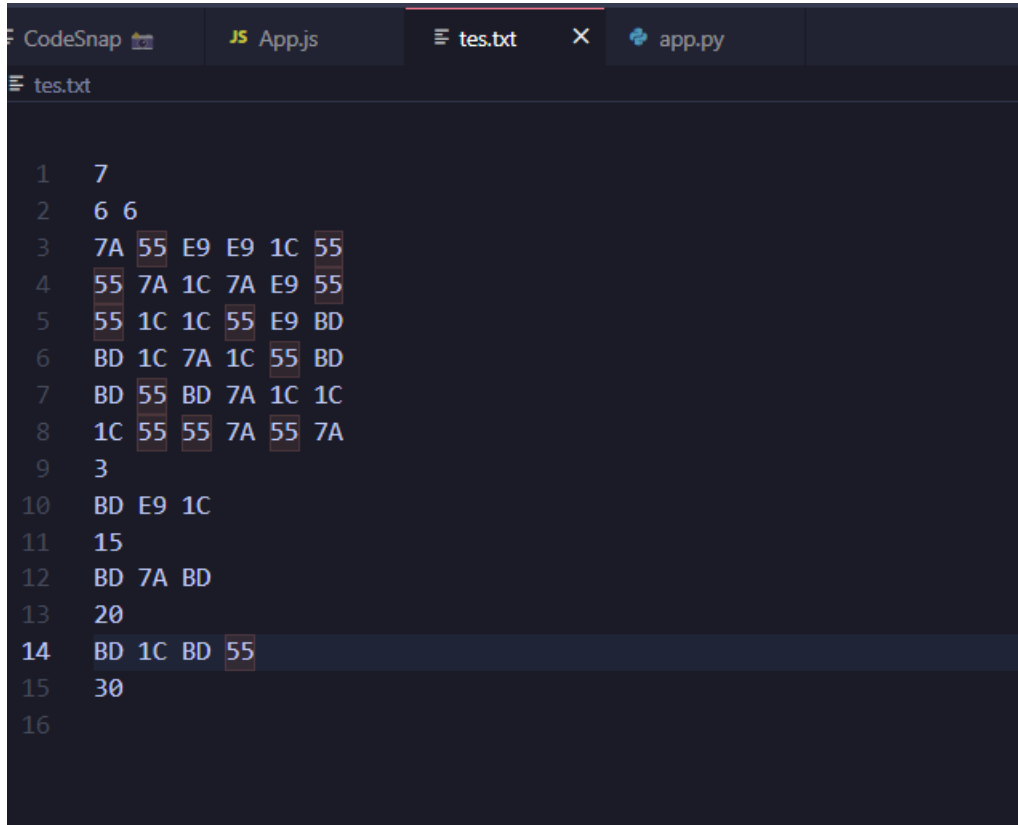
Route /upload pada aplikasi Flask ini bertugas menangani pengunggahan file dari pengguna. Pertama, ia memeriksa apakah ada file yang dikirimkan dengan permintaan. Jika tidak ada file atau nama file kosong, server akan mengembalikan respons JSON yang menunjukkan kegagalan. Jika file ada dan memiliki ekstensi .txt, file tersebut disimpan ke lokasi yang aman menggunakan secure_filename dan disimpan ke folder yang ditentukan dalam konfigurasi aplikasi.

Setelah file disimpan, fungsi readfile() dipanggil untuk membaca dan memproses isi file. Fungsi ini mengembalikan ukuran buffer, matriks, sequences, dan skor yang dibaca dari file. Ukuran matriks (lebar dan tinggi) dihitung dan semua data yang relevan dikirim kembali sebagai respons JSON jika berhasil. Jika terjadi kesalahan selama proses ini, server akan menangkap eksepsi tersebut dan mengembalikan pesan kesalahan melalui respons JSON.

BAB IV

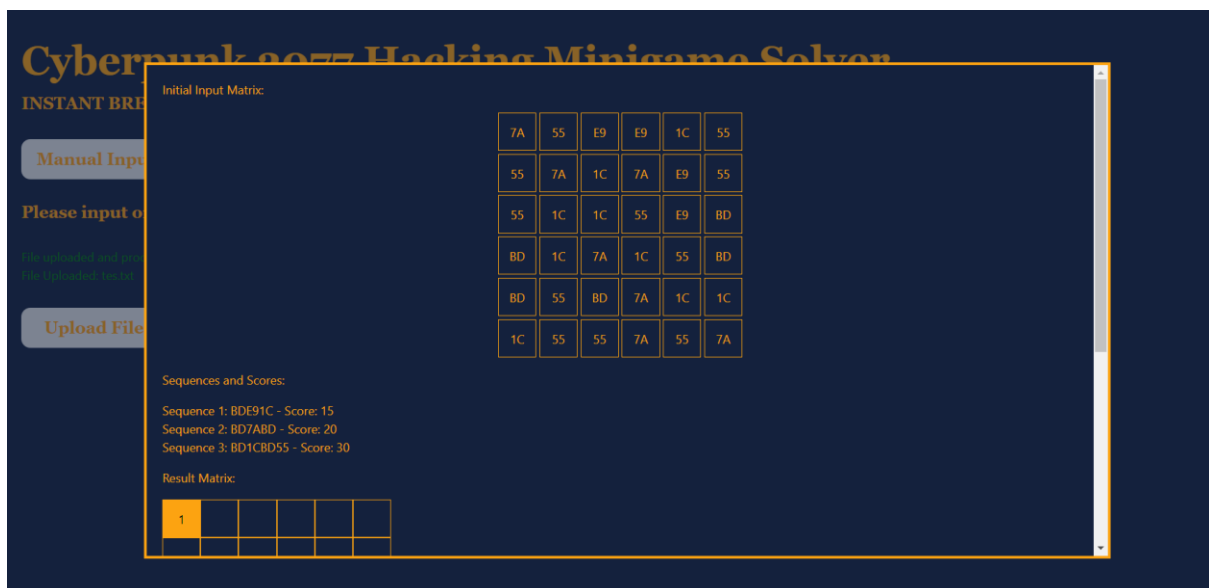
PENGUJIAN

1. Test-Case 1 (File Input)

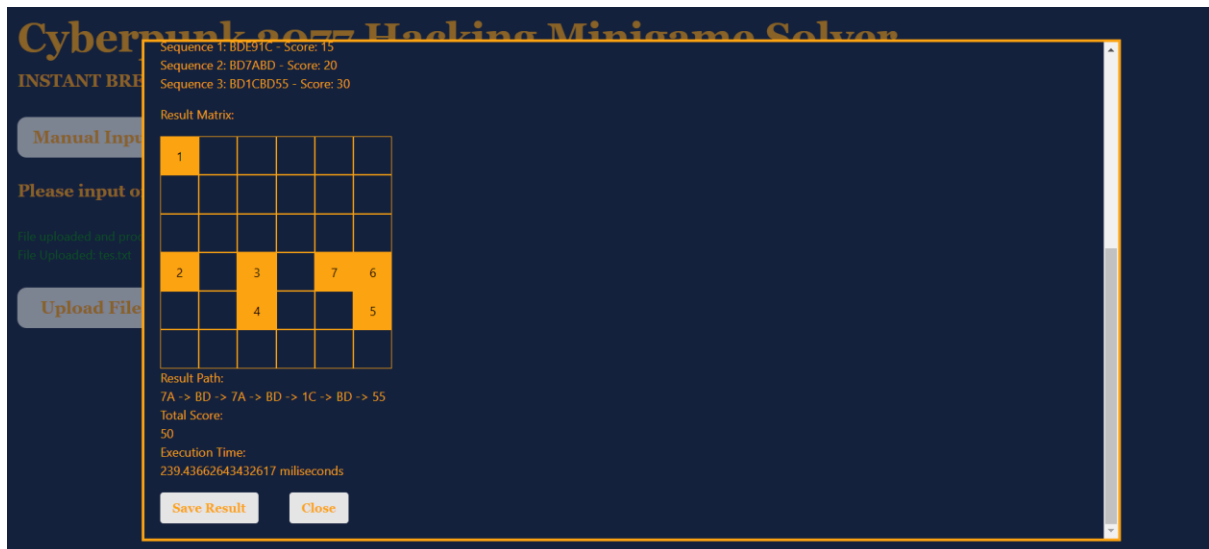


```
1 7
2 6 6
3 7A 55 E9 E9 1C 55
4 55 7A 1C 7A E9 55
5 55 1C 1C 55 E9 BD
6 BD 1C 7A 1C 55 BD
7 BD 55 BD 7A 1C 1C
8 1C 55 55 7A 55 7A
9 3
10 BD E9 1C
11 15
12 BD 7A BD
13 20
14 BD 1C BD 55
15 30
16
```

Gambar 23 File Test-Case 1 (File input)



Gambar 24 Hasil Test-Case 1 (1)



Gambar 25 Hasil Test-Case 1 (2)

2. Test Case 2 (Random Input)

Cyberpunk 2077 Hacking Minigame Solver
 INSTANT BREACH PROTOCOL SOLVER - START CRACKING, SAMURAI

Manual Input Random Input File Input

Number of Tokens: 5

Buffer Size: 7

Matrix Width: 6

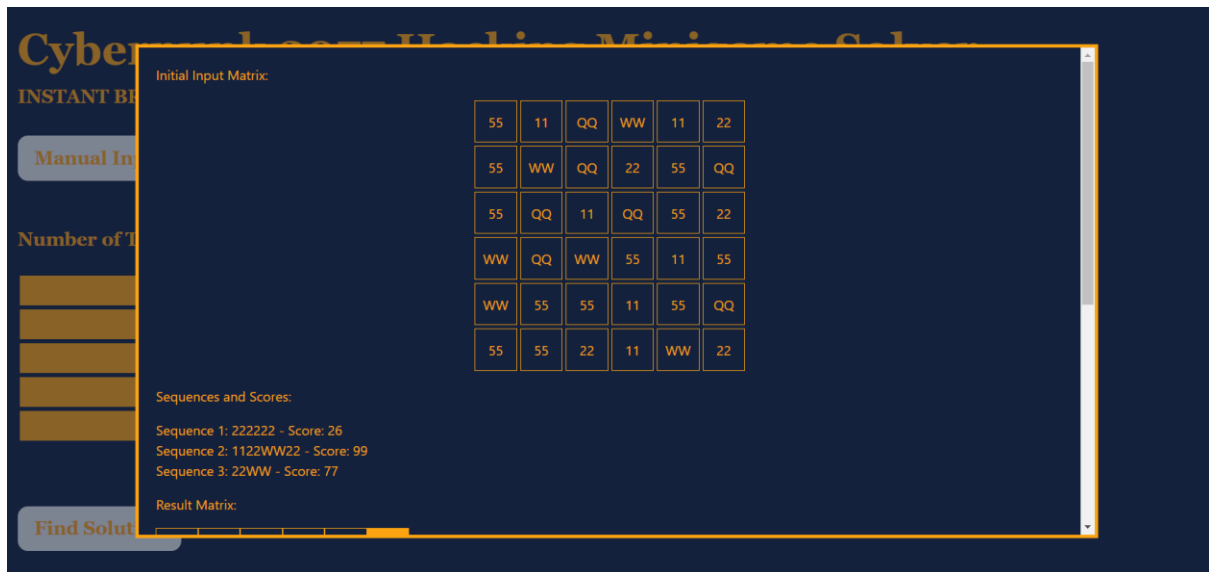
Matrix Height: 6

Number of Sequences: 3

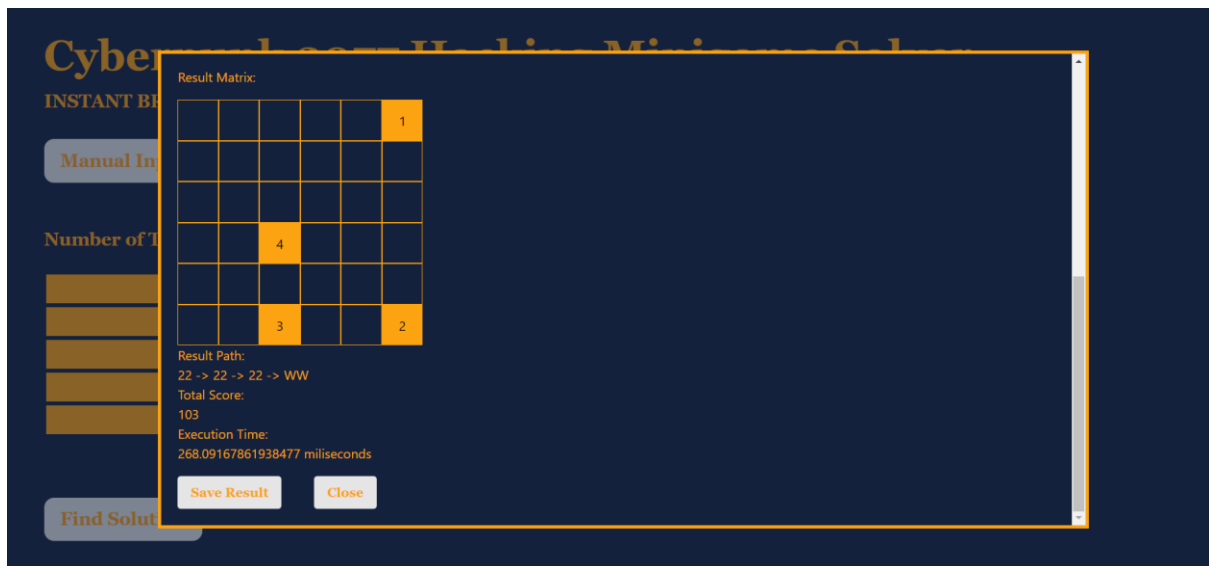
Max Sequence Size: 4

Find Solution

Gambar 26 Test-Case 2 (Random Input)

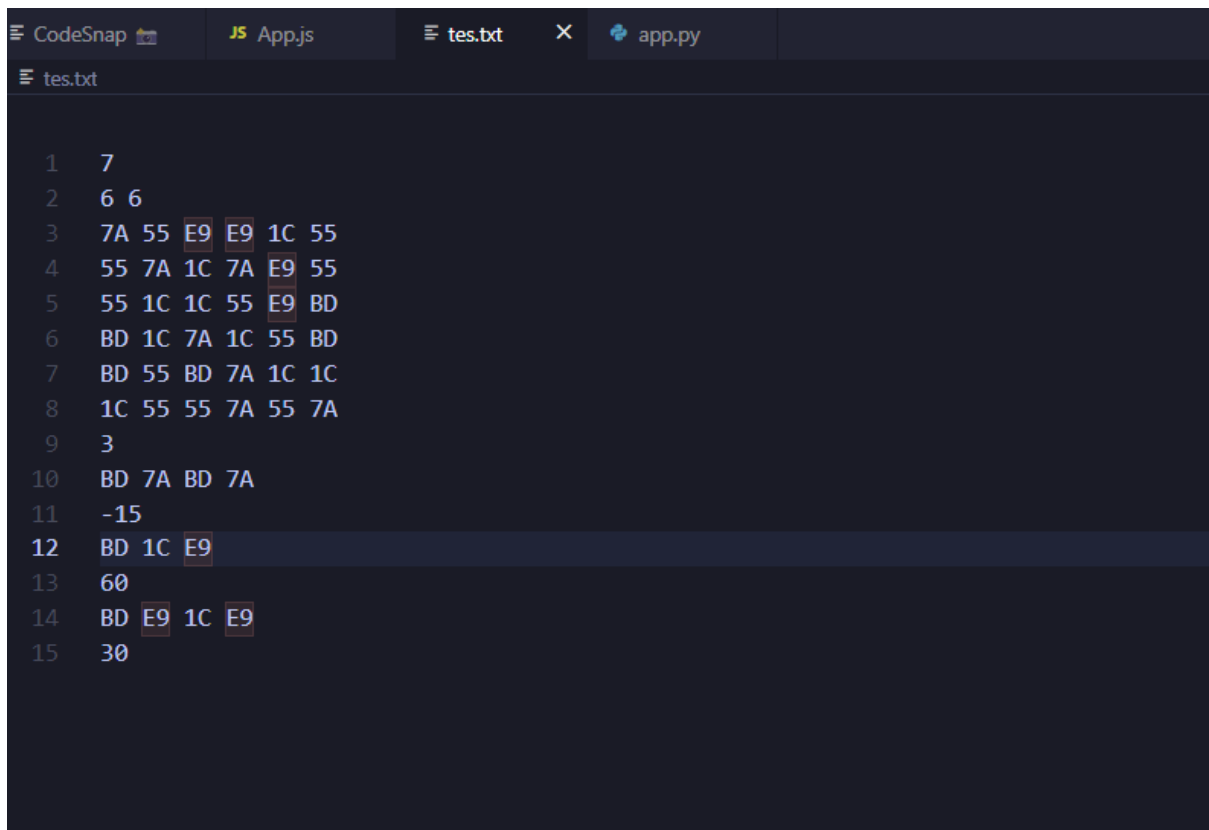


Gambar 27 Hasil Test-Case 2 (1)



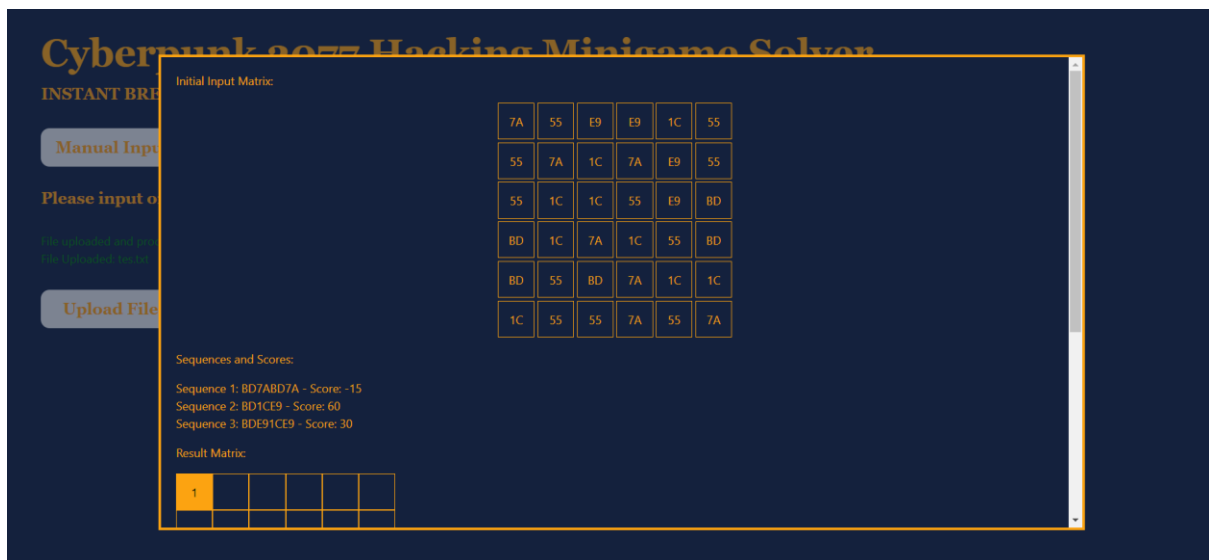
Gambar 28 Hasil Test-Case 2 (2)

3. Test Case 3 (File Input)



```
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 7A BD 7A
11 -15
12 BD 1C E9
13 60
14 BD E9 1C E9
15 30
```

Gambar 29 File Test-Case 3



Cyberpunk 2077 Hacking Minigame Solver

INSTANT BRUTE FORCE

Manual Input

Please input the sequence of the hacking minigame

Downloaded and processed the input file

Upload File

Initial Input Matrix:

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

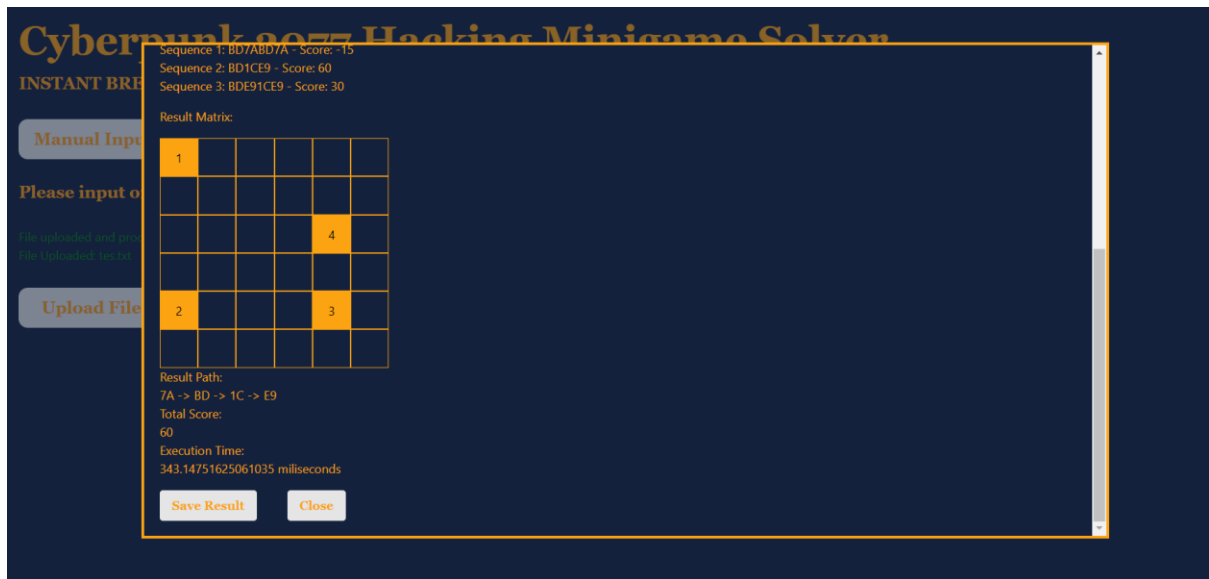
Sequences and Scores:

Sequence 1: BD7ABD7A - Score: -15
Sequence 2: BD1CE9 - Score: 60
Sequence 3: BDE91CE9 - Score: 30

Result Matrix:

1					

Gambar 30 Hasil Test-Case 3 (1)

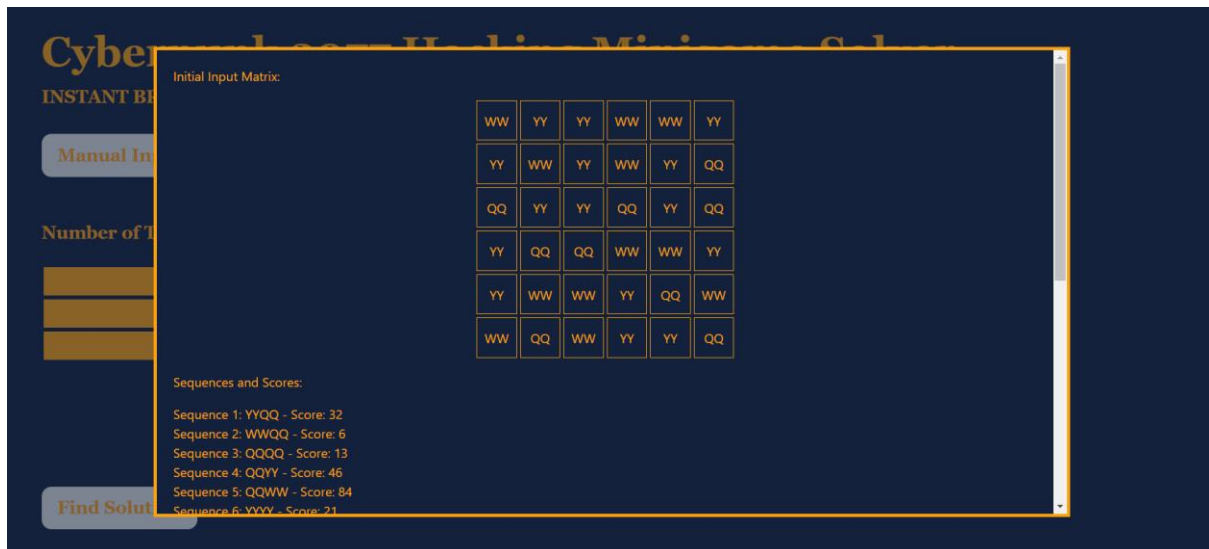


Gambar 31 Hasil Test-Case 3 (2)

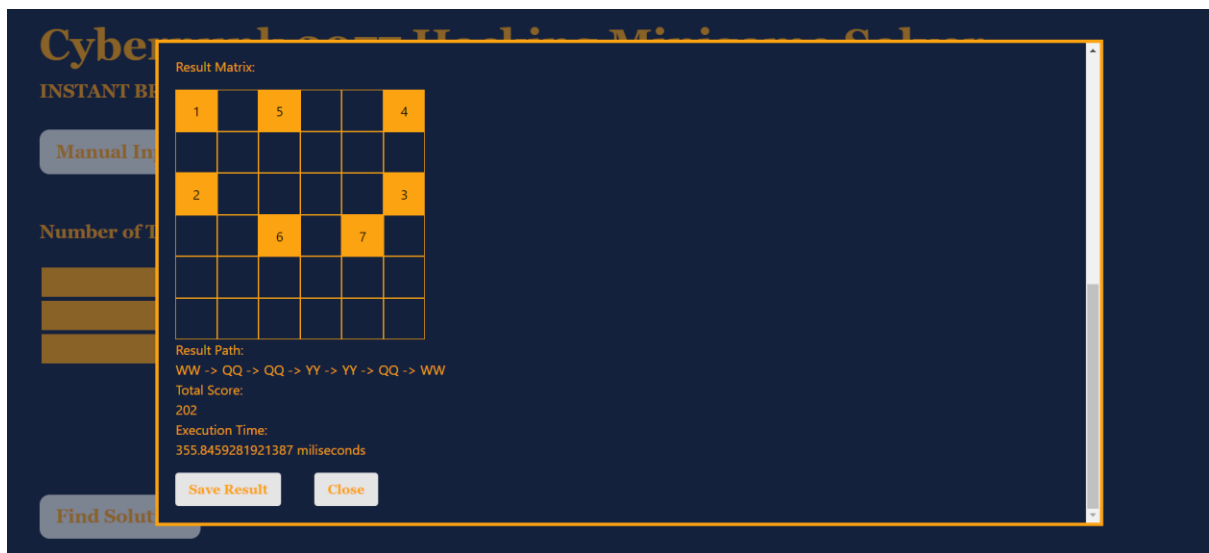
4. Test Case 4 (Random Input)



Gambar 32 Test-Case 4 (Random Input)



Gambar 33 Hasil Test-Case 4 (1)



Gambar 34 Hasil Test-Case 4 (2)

5. Test Case 5 (Manual Input)

Cyberpunk 2077 Hacking Minigame Solver

INSTANT BREACH PROTOCOL SOLVER - START CRACKING, SAMURAI

Manual Input Random Input File Input

Rows: Columns: **Generate Matrix**

Buffer:

AA	QQ	CC
BB	QQ	BB
CC	QQ	AA

Number of Sequences: **Generate Sequences**

DDCCEE	Score:	89
CCCCD	Score:	83
CCCC	Score:	68
BBAABB	Score:	85

Find Solution

Cyberpunk 2077 Hacking Minigame Solver

INSTANT BREACH PROTOCOL SOLVER - START CRACKING, SAMURAI

Manual Input Random Input File Input

Rows: Columns: **Generate Matrix**

Buffer:

AA	QQ	CC
BB	QQ	BB
CC	QQ	AA

Number of Sequences: **Generate Sequences**

DDCCEE	Score:	89
CCCCD	Score:	83
CCCC	Score:	68
BBAABB	Score:	85

Find Solution

Initial Input Matrix

AA	QQ	CC
BB	QQ	BB
CC	QQ	AA

Sequences and Scores:

Sequence 1: DDCCEE - Score: 89
Sequence 2: CCCCCD - Score: 83
Sequence 3: CCCC - Score: 68
Sequence 4: BBAABB - Score: 85

No Solution Found

Result Path:
Total Score:
0
Execution Time:
0.9720325469970703 milliseconds

Close

Gambar 35 Hasil Test-Case 5 (1)

6. Test Case 6 (Random Input)

Cyberpunk 2077 Hacking Minigame Solver
INSTANT BREACH PROTOCOL SOLVER - START CRACKING, SAMURAI

Manual Input Random Input File Input

Number of Tokens: 7

AA
BB
CC
DD
EE
FF
GG

Buffer Size: 5

Matrix Width: 6

Matrix Height: 6

Number of Sequences: 4

Max Sequence Size: 6

Find Solution

Gambar 36 Test-Case 6 (Random Input)

Cyberpunk 2077 Hacking Minigame Solver
INSTANT BREACH PROTOCOL SOLVER - START CRACKING, SAMURAI

Manual Input

Number of Tokens: 7

Initial Input Matrix:

AA	CC	FF	AA	DD	EE
AA	AA	EE	DD	DD	CC
EE	EE	CC	CC	GG	AA
DD	DD	CC	CC	FF	CC
EE	BB	GG	CC	CC	DD
EE	GG	AA	CC	FF	FF

Sequences and Scores:

Sequence 1: DDCCCEGGFFBB - Score: 89

Sequence 2: CCCCDDDDDEEDD - Score: 83

Sequence 3: CCCC - Score: 68

Sequence 4: BBAAABBB - Score: 85

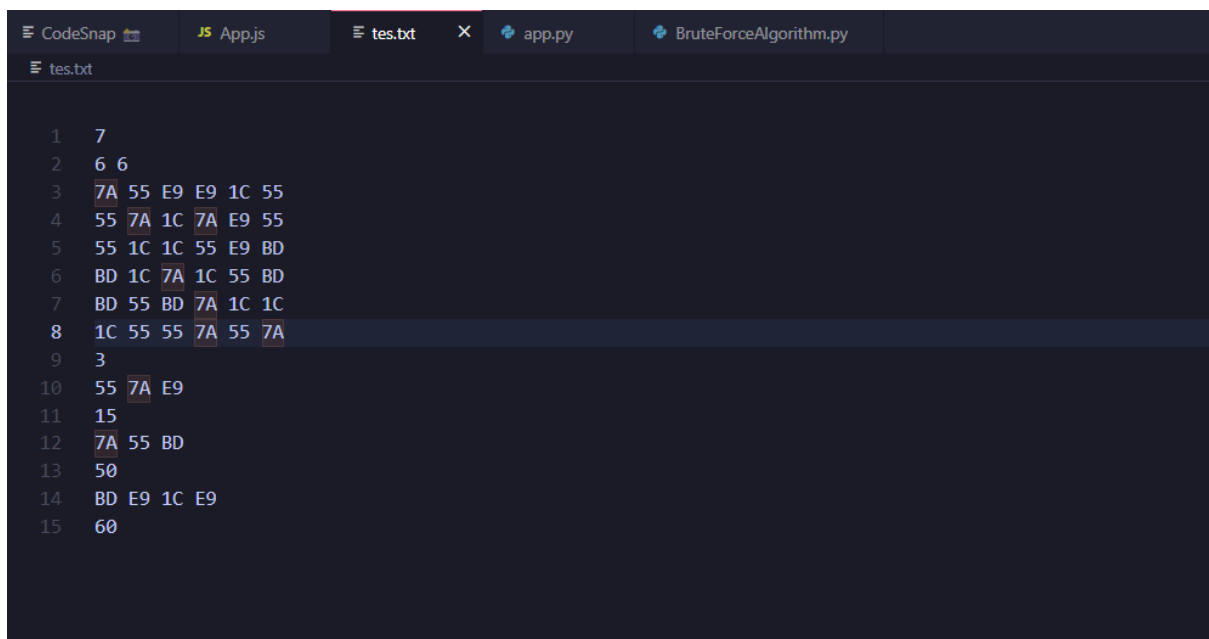
Result Matrix:

Gambar 37 Hasil Test-Case 6 (1)



Gambar 38 Hasil Test-Case 6 (2)

7. Test Case 7 (File Input)



Gambar 39 File Test-Case 7

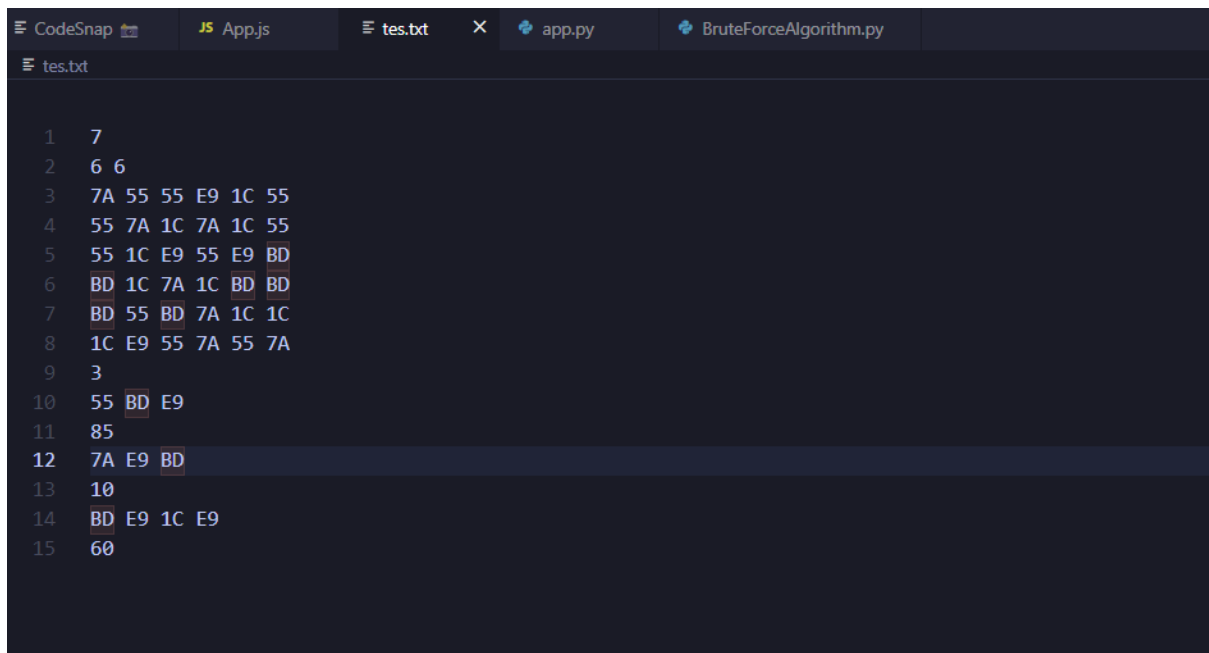


Gambar 40 Hasil Test-Case 7 (1)



Gambar 41 Hasil Test-Case 7 (2)

8. Test Case 8 (File Input)



```
1 7
2 6 6
3 7A 55 55 E9 1C 55
4 55 7A 1C 7A 1C 55
5 55 1C E9 55 E9 BD
6 BD 1C 7A 1C BD BD
7 BD 55 BD 7A 1C 1C
8 1C E9 55 7A 55 7A
9 3
10 55 BD E9
11 85
12 7A E9 BD
13 10
14 BD E9 1C E9
15 60
```

Gambar 42 File Test-Case 8



Cyberpunk 2077 Hacking Minigame Solver

INSTANT BRUTE FORCE

Manual Input

Please input or upload the initial input matrix

Upload File

Initial Input Matrix

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

Sequences and Scores:

Sequence 1: 55BDE9 - Score: 85

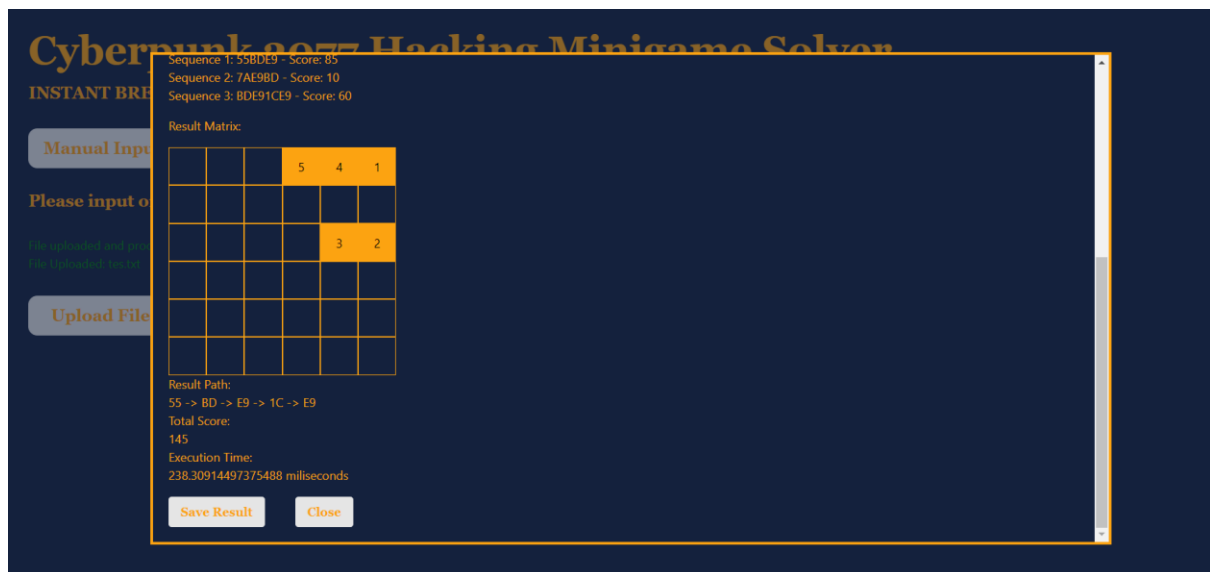
Sequence 2: 7AE9BD - Score: 10

Sequence 3: BDE91CE9 - Score: 60

Result Matrix:

			5	4	1

Gambar 43 Hasil Test-Case 8 (1)



Gambar 44 Hasil Test-Case 8 (2)

BAB V PENUTUP

1. Kesimpulan

Dalam tugas besar IF2211 Strategi Algoritma ini, saya berhasil membuat program penyelesaian dan pencarian solusi optimal Breach Protocol dari game Cyberpunk 2077. Saya berhasil membuat algoritma yang mengimplementasikan konsep Brute-Force namun tetap menjalankan hasil yang optimal. Selain itu, saya juga berhasil menyelesaikan spesifikasi wajib dari tugas ini beserta bonus yakni berupa tampilan GUI dalam bentuk tampilan web.

2. Saran

Beberapa saran yang saya dapat dari pengerjaan pengembangan aplikasi ini adalah sebagai berikut:

1. Untuk kesalahan input , yakni human error, sangat bervariasi sehingga perlu ditulis secara jelas mengenai penanganan serta prosedur yang tepat agar error bisa dihindari.
2. Kemungkinan adanya response time dan input file yang kurang optimal mengakibatkan perlunya meng-refresh jika terdapat kegagalan.
3. Adanya run time yang terkadang bisa terlalu lama tergantung dari ukuran max buffer

BAB VI LAMPIRAN

1. Link Repository

Link github repository: https://github.com/Razark-Y/Tucil1_13522019

2. Program Checkpoint

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	✓	