LAPORAN TUGAS KECIL 1 : BRUTE FORCE

DIBUAT OLEH:

ANDI MARIHOT SITORUS

13522138

1. Algoritma Brute Force

1. Menentukan semua kombinasi cell matriks yang dimulai dari baris pertama yang berisi (baris,kolom,value)
2. Mengeliminasi kombinasi yang tidak sesuai dengan aturan : vertikal, horizontal, vertikal, horizontal, ...
3. Mencocokkan sequence input pada kombinasi untuk menemukan reward terbesar

2. Source Code

```
import random
import time
import os
def print_kombinasi(list):
    for a in list:
        print(a,end=' ')
    print()
def get_kombinasi(matrix, length):
    def generate_kombinasi(path, row, col, length, visited):
        if length == 0:
            kombinasi.append([(row, col, matrix[row][col]) for row, col in
path])
            return

        visited.add((row, col))
        for r in range(len(matrix)):
            if r != row and (r, col) not in visited:
                new_path = path + [(r, col)]
                generate_kombinasi(new_path, r, col, length - 1,
visited.copy())

        for c in range(len(matrix[0])):
            if c != col and (row, c) not in visited:
                new_path = path + [(row, c)]
                generate_kombinasi(new_path, row, c, length - 1,
visited.copy())
```

```
    kombinasi = []
    for col in range(len(matrix[0])):
        start_path = [(0, col)]  # Start only from the first row
        generate_kombinasi(start_path, 0, col, length - 1, set())

    return kombinasi
def read_txt(file_path):
    with open(file_path, 'r') as file:
        size = int(file.readline().strip())

        _, matrix_height = map(int, file.readline().strip().split())

        matrix = []
        for _ in range(matrix_height):
            row = file.readline().strip().split()
            matrix.append(row)


        num_sequences = int(file.readline().strip())
        sequences = []
        rewards = []

        for _ in range(num_sequences):
            sequence = tuple(map(str, file.readline().strip().split()))
            sequences.append(sequence)
            reward = int(file.readline().strip())
            rewards.append(reward)


    return (size,matrix,sequences,rewards)
def generate_data(unique_tokens, size, matrix_width, matrix_height,
num_sequences, max_sequence_length):
    # Generate matrix
    matrix = [[random.choice(unique_tokens) for _ in range(matrix_width)] for
_ in range(matrix_height)]

    # Generate sequences
    sequences = []
    for _ in range(num_sequences):
        sequence_length = random.randint(2, max_sequence_length)
        sequence = tuple([random.choice(unique_tokens) for _ in
range(sequence_length)])
        sequences.append(sequence)

    # Generate rewards for sequences
    rewards = [random.randint(10, 30) for _ in range(num_sequences)]
```

```
    return size, matrix, sequences, rewards
def write_to_txt(file_path, final_reward, final):
    with open(file_path, 'w') as file:
        file.write(str(final_reward)+'\n')
        third_elements = [tup[2] for tup in final]
        for a in third_elements:
            file.write(str(a) + ' ')
        file.write('\n')
        kordinat = [(tup[0], tup[1]) for tup in final]
        for i in kordinat:
            file.write(f'{i[0]}, {i[1]}\n')
        file.write('\n')
        file.write(str(round(runtime*1000))+' ms')
def print_data(unique_tokens, size, matrix, sequences, rewards):
    print("Unique Tokens:", unique_tokens)
    print("Buffer Size:", size)
    print("Matrix:")
    for row in matrix:
        print(" ".join(row))
    print("number of Sequences:", len(sequences))
    for i, seq in enumerate(sequences):
        print("Sequence", i+1, ":", " ".join(seq))
        print( "Reward:", rewards[i])
def matching(buffer, sequences):
    total_seq = 0
    total_reward = 0
    for seq, reward in sequences.items():
        mark = 0
        for i in range(len(buffer) - len(seq) + 1):
            j = 0
            while j < len(seq):
                if seq[j] != buffer[i+j]:
                    break
                j += 1
            if j == len(seq):
                mark += 1
                break

        if mark > 0:
            total_seq += 1
            total_reward += mark * reward

    return total_seq, total_reward

Opsi = input("Apakah anda ingin input melalui file .txt?(y/n) ")
while Opsi != "y" and Opsi != "n":
    Opsi = input("Apakah anda ingin input melalui file .txt?(y/n) ")
if Opsi == "y" :
```

```
    current_folder =
os.path.dirname(os.path.dirname(os.path.abspath(__file__)))
    input_name = input("Masukkan nama file (tambahkan extension .txt): ")
    test = os.path.join(current_folder, 'test')
    file_path = os.path.join(test, input_name)
    size, matrix, sequences, rewards = read_txt(file_path)
else :
    unique_tokens = input("Enter unique tokens (separated by space):
").split()
    size = int(input("Enter buffer size: "))
    matrix_width, matrix_height = map(int, input("Enter matrix size (width
height): ").split())
    num_sequences = int(input("Enter number of sequences: "))
    max_sequence_length = int(input("Enter maximum sequence length: "))

    size, matrix, sequences, rewards = generate_data(unique_tokens, size,
matrix_width, matrix_height, num_sequences, max_sequence_length)

    print("\nGenerated Game Data:")
    print_data(unique_tokens, size, matrix, sequences, rewards)

start = time.time()
sequences = {seq: rew for seq, rew in zip(sequences, rewards)}
kombinasi = get_kombinasi(matrix, size)
filtered_kombinasi = []

#filter untuk yang memenuhi rule
for j in range(len(kombinasi)):

    mark = 0

    for i in range(1, len(kombinasi[j])):

        if i % 2 == 0:

            if kombinasi[j][i][1] == kombinasi[j][i - 1][1]:
                mark+=1
        else:

            if kombinasi[j][i][0] == kombinasi[j][i - 1][0]:
                mark+=1

    if mark > 0 :
        continue

    filtered_kombinasi.append(kombinasi[j])

final = []
```

```
total_seq = 0
final_reward= 0
for seq in filtered_kombinasi:
    buffer = [tup[2] for tup in seq]
    found_seq, total_reward = matching(buffer, sequences)
    if total_reward > final_reward:
        final = seq
        final_reward = total_reward
    if found_seq == len(rewards):
        print('true')
        break
stop = time.time()
runtime = stop-start
print(final_reward)
third_elements = [tup[2] for tup in final]
print_kombinasi(third_elements)
kordinat = [(tup[0], tup[1]) for tup in final]
for i in kordinat:
    print(f'{i[0]}, {i[1]}')
print()
print(round(runtime*1000), 'ms')

Opsi = input("Apakah anda ingin menyimpan hasil di file .txt?(y/n) ")
while Opsi != "y" and Opsi != "n":
    Opsi = input("Apakah anda ingin menyimpan hasil di file .txt?(y/n) ")

if Opsi == "y":
    current_folder =
os.path.dirname(os.path.dirname(os.path.abspath(__file__)))
    input_name = input("Masukkan nama file(tambahkan extension .txt): ")
    test = os.path.join(current_folder, 'test')
    file_path = os.path.join(test, input_name)
        # Write the data to the text file
    write_to_txt(file_path, final_reward, final,)
    print("Data has been written to", file_path)
```

3. Test

1) Input:

test > test.txt

```
7
6 6
7A 55 E9 E9 1C 55
55 7A 1C 7A E9 55
55 1C 1C 55 E9 BD
BD 1C 7A 1C 55 BD
BD 55 BD 7A 1C 1C
1C 55 55 7A 55 7A
3
BD E9 1C
15
BD 7A BD
20
BD 1C BD 55
30
```

Output:

test > result

```
50
7A BD 7A BD 1C BD 55
0, 0
3, 0
3, 2
4, 2
4, 5
2, 5
2, 0

13857 ms
```

2) Input:

```
Apakah anda ingin input melalui file .txt?(y/n) n
Enter unique tokens (separated by space): BD 1C 7A 55 E9
Enter buffer size: 5
Enter matrix size (width height): 5 6
Enter number of sequences: 2
Enter maximum sequence length: 4
```

Output:

```
Generated Game Data:
Unique Tokens: ['BD', 'E9', '1C', '7A', '55']
Buffer Size: 5
Matrix:
7A 55 E9 7A BD
7A 1C 7A 7A 7A
55 55 E9 7A 55
E9 7A 1C BD 7A
55 55 1C BD BD
E9 BD BD E9 7A
number of Sequences: 2
Sequence 1 : BD E9
Reward: 20
Sequence 2 : 7A 55 BD
Reward: 13
33
7A 55 BD BD E9
0, 0
4, 0
4, 3
3, 3
3, 0

56 ms
Apakah anda ingin menyimpan hasil di file .txt?(y/n)
```

3) Input:

```
Apakah anda ingin input melalui file .txt?(y/n) n
Enter unique tokens (separated by space): BD 1C 7A 55 E9
Enter buffer size: 5
Enter matrix size (width height): 3 7
Enter number of sequences: 3
Enter maximum sequence length: 4
```

Output:

```
Generated Game Data:
Unique Tokens: ['BD', 'E9', '1C', '7A', '55']
Buffer Size: 5
Matrix:
7A E9 1C
7A 1C 1C
7A BD 1C
55 7A E9
7A BD E9
1C E9 55
E9 55 7A
number of Sequences: 3
Sequence 1 : 7A 55 7A E9
Reward: 23
Sequence 2 : E9 E9 1C
Reward: 15
Sequence 3 : 1C 7A
Reward: 14
37
1C 7A 55 7A E9
0, 2
6, 2
6, 1
3, 1
3, 2

19 ms
Apakah anda ingin menyimpan hasil di file .txt?(y/n)
```

4) Input:

```
Apakah anda ingin input melalui file .txt?(y/n) n
Enter unique tokens (separated by space): BD 1C 7A 55 E9
Enter buffer size: 6
Enter matrix size (width height): 6 4
Enter number of sequences: 3
Enter maximum sequence length: 5
```

Output:

```
Generated Game Data:
Unique Tokens: ['BD', 'E9', '1C', '7A', '55']
Buffer Size: 6
Matrix:
55 1C 7A 7A 7A 1C
BD 1C 55 7A 55 E9
1C BD BD E9 E9 7A
7A 55 7A 1C 7A 1C
number of Sequences: 3
Sequence 1 : 7A E9 7A 1C 1C
Reward: 25
Sequence 2 : 1C 1C 1C 7A 55
Reward: 22
Sequence 3 : 55 BD BD
Reward: 22
25
7A E9 7A 1C 1C 1C
0, 3
2, 3
2, 5
0, 5
0, 1
1, 1

353 ms
Apakah anda ingin menyimpan hasil di file .txt?(y/n)
```

5) Input:

```
Apakah anda ingin input melalui file .txt?(y/n) n
Enter unique tokens (separated by space): BD 1C 7A 55 E9
Enter buffer size: 6
Enter matrix size (width height): 6 4
Enter number of sequences: 3
Enter maximum sequence length: 5
```

Output:

```
Generated Game Data:
Unique Tokens: ['BD', 'E9', '1C', '7A', '55']
Buffer Size: 6
Matrix:
E9 BD 7A 55 1C BD
1C E9 55 1C 55 E9
7A BD 55 55 1C 55
1C 1C 7A 55 1C E9
number of Sequences: 3
Sequence 1 : E9 BD BD
Reward: 27
Sequence 2 : E9 BD 55
Reward: 17
Sequence 3 : 7A 7A BD 7A
Reward: 30
27
E9 1C E9 BD BD E9
0, 0
1, 0
1, 1
0, 1
0, 5
1, 5

348 ms
Apakah anda ingin menyimpan hasil di file .txt?(y/n)
```

6) Input:

```
Apakah anda ingin input melalui file .txt?(y/n) n
Enter unique tokens (separated by space): A B C D E F
Enter buffer size: 7
Enter matrix size (width height): 6 6
Enter number of sequences: 4
Enter maximum sequence length: 5
```

Output:

```
Generated Game Data:
Unique Tokens: ['A', 'B', 'C', 'D', 'E', 'F']
Buffer Size: 7
Matrix:
D B F E D B
F A D B D C
C F E F A F
A B A A F C
E A C B E C
B D E C F B
number of Sequences: 4
Sequence 1 : D A
Reward: 23
Sequence 2 : D D
Reward: 23
Sequence 3 : A D
Reward: 26
Sequence 4 : B A B E B
Reward: 29
72
D C A D D A A
0, 0
2, 0
2, 4
1, 4
1, 2
3, 2
3, 0

13506 ms
Apakah anda ingin menyimpan hasil di file .txt?(y/n)
```

4. Pranala Repository

https://github.com/MelonSeed9/Tucil1_13522138.git

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		✓