LAPORAN TUGAS KECIL 1
IF2211 STRATEGI ALGORITMA

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

Disusun Oleh:
Muhammad Dzaki Arta 13522149

Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung
Semester II tahun 2023/2024

BAB I
DESKRIPSI MASALAH

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 bermainnya adalah

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

BAB II
Analisis Algoritma

Brute force adalah sebuah pendekatan langsung(straight forward) 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).

Pada program ini saya memakai bahasa pemrograman python, pada bagian awal program akan menanyakan apakah pengguna ingin menerima input dari file atau dari keyboard (CLI). jika pengguna memilih menerima input dari file maka program akan secara otomatis pergi ke folder test dan mengambil file dari folder itu dengan nama file berasal dari masukan pengguna.

Apabila pengguna memilih input dari keyboard maka program akan menanyakan input seperti besar buffer, tinggi dan lebar matrix, banyak sekuens, serta sekuens dan bobot sekuens.Program juga akan meminta masukan matrix dari pengguna akan tetapi pengguna dapat memilih memasukan matrix secara manual atau secara acak.

Setelah semuanya sudah terbaca pada program, maka program akan menampilkan semua yang diminta kedalam layar. dan akan menggunakan algoritma Brute Force untuk memecahkan masalah tersebut.

Pertama program akan mencari semua kemungkinan sekuens yang terdapat pada matrix sesuai besar buffer. program ini disimpan pada fungsi “findAllSolution”. Ini merupakan fungsi rekursif yang iterasi nya akan selalu berubah setiap rekursi sesuai dengan kondisi yang ada. Untuk memastikan tidak ada 2 atau lebih token yang sama pada 1 sekuens maka saya menggunakan matrix boolean representasi dari matrix token. Apabila sebuah token sudah di ambil maka representasi token pada matrix boolean akan bernilai true dan token pada posisi yang sama tidak akan bisa diambil lagi. Semua hasil yang didapat akan disimpan dalam sebuah array.

Kemudian untuk menghitung solusi terbaik pada array yang dihasilkan, saya menggunakan algiritma Brute Force yaitu dengan me-iterasi setiap sekuens pada array yang dihasilkan dari fungsi “findAllSolution” dan kemudian menghitung poin setiap sekuen sekaligus mencari sekuen dengan nilai terbaik.

Untuk menghitung koordinat pada sekuen saya juga menggunakan algoritma Brute Force yaitu menghitung setiap kemungkinan posisi token yang ada pada matrix dan menghasilkan posisi yang sesuai dengan token. Algoritma ini disimpan pada fungsi rekursif yang saya namai “findCoordinate”.

Setelah program mendapatkan sekuen terbaik, poin, serta koordinatnya. Maka program akan menampilkan semuanya dan menanyakan apakah user ingin menyimpan solusi tersebut, jika iya maka program akan menanyakan judul kepada user dan program akan menyimpannya sebagai file “.txt”.

Bab III
Source Code

Saya memiliki 3 file utama yaitu

1. file_operation.py yang berisi fungsi operasi file seperti readFile dan saveFile
2. command.py yang berisi semua fungsi utama dalam program ini
3. main.py yang berisi satu kesatuan kode program yang disusun dari fungsi dan prosedur yang berasal dari file_operation.py dan command.py

adapun saya menggunakan 2 library python yaitu

1. random, library ini saya gunakan untuk membuat matrix acak
2. time, library ini saya gunakan untuk menghitung waktu eksekusi program

isi dari file_operation.py

```python
def readFile(filename):
    # Membaca file txt
    filename = "../test/"+ filename  +".txt"# Gantilah 'nama_file.txt' dengan nama file yang sesuai
    with open(filename, 'r') as file:
        lines = file.readlines()

    # Mengonversi isi file menjadi array
    array_data = []
    for line in lines:
        line = line.strip()
        if line:
            array_data.append(line.split())

    # Membaca buffer size dari file
    buffer_size = int(array_data[0][0])

    # Membaca ukuran matrix dari file
    matrix_width, matrix_height = int(array_data[1][0]), int(array_data[1][1])
    # Membaca matrix dari file
    matrix = [[0 for j in range(matrix_width)]for i in range (matrix_height)]

    for i in range(matrix_height):
        for j in range(matrix_width):
            matrix[i][j] = array_data[i+2][j]

    # Membaca banya sequence dari file
    number_of_sequence = int(array_data[matrix_height+2][0])

    # Mengkonversi sequence ke dalam array
    array_of_sequence = [array_data[matrix_height+3+i] for i in range(0, 2*number_of_sequence, 2)]
    array_of_value = [int(array_data[matrix_height+3+i][0]) for i in range(1, 2*number_of_sequence, 2)]

    return buffer_size, matrix_width, matrix_height, number_of_sequence, array_of_sequence, array_of_value, matrix
```

```python
def saveFile(filename,value,sequence,array_of_coodinate,time):
    try:
        with open("../test/" + filename + ".txt", 'w') as file:
            file.write(f"{value}\n")
            for i in sequence:
                file.write(f"{i} ")
            file.write("\n")
            for i in array_of_coodinate:
                file.write(f"{i}\n")
            file.write("\n")
            file.write(f"{time}ms\n\n")
        print(f"berhasil disimpan kedalam {filename}.txt")
    except Exception as e:
        print(f"Terjadi kesalahan: {str(e)}")
```

isi dari file command.py

```python
import random

def findAllSolution(buffer_size,matrix_width,matrix_height,matrix,buffer_solutions, temp,boolean_matrix, number_of_elmt,axis):
    if number_of_elmt == buffer_size :
        buffer_solutions.append(list(temp))
    elif number_of_elmt == 0 :
        for i in range(matrix_width):
            if(boolean_matrix[0][i] == False):
                boolean_matrix[0][i] = True
                findAllSolution(buffer_size,matrix_width,matrix_height,matrix,buffer_solutions,[matrix[i][0]],boolean_matrix,number_of_el
                boolean_matrix[0][i] = False
    elif number_of_elmt%2 == 1:
        for i in range(matrix_height):
            if(boolean_matrix[i][axis] == False):
                boolean_matrix[i][axis] = True
                temp.append(matrix[i][axis])
                findAllSolution(buffer_size,matrix_width,matrix_height,matrix,buffer_solutions,temp,boolean_matrix,number_of_elmt+1,i)
                temp.pop()
                boolean_matrix[i][axis] = False

    else:
        for i in range(matrix_width):
            if(boolean_matrix[axis][i] == False):
                boolean_matrix[axis][i] = True
                temp.append(matrix[axis][i])
                findAllSolution(buffer_size,matrix_width,matrix_height,matrix,buffer_solutions,temp,boolean_matrix,number_of_elmt+1,i)
                temp.pop()
                boolean_matrix[axis][i] = False
```

```python
def is_subarray(subarray, array):
    len_array = len(array)
    len_subarray = len(subarray)

    if len_subarray > len_array:
        return False

    for i in range(len_array - len_subarray + 1):
        if array[i:i + len_subarray] == subarray:
            return True

    return False
```

```python
def hitungPoint(number_of_sequence,array_of_sequence,array_of_value,buffer_solutions):
    best_seq = buffer_solutions[0]
    best_val = 0
    for sol in  buffer_solutions:
        val = 0
        for i in range (number_of_sequence):
            if is_subarray(array_of_sequence[i],sol):
                val += array_of_value[i]
        if(val > best_val):
            best_seq = sol
            best_val = val
    return best_seq, best_val
```

```python
def findCoordinate(buffer_size,matrix_width,matrix_height,matrix,result, array_of_coor,sequence,index_elmt,axis):
    if len(array_of_coor) == buffer_size:
        result.clear()
        for el in array_of_coor:
            result.append(el)
    elif index_elmt== 0:
        array_of_coor = []
        for i in range(matrix_width):
            if matrix[0][i] == sequence[0]:
                coor = (i+1,1)
                array_of_coor.append(coor)
                findCoordinate(buffer_size,matrix_width,matrix_height,matrix,result,array_of_coor,sequence,index_elmt+1,i)
                array_of_coor.pop()
    elif index_elmt % 2 == 1 and index_elmt <= buffer_size:
        for i in range(matrix_height):
            if matrix[i][axis] == sequence[index_elmt]:
                coor = (axis+1,i+1)
                array_of_coor.append(coor)
                findCoordinate(buffer_size,matrix_width,matrix_height,matrix,result,array_of_coor,sequence,index_elmt+1,i)
                array_of_coor.pop()
    elif index_elmt % 2 == 0 and index_elmt <= buffer_size:
        for i in range(matrix_width):
            if matrix[axis][i] == sequence[index_elmt]:
                coor = (i+1,axis+1)
                array_of_coor.append(coor)
                findCoordinate(buffer_size,matrix_width,matrix_height,matrix,result,array_of_coor,sequence,index_elmt+1,i)
                array_of_coor.pop()
```

```python
def keyboardInput():
    buffer_size = int(input("Masukan besar buffer : "))
    matrix_height = int(input("Masukan Tinggi matrix: "))
    matrix_width = int(input("Masukan lebar matrik: "))
    matrix = []
    masukan = input("Apakah anda ingin menghasilkan matrix secara acak? (y/n) : ")
    valid = False
    while(valid == False):
        if(masukan == 'y'):
            matrix = randomMatrix(matrix_height,matrix_width)
            valid = True
        elif(masukan == 'n'):
            valid = True
            # Membaca input matriks dari pengguna
            print("Masukan matrix (Setiap elemen dipisahkan dengan spasi)")
            for i in range (matrix_height):
                matrix_input = input()
                temp = list(map(str, matrix_input.split()))
                valid = False
                while (valid == False):
                    if (len(temp) == matrix_width):
                        valid = True
                        matrix.append(temp)
                    else:
                        print("lebar matrix harus {matrix_width} silahkan masukan kembali")
                        matrix_input = input()
                        temp = list(map(str, matrix_input.split()))
                        valid = False
        else:
            masukan = input("Masukan hanya boleh (y/n) silahkan masukan kembali: ")
```

```python
        else:
            masukan = input("Masukan hanya boleh (y/n) silahkan masukan kembali: ")
            valid = False

    number_of_sequence = int(input("Masukan banyak sequence yang anda mau: "))

    list_of_sequence = []
    list_of_value = []
    for i in range(number_of_sequence):
        seq = input(f"masukan sequence ke-{i+1}: ")
        list_of_sequence.append(list(map(str,seq.split())))
        val = int(input("Masukan nilai yang anda mau dari sequence diatas: "))
        list_of_value.append(val)

    return buffer_size,matrix_width,matrix_height, number_of_sequence,list_of_sequence,list_of_value,matrix
```

```
def randomMatrix(height, width):
    elmt = ['1C', '55', 'E9', '7A', 'BD']
    matrix = []

    for i in range(height):
        row = [random.choice(elmt) for j in range(width)]
        matrix.append(row)

    print("Matrix :")
    for i in matrix:
        print(i)

    return matrix
```

isi dari file main.py

```
import command
import file_operation
import time

print(f"Selamat datang \nsilahkan pilih input yang anda mau :")
print("1. Keyboard input")
print("2. File input")

masukan = int(input("Silahkan masukan pilihan anda : "))
valid = False
while (valid == False):
    if(masukan == 1):
        buffer_size,matrix_width,matrix_height, number_of_sequence,array_of_sequence,array_of_value,matrix = command.keyboardInput()
        valid = True
    elif(masukan == 2):
        valid = True
        masukan = input("Silahkan masukan nama file anda (tanpa diakhiri .txt): ")
        buffer_size,matrix_width,matrix_height, number_of_sequence,array_of_sequence,array_of_value,matrix = file_operation.readFile(masukan)
    else:
        masukan = int(input(("Masukan anda tidak valid silahkan masukan kembali: ")))
        valid = False
```

```
print()
print()
print(f"besar buffer = {buffer_size}")
print(f"lebar matrix = {matrix_width}")
print(f"tinggi matrix = {matrix_height}")
print(f"matrix :")
for i in matrix:
    print(i)
print(f"number of sequence = {number_of_sequence}")
print(f"array of sequence = {array_of_sequence}")
print(f"array of value = {array_of_value}")
```

```
buffer_solutions = []
temp = []
boolean_matrix = [[False for j in range(matrix_width)]for i in range (matrix_height)]

start = time.time()
command.findAllSOlution(buffer_size,matrix_width,matrix_height,matrix,buffer_solutions, temp,boolean_matrix, 0,0)

best_seq, best_val = command.hitungPoint(number_of_sequence,array_of_sequence,array_of_value,buffer_solutions)
```

```
print("\n\n")
print(f"nilainya adalah  {best_val}")
print(f"sequence terbaik adalah : {best_seq}")
array_of_coordinate = []
temp = []
command.findCoordinate(buffer_size,matrix_width,matrix_height,matrix,array_of_coordinate, temp,best_seq,0,0)
end = time.time()
duration = round((end - start)*10**3)

print(f"koordinat dari sequencenya adalah ")
for coor in array_of_coordinate:
    print(coor)
print()
print()
print(f"{duration}ms")
```

```
masukan = input("Apakah anda ingin menyimpan solusi ini? (y/n) : ")
valid = False
while(valid == False):
    if (masukan == 'y'):
        valid = True
        masukan = input("silahkan masukan nama file yang ingin disimpan (tanpa menggunakan .txt): ")
        file_operation.saveFile(masukan,best_val,best_seq,array_of_coordinate,duration)
    elif(masukan == 'n'):
        valid = True
        # do nothing
    else :
        masukan = input(("Masukan anda tidak sesuai silahkan masukan kembali: "))
```

BAB IV
TEST CASE

4.1 Masukan dari file.txt

Isi dari file.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
```

Tampilan Terminal

```
Selamat datang
silahkan pilih input yang anda mau :
1. Keyboard input
2. File input
Silahkan masukan pilihan anda : 2
Silahkan masukan nama file anda (tanpa diakhiri .txt): file
```

```
besar buffer = 7
lebar matrix = 6
tinggi matrix = 6
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']
number of sequence = 3
array of sequence = [['BD', 'E9', '1C'], ['BD', '7A', 'BD'], ['BD', '1C', 'BD', '55']]
array of value = [15, 20, 30]



nilainya adalah  50
sequence terbaik adalah : ['7A', 'BD', '7A', 'BD', '1C', 'BD', '55']
koordinat dari sequencenya adalah
(1, 1)
(1, 4)
(3, 4)
(3, 5)
(6, 5)
(6, 4)
(5, 4)


420ms
Apakah anda ingin menyimpan solusi ini? (y/n) :
```

4.2 Masukan dari keyboard

```
Selamat datang
silahkan pilih input yang anda mau :
1. Keyboard input
2. File input
Silahkan masukan pilihan anda : 1
Masukan besar buffer : 4
Masukan Tinggi matrix: 3
Masukan lebar matrik: 3
Apakah anda ingin menghasilkan matrix secara acak? (y/n) : a
Masukan hanya boleh (y/n) silahkan masukan kembali: n
Masukan matrix (Setiap elemen dipisahkan dengan spasi)
7A 55 E9
55
lebar matrix harus {matrix_width} silahkan masukan kembali
55 7A 1C
55 1C 1C
Masukan banyak sequence yang anda mau: 2
masukan sequence ke-1: 7A 55
Masukan nilai yang anda mau dari sequence diatas: 20
masukan sequence ke-2: 55 1C 1C
Masukan nilai yang anda mau dari sequence diatas: 30
```

```
besar buffer = 4
lebar matrix = 3
tinggi matrix = 3
matrix :
['7A', '55', 'E9']
['55', '7A', '1C']
['55', '1C', '1C']
number of sequence = 2
array of sequence = [['7A', '55'], ['55', '1C', '1C']]
array of value = [20, 30]



nilainya adalah  50
sequence terbaik adalah : ['7A', '55', '1C', '1C']
koordinat dari sequencenya adalah
(1, 1)
(1, 3)
(3, 3)
(3, 2)


0ms
Apakah anda ingin menyimpan solusi ini? (y/n) : n
```

4.3 Masukan dari keyboard tetapi acak

```
Selamat datang
silahkan pilih input yang anda mau :
1. Keyboard input
2. File input
Silahkan masukan pilihan anda : 1
Masukan besar buffer : 4
Masukan Tinggi matrix: 3
Masukan lebar matrik: 3
Apakah anda ingin menghasilkan matrix secara acak? (y/n) : y
Matrix :
['E9', 'E9', 'BD']
['1C', 'BD', '7A']
['E9', 'BD', '7A']
Masukan banyak sequence yang anda mau: 2
masukan sequence ke-1: 7A 7A
Masukan nilai yang anda mau dari sequence diatas: 10
masukan sequence ke-2: BD BD
Masukan nilai yang anda mau dari sequence diatas: 10
```

```
besar buffer = 4
lebar matrix = 3
tinggi matrix = 3
matrix :
['E9', 'E9', 'BD']
['1C', 'BD', '7A']
['E9', 'BD', '7A']
number of sequence = 2
array of sequence = [['7A', '7A'], ['BD', 'BD']]
array of value = [10, 10]



nilainya adalah  10
sequence terbaik adalah : ['E9', '1C', 'BD', 'BD']
koordinat dari sequencenya adalah
(1, 1)
(1, 2)
(2, 2)
(2, 3)


0ms
Apakah anda ingin menyimpan solusi ini? (y/n) : n
```

4.4 Masukan dari file apabila ada sequence nilai negatif

```
Selamat datang
silahkan pilih input yang anda mau :
1. Keyboard input
2. File input
Silahkan masukan pilihan anda : 2
Silahkan masukan nama file anda (tanpa diakhiri .txt): file


besar buffer = 7
lebar matrix = 6
tinggi matrix = 6
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']
number of sequence = 3
array of sequence = [['BD', 'E9', '1C'], ['BD', '7A', 'BD'], ['BD', '1C', 'BD', '55']]
array of value = [-15, -20, 30]
```

```
nilainya adalah  30
sequence terbaik adalah : ['7A', '55', '1C', 'BD', '1C', 'BD', '55']
koordinat dari sequencenya adalah
(1, 1)
(1, 3)
(3, 3)
(3, 5)
(6, 5)
(6, 4)
(5, 4)


461ms
Apakah anda ingin menyimpan solusi ini? (y/n) : y
silahkan masukan nama file yang ingin disimpan (tanpa menggunakan .txt): test1
```

4.5 Masukan dari keyboard apabila ada sequence dengan nilai negatif

```
Selamat datang
silahkan pilih input yang anda mau :
1. Keyboard input
2. File input
Silahkan masukan pilihan anda : 1
Masukan besar buffer : 4
Masukan Tinggi matrix: 3
Masukan lebar matrik: 3
Apakah anda ingin menghasilkan matrix secara acak? (y/n) : N
Masukan hanya boleh (y/n) silahkan masukan kembali: n
Masukan matrix (Setiap elemen dipisahkan dengan spasi)
7A 55 E9
55 7A 1C
55 1C 1C
Masukan banyak sequence yang anda mau: 2
masukan sequence ke-1: E9 1C 1C
Masukan nilai yang anda mau dari sequence diatas: -20
masukan sequence ke-2: 1C 1C
Masukan nilai yang anda mau dari sequence diatas: 20
```

```
besar buffer = 4
lebar matrix = 3
tinggi matrix = 3
matrix :
['7A', '55', 'E9']
['55', '7A', '1C']
['55', '1C', '1C']
number of sequence = 2
array of sequence = [['E9', '1C', '1C'], ['1C', '1C']]
array of value = [-20, 20]



nilainya adalah  20
sequence terbaik adalah : ['7A', '55', '1C', '1C']
koordinat dari sequencenya adalah
(1, 1)
(1, 3)
(3, 3)
(3, 2)


2ms
Apakah anda ingin menyimpan solusi ini? (y/n) : y
silahkan masukan nama file yang ingin disimpan (tanpa menggunakan .txt): test2
```

4.6 Masukan Acak tetapi memiliki sekuens bernilai negatif

```
Selamat datang
silahkan pilih input yang anda mau :
1. Keyboard input
2. File input
Silahkan masukan pilihan anda : 1
Masukan besar buffer : 7
Masukan Tinggi matrix: 6
Masukan lebar matrik: 6
Apakah anda ingin menghasilkan matrix secara acak? (y/n) : y
Matrix :
['1C', '55', '55', 'E9', '7A', '1C']
['E9', '55', 'E9', '7A', '55', 'BD']
['E9', '7A', '55', '7A', '7A', '1C']
['E9', 'E9', '55', 'BD', 'BD', '7A']
['1C', '7A', '1C', '7A', '1C', '55']
['BD', 'BD', 'E9', 'E9', 'E9', 'BD']
Masukan banyak sequence yang anda mau: 3
masukan sequence ke-1: 55 55 1C
Masukan nilai yang anda mau dari sequence diatas: -10
masukan sequence ke-2: E9 55 55
Masukan nilai yang anda mau dari sequence diatas: 20
masukan sequence ke-3: BD 7A 1C
Masukan nilai yang anda mau dari sequence diatas: 30
```

```
besar buffer = 7
lebar matrix = 6
tinggi matrix = 6
matrix :
['1C', '55', '55', 'E9', '7A', '1C']
['E9', '55', 'E9', '7A', '55', 'BD']
['E9', '7A', '55', '7A', '7A', '1C']
['E9', 'E9', '55', 'BD', 'BD', '7A']
['1C', '7A', '1C', '7A', '1C', '55']
['BD', 'BD', 'E9', 'E9', 'E9', 'BD']
number of sequence = 3
array of sequence = [['55', '55', '1C'], ['E9', '55', '55'], ['BD', '7A', '1C']]
array of value = [-10, 20, 30]



nilainya adalah  50
sequence terbaik adalah : ['1C', 'E9', '55', '55', 'BD', '7A', '1C']
koordinat dari sequencenya adalah
(1, 1)
(1, 3)
(3, 3)
(3, 4)
(5, 4)
(5, 3)
(6, 3)
```

```
515ms
Apakah anda ingin menyimpan solusi ini? (y/n) : y
silahkan masukan nama file yang ingin disimpan (tanpa menggunakan .txt): test3
```

BAB V
REFERENSI

Munir, R. (2024). Algoritma Brute Force (*Bagian 1)*. Diakses pada 12 Februari 2024 dari https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2021-2022/Algoritma-Brute-Force-(2022)-Bag1.pdf

Link ke repository Github:
https://github.com/TuanOnta/Tucil1_13522149

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		✅