LAPORAN TUGAS BESAR
IF2211 STRATEGI ALGORITMA

Disusun Oleh :
Maria Flora Renata Siringoringo (13522010)

PROGRAM STUDI TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
2023

Daftar Isi

Daftar Isi **2**
Bab 1 : Teori Dasar **3**
1.1. HyperText Markup Language (HTML) 3
1.2. Pushdown Automata (PDA) 3
Bab 2 : Hasil PDA **4**
2.2.1. State (Q) 4
2.2.2. Alfabet Input (∑) 4
2.2.3. Simbol Stack (Γ) 4
2.2.4. State Awal (q0) 4
2.2.5. Simbol Stack Awal (Z) 4
2.2.6. Himpunan State Penerima (F) 4
2.2.7. Fungsi Transisi (δ) 4
Bab 3 : Implementasi dan Pengujian **11**
3.1 Test Case 1 11
3.2 Test Case 2 11
3.3 Test Case 3 12
3.4 Test Case 4 13
3.5 Test Case 5 13
3.6 Test Case 6 14
3.7 Test Case 7 14
Lampiran **16**
Link Repository 16
Link Diagram State 16
Pembagian Tugas 16

Bab 1 : Algoritma

1.1. *Class*

Sekuens, sel matriks token, dan sekuens hasil dijadikan tiga *class* yang berbeda untuk memudahkan pengaksesan informasi terkait setiap objek tersebut. *Class* sel matriks terdiri dari string token, boolean yang merepresentasikan sudah atau belumnya sel tersebut dikunjungi, dan koordinat kolom dan baris sel tersebut. *Class* sekuens terdiri dari sekuens dalam bentuk string dan nilai poin yang didapatkan jika menyelesaikan sekuens tersebut. *Class* sekuens hasil terdiri dari sekuens sel yang dikunjungi, koordinat masing-masing sel, dan total poin yang didapatkan oleh sekuens tersebut.

1.2. *Input*

Input dapat dilakukan dengan dua cara, yaitu melalui file txt dan CLI. Input melalui file txt dapat dilaksanakan dengan file browser atau dengan mengetik nama file. Jika melalui input nama file, file harus diletakkan di folder test dan tidak perlu mengetik ekstensi file. Input melalui CLI akan meminta user untuk menginput jumlah token unik, token-token unik, ukuran buffer jawaban, ukuran matriks token, jumlah sekuens dan jumlah maksimal token dalam setiap sekuens. Program kemudian akan menyusun matriks sesuai input dengan isi token di setiap sel *random*. Sekuens juga akan dibentuk secara *random* dengan range poin dari -100 sampai 100.

1.2. *Brute Force*

Setelah matriks dan sekuens siap, program akan men-*generate* semua kemungkinan jawaban dengan menggunakan rekursi dan traversal. Program melakukan traversal dari indeks ke 0 sampai indeks terakhir di baris paling atas, memanggil fungsi *generator*, menandai sel yang sudah dikunjungi (mengubah nilai boolean pada *class* sel menjadi true), lalu melakukan traversal untuk semua indeks di baris atau kolom yang sedang ditelusuri, lalu memanggil fungsi *generator* lagi secara rekursif. Saat sampai ke basis (panjang sekuens yang tergenerasi sama dengan panjang buffer maksimum), program akan mengubah *array* dari token yang sudah dilalui ke dalam bentuk sekuens hasil, lalu menambahkannya ke *array* jawaban.

Saat semua kemungkinan jawaban terbentuk dan dimasukkan ke *array*, program akan mengiterasi melalui setiap jawaban tersebut untuk mengecek keberadaan setiap sekuens berpoin di dalamnya. Program akan mencari jawaban yang menghasilkan poin terbanyak.

1.2. *Output*

Program menampilkan pesan apakah jawaban dapat ditemukan atau tidak. Jika iya program akan menampilkan total poin maksimal, sekuens jawaban, koordinat setiap token, dan waktu proses. Program akan menanyakan apakah user ingin menyimpan jawaban ke file, jika iya, program akan menuliskan jawaban ke file txt sesuai nama yang diinput oleh user. File txt akan disimpan di subfolder saved di dalam folder test

Bab 2 : Source program

```
import tkinter as tk
from tkinter import filedialog
import os
import time
import random

class Cell:
    def __init__(self, token, row, col):
        self.picked = False
        self.token = token
        self.row = row
        self.col = col

class Sequence:
    def __init__(self, seq, reward):
        self.seq = seq
        self.reward = reward

class Solves:
    def __init__(self, token_arr):
        self.sum_reward = 0
        str = ""
        for i in token_arr:
            str += i.token + " "
        self.str = str
        coordinates = []
        for i in token_arr:
            coordinates.append(i.col + ", " + i.row)
        self.coordinates = coordinates
    def add(self):
        global solver_arr
        solver_arr.append(self)

def readfromtxt(arr):
    print("1. Menggunakan file browser\n2. Input nama file")
    metode2 = int(input())
    while (metode2!=1 and metode2!=2):
```

```
        print("1. Menggunakan file browser\n2. Input nama file")
        metode2=int(input())
    if (metode2==1):
        root = tk.Tk()
        root.withdraw()
        pth = filedialog.askopenfilename()
    else:
        print("Masukkan nama file txt!")
         pth = os.path.dirname(os.path.abspath(__file__)) + "\\..\\test\\"
+ input() + ".txt"
    setting = open(pth,"r")
    global buffersize; buffersize = int(setting.readline())
    dimension = setting.readline().split()
    global height; height = int(dimension[1])
    global width; width = int(dimension[0])
     mat = [[Cell("",-1,-1) for _ in range (int(dimension[1]))] for _ in
range (int(dimension[0]))]
    for i in range (int(dimension[1])):
        tokens = setting.readline().split()
        for j in range (int(dimension[0])):
            temp = Cell(tokens[j], str(i+1), str(j+1))
            mat[i][j] = temp
    numseqs = int(setting.readline())
    for i in range (numseqs):
        tempstr = setting.readline().strip('\n')
        tempreward = int(setting.readline())
        tempseq = Sequence(tempstr,tempreward)
        arr.append(tempseq)
    return mat

def readinput(arr):
    print("Jumlah token unik:")
    unique = int(input())
    print("Masukkan token:")
    tokens = input().split()
    print("Masukkan ukuran buffer:")
    global buffersize; buffersize = int(input())
    print("Masukkan ukuran matriks (row col):")
    dimension = input().split()
    global height; height = int(dimension[0])
```

```
    global width; width = int(dimension[1])
     mat = [[Cell("",-1,-1) for _ in range (int(dimension[1]))] for _ in
range (int(dimension[0]))]
    for i in range (int(dimension[0])):
        for j in range (int(dimension[1])):
            temp = Cell(random.choice(tokens), str(i+1), str(j+1))
            mat[i][j] = temp
    print("Masukkan jumlah sekuens:")
    numseqs = int(input())
    print("Masukkan ukuran maksimal sekuens:")
    maxlength = int(input())
    for i in range (numseqs):
        templst = random.sample(tokens, random.randint(1, maxlength))
        tempstr = " ".join(templst)
        tempreward = random.randint(-100, 100)
        tempseq = Sequence(tempstr,tempreward)
        arr.append(tempseq)
    return mat

def generator(generated, length, buffermax, isHorizontal, matrix, m, n,
row, col):
    matrix[row][col].picked = True
    generated.append(matrix[row][col])

    if (length<buffermax-1):
        if (isHorizontal):
           for i in range (n):
               if (matrix[row][i].picked == False):
                           generator(generated, length+1, buffermax, not
isHorizontal, matrix, m, n, row, i)
        else:
           for i in range (m):
               if (matrix[i][col].picked == False):
                           generator(generated, length+1, buffermax, not
isHorizontal, matrix, m, n, i, col)
    elif (length==buffermax-1):
        temp = Solves(generated)
        temp.add()

    matrix[row][col].picked = False
```

```
    if (len(generated)>0):
        del generated[-1]

def compare(solver, sequence):
    maxscore = 0
    idx = -1
    for i in range (len(solver)):
        for j in sequence:
            if (j.seq in solver[i].str):
                solver[i].sum_reward += j.reward
        if ((solver[i].sum_reward > maxscore)):
            idx = i
            maxscore = solver[i].sum_reward
    return idx

buffersize = 0
maxscore = 0
sequence_arr = []
solver_arr = []
height = 0
width = 0

print("Selamat datang di Breach Solver!")
print("Pilih metode input\n1. File txt\n2. CLI")
metode = int(input())
while (metode!=1 and metode!=2):
    print("Pilih metode input\n1. File txt\n2. CLI")
    metode=int(input())
if (metode==1):
    mat = readfromtxt(sequence_arr)
else:
    mat = readinput(sequence_arr)

print("Matriks yang akan dikerjakan:")
for i in range (height):
    for j in range (width):
        print (mat[i][j].token, end=" ")
    print("")
print("Sekuens:")
for i in sequence_arr:
```

```
    print(i.seq, end="      ")
    print(i.reward, end=" "); print("poin")

start=time.time()
for i in range (width):
    generator([], 0, buffersize, False, mat, height, width, 0, 0)
idx_sol = compare(solver_arr, sequence_arr)
end=time.time()

if (idx_sol == -1):
    print("Tidak ada solusi")
else:
    print("Solusi ditemukan!")
    print(solver_arr[idx_sol].sum_reward)
    print(solver_arr[idx_sol].str)
    for i in solver_arr[idx_sol].coordinates:
        print(i)
    print(round((end-start)*1000), "ms")

print("Simpan hasil ke file txt? (y/n)")
choice = input()
while (choice!="y" and choice!="n"):
    print("Simpan hasil ke file txt? (y/n)")
    choice = input()
if (choice=="y"):
      if  (not  os.path.isdir(os.path.dirname(os.path.abspath(__file__))  +
"\\saved")):
        os.mkdir(os.path.dirname(os.path.abspath(__file__)) + "\\saved")
    print("Masukkan nama file:")
     filename = os.path.dirname(os.path.abspath(__file__)) + "\\saved\\" +
input() + ".txt"
    file = open(filename, "w")
    file.write(str(solver_arr[idx_sol].sum_reward) + "\n")
    file.write(solver_arr[idx_sol].str + "\n")
    for i in solver_arr[idx_sol].coordinates:
        file.write(i + "\n")
    file.write(str(round((end-start)*1000)) + " ms")
    file.close()
    print("File berhasil disimpan!")
```

Bab 3 : Implementasi dan Pengujian

3.1 Test Case 1

```
Selamat datang di Breach Solver!
Pilih metode input
1. File txt
2. CLI
1
1. Menggunakan file browser
2. Input nama file
2
Masukkan nama file txt!
test1
Matriks yang akan dikerjakan:
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
Sekuens:
BD E9 1C      15 poin
BD 7A BD      20 poin
BD 1C BD 55      30 poin
Solusi ditemukan!
50
7A BD 7A BD 1C BD 55
1, 1
1, 4
3, 4
3, 5
6, 5
6, 3
1, 3
1078 ms
Simpan hasil ke file txt? (y/n)
y
```

3.2 Test Case 2

```
Selamat datang di Breach Solver!
Pilih metode input
1. File txt
2. CLI
1
1. Menggunakan file browser
2. Input nama file
2
Masukkan nama file txt!
test2
Matriks yang akan dikerjakan:
E9 BD 55 1C
E9 1C BD BD
1C BD 55 E9
55 E9 1C BD
Sekuens:
1C 55      10 poin
1C BD      10 poin
BD 1C BD 55      25 poin
Solusi ditemukan!
20
E9 1C 55 1C BD
1, 1
1, 3
3, 3
3, 4
4, 4
3 ms
Simpan hasil ke file txt? (y/n)
y
```

3.3 Test Case 3

```
Selamat datang di Breach Solver!
Pilih metode input
1. File txt
2. CLI
1
1. Menggunakan file browser
2. Input nama file
2
Masukkan nama file txt!
test3
Matriks yang akan dikerjakan:
E9 BD 55 1C
E9 1C BD BD
1C BD 55 E9
55 E9 1C BD
Sekuens:
E9 55      10 poin
BD BD      10 poin
Solusi ditemukan!
20
E9 55 1C BD BD
1, 1
1, 4
3, 4
3, 2
4, 2
3 ms
Simpan hasil ke file txt? (y/n)
y
```

3.4 Test Case 4

```
Selamat datang di Breach Solver!
Pilih metode input
1. File txt
2. CLI
1
1. Menggunakan file browser
2. Input nama file
2
Masukkan nama file txt!
test4
Matriks yang akan dikerjakan:
1C 55 BD 1C 1C
BD 55 BD 1C 55
1C BD BD E9 1C
BD E9 E9 1C E9
1C 1C BD 55 E9
Sekuens:
BD BD      10 poin
E9 1C      15 poin
BD 55 1C      15 poin
Solusi ditemukan!
15
1C BD 55 1C
1, 1
1, 2
2, 2
2, 5
4 ms
Simpan hasil ke file txt? (y/n)
y
Masukkan nama file:
hasil4
File berhasil disimpan!
```

3.5 Test Case 5

```
Jumlah token unik:
5
Masukkan token:
BD 1C 7A 55 E9
Masukkan ukuran buffer:
7
Masukkan ukuran matriks (row col):
6 6
Masukkan jumlah sekuens:
3
Masukkan ukuran maksimal sekuens:
4
Matriks yang akan dikerjakan:
7A 55 1C 55 E9 BD
55 1C 55 BD BD 7A
1C 1C BD 1C 1C E9
BD 1C E9 E9 55 7A
BD 1C 7A 7A E9 E9
55 55 55 7A BD 55
Sekuens:
1C 7A 55      -39 poin
BD 1C      81 poin
1C BD 55 E9      21 poin
Solusi ditemukan!
102
7A BD 1C 1C BD 55 E9
1, 1
1, 4
2, 4
2, 2
4, 2
4, 1
5, 1
1009 ms
Simpan hasil ke file txt? (y/n)
```

3.6 Test Case 6

```
Selamat datang di Breach Solver!
Pilih metode input
1. File txt
2. CLI
2
Jumlah token unik:
5
Masukkan token:
BD 1C 7A 55 E9
Masukkan ukuran buffer:
7
Masukkan ukuran matriks (row col):
6 6
Masukkan jumlah sekuens:
3
Masukkan ukuran maksimal sekuens:
4
Matriks yang akan dikerjakan:
E9 1C 55 BD 7A 55
E9 55 E9 E9 55 1C
55 7A E9 BD 7A 55
7A BD 1C BD 1C 55
7A 1C 1C 1C E9 E9
BD 7A E9 E9 7A BD
Sekuens:
55 BD      -4 poin
55      -60 poin
BD 55      -68 poin
Tidak ada solusi
Simpan hasil ke file txt? (y/n)
y
Masukkan nama file:
hasil6
File berhasil disimpan!
```

Lampiran

Link Repository

https://github.com/florars/Tucil1_13522010

Tabel

Poin	Ya	Tidak
1. Program berhasil dikompilasi tanpa kesalahan	v	
2. Program berhasil dijalankan	v	
3. Program dapat membaca masukan berkas .txt	v	
4. Program dapat menghasilkan masukan secara acak	v	
5. Solusi yang diberikan program optimal	v	
6. Program dapat menyimpan solusi dalam berkas .txt	v	
7. Program memiliki GUI		v