LAPORAN TUGAS KECIL 1 STRATEGI ALGORITMA

Jeremia Axel - 13519188

Kelas 04

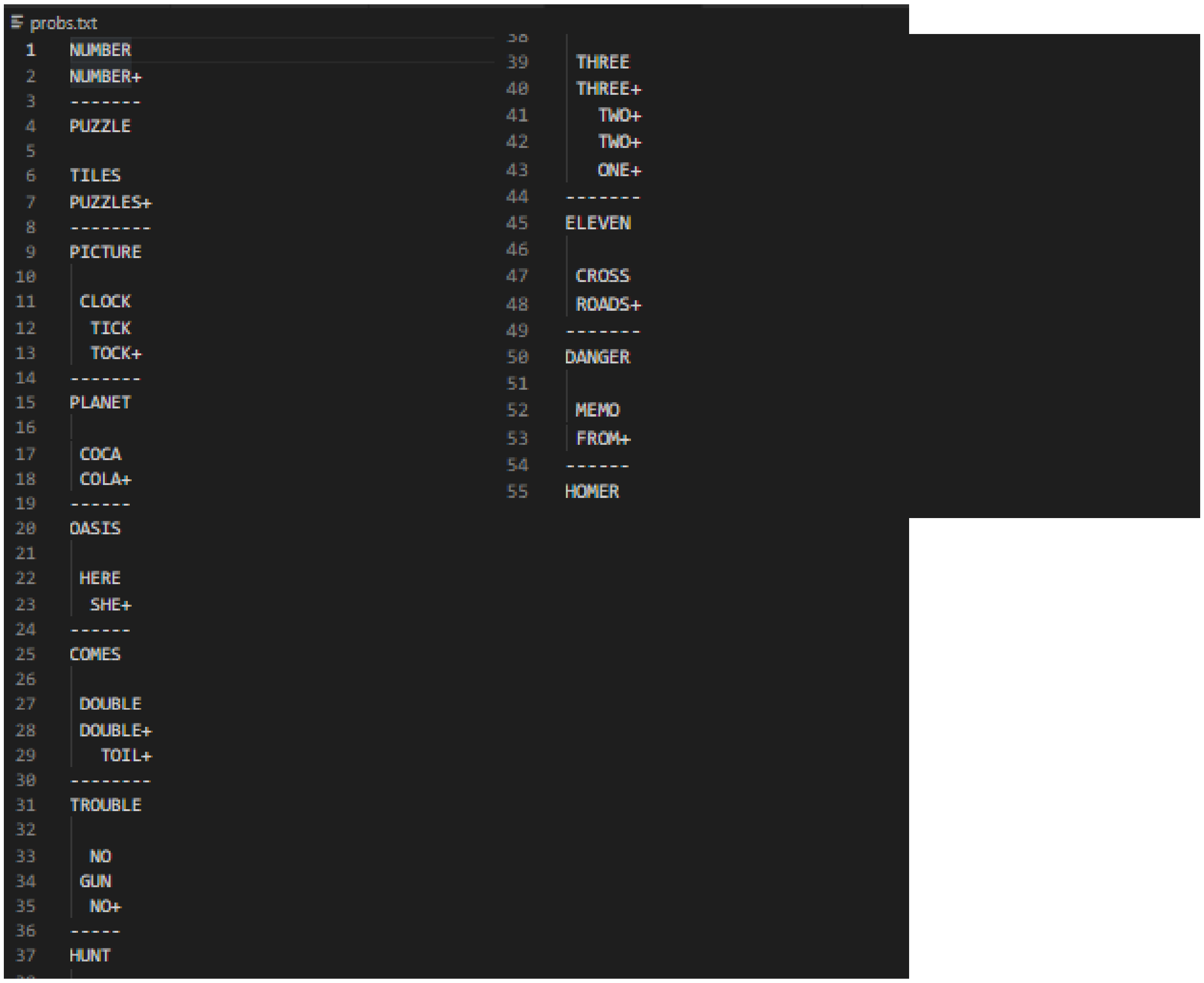
1. Algoritma Brute Force
    a. Ambil seluruh kata dan petakan tiap hurufnya, periksa banyak huruf yang dipetakan. Proses berlanjut jika banyak huruf kurang dari/sama dengan 10.
    b. Buat permutasi dengan angka dari 0 sampai 9 sebanyak huruf yang telah dipetakan.
    c. Pasangkan huruf yang dipetakan dengan kemungkinan susunan angka.
    d. Terjemahkan kata-kata pada soal menjadi angkanya.
    e. Jika penjumlahan nilai operand sama dengan nilai hasilnya. Maka jawaban ditemukan.
    f. Jika belum ditemukan jawabannya, ulangi dari nomor 2.
    g. Jika sudah semua kemungkinan susunan angka dan masih belum ditemukan jawabannya, tidak ada jawaban.

2. Source program (Python)

```python
from pathlib import Path
from os.path import join
import time

TEST_DIR = join(Path(__file__).resolve().parent.parent, "test")

def file_to_list_of_soal(f):
    '''
    Takes file argument and return cleaned list of (list of words)
    file_to_list_of_soal(foo) ->
        [['NUMBER', 'NUMBER', 'PUZZLE'], ['TILES', 'PUZZLES', 'PICTURE']]
    '''
    try:
        print("File read success")
        lines = f.read().splitlines()

        list_of_soals = []
        soal = []
        for i in range(len(lines)):
            lines[i] = lines[i].replace(" ", "").replace("+", "")
            if (lines[i] == ""):
                list_of_soals.append(soal)
                soal = []
            elif not ("-" in lines[i]):
                soal.append(lines[i])

        # soal before end of file
        list_of_soals.append(soal)
        return list_of_soals

    except FileNotFoundError:
        print("File not found")
        exit(1)
    finally:
        f.close()

def is_letter_greater_than_ten(letter_map):
    '''
    Memeriksa apakan panjang map of letter lebih dari 10
    '''
    return len(letter_map) > 10

def permutasyen(sample, perm_len):
    '''
    Membuat permutasi susunan semua kemungkinan dari sample sepanjang perm_len
    '''
    sample = list(sample)
    sample_len = len(sample)

    if sample_len == 1:
        return sample

    # jika panjang susunan lebih dari panjang sample, tidak ada lanjutan
    if (perm_len <= sample_len):
        indexes = list(range(sample_len))
        cycle_list = list(range(sample_len, sample_len-perm_len, -1)) # membuat list banyaknya siklus tiap angka
        yield list(sample[i] for i in indexes[:perm_len]) # permutasi pertama, [0, 1, 2, ... perm_len]

        while cycle_list[0] > 0: # mengulang loop selama siklus digit pertama belum nol
            for i in range(perm_len-1,-1,-1):
                cycle_list[i] -= 1
                if cycle_list[i] == 0: # reset
                    cycle_list[i] = sample_len - i
                    current_index = indexes[i]
                    for x in range(i, sample_len-1):
                        indexes[x] = indexes[x+1]
                    indexes[sample_len-1] = current_index
                else:
                    for x in range(i, sample_len-1):
                        indexes[x], indexes[x+1] = indexes[x+1], indexes[x]
                    yield list(sample[i] for i in indexes[:perm_len])
                    break

def reverse_string(foo):
    '''
    Menyusun string dengan urutan terbalik
    '''
    return foo[::-1]

def any_first_zero(list_of_words, diction):
    '''
    Memeriksa apakah ada kata yang jika diterjemahkan menggunakan diction, berawalan nol
    '''
    for word in list_of_words:
        if diction[word[0]] == 0:
            return True
    return False

def print_result(list_of_words, list_of_results, padding=5):
    '''
    Menerima input dua buah list of string dan integer (opsional)
    Mencetak ke terminal dengan format:
     list_of_string[1]          list_of_results[1]
     list_of_string[2]+         list_of_results[1]+
     ....                       ....
     list_of_string[n-1]+       list_of_results[n-1]+
     --------------------       ---------------------
     list_of_string[n]+         list_of_results[n]+
    jarak antara list_of_string dengan list_of_result sebanyak padding (5, jika tidak dispesifikasikan)
    '''
    n_length = len(list_of_words[-1]) + 1
    for i in range(0, len(list_of_words), 1):
        if i == 0:
            print("{spaces1}{word}{inter_word_padding}{spaces2}{result}".format(
                spaces1=" "*(n_length - len(list_of_words[i]) - 1),
                word=list_of_words[i],
                inter_word_padding=" "*(padding+1),
                spaces2=" "*(n_length - len(list_of_results[i]) - 1),
                result=list_of_results[i]
            ))
        elif (i != len(list_of_words) - 1):
            print("{spaces1}{word}+{inter_word_padding}{spaces2}{result}+
".format(
                spaces1=" "*(n_length - len(list_of_words[i]) - 1),
                word=list_of_words[i],
                inter_word_padding=" "*padding,
                spaces2=" "*(n_length - len(list_of_results[i]) - 1),
                result=list_of_results[i]
            ))
        else :
            print("{stripes}{inter_word_padding}{stripes}".format(
                stripes="-"*n_length,
                inter_word_padding=" "*padding,
            ))
            print("{word}{inter_word_padding}{result}".format(
                word=list_of_words[i],
                inter_word_padding=" "*(padding+1),
                result=list_of_results[i]
            ))

### MAIN ####
print("File input otomatis dipindah ke folder test.")
filename = input("Insert file name : ")

f = open(join(TEST_DIR, filename), 'r')

# total time accumulator
total_time_length = 0

time_start = time.time()
list_of_soals = file_to_list_of_soal(f)

# Iterate for each soal
print("Started.. please wait bekos this is brute force")
soal_pertama = True
for list_of_words in list_of_soals:

    # Time count starts for each soal
    time_start = time.time()

    # Membuat map of letters
    letters_map = []
    for word in list_of_words:
        for letter in reverse_string(word):
            if letter not in letters_map:
                letters_map.append(letter)

```

```python
    # Memeriksa banyak huruf, hanya lanjut jika banyaknya <= 10
    if not (is_letter_greater_than_ten(letters_map)):

        # Penghitung banyaknya percobaan
        try_counter = 0

        # Melakukan permutasi susunan angka 0 sampai 9 sebanyak huruf yang dipetakan
        for numbers in permutasyen(range(0, 10), len(letters_map)):
            try_counter += 1 # increment banyaknya percobaan

            # membuat kamus untuk menerjemahkan huruf
            # -> Memasangkan huruf dengan angka
            myDict = {}
            for i in range(len(letters_map)):
                myDict[letters_map[i]] = numbers[i]

            # check if no word starts with zero in the dictionary
            if not (any_first_zero(list_of_words, myDict)):

                # menerjemahkan tiap kata
                results = []
                for word in list_of_words[:len(list_of_words)]:
                    word_result = ""
                    for letter in word:
                        word_result += str(myDict[letter])
                    results.append(word_result)

                # menjumlahkan operand
                result = 0
                for i in range(len(results)-1):
                    result += int(results[i])

                # check if result is right
                if (result == int(results[-1])):

                    # Time count ends
                    if soal_pertama :
                        time_end = time.time()
                        time_length = time_end - time_start
                        soal_pertama = False
                    else :
                        time_length = time.time() - time_end
                        time_end = time.time()

                    total_time_length += time_length

                    # Show result
                    print("")

                    print_result(list_of_words, results)

                    print("Attempts : {}".format(try_counter))
                    print("Time elapsed {:.4f} second(s)".format(time_length))
                    break

print("")
print("Finished in {:.0f}:{:.0f}:{:.4f}".format(total_time_length//3600, total_time_length//60, total_time_length%60))
```

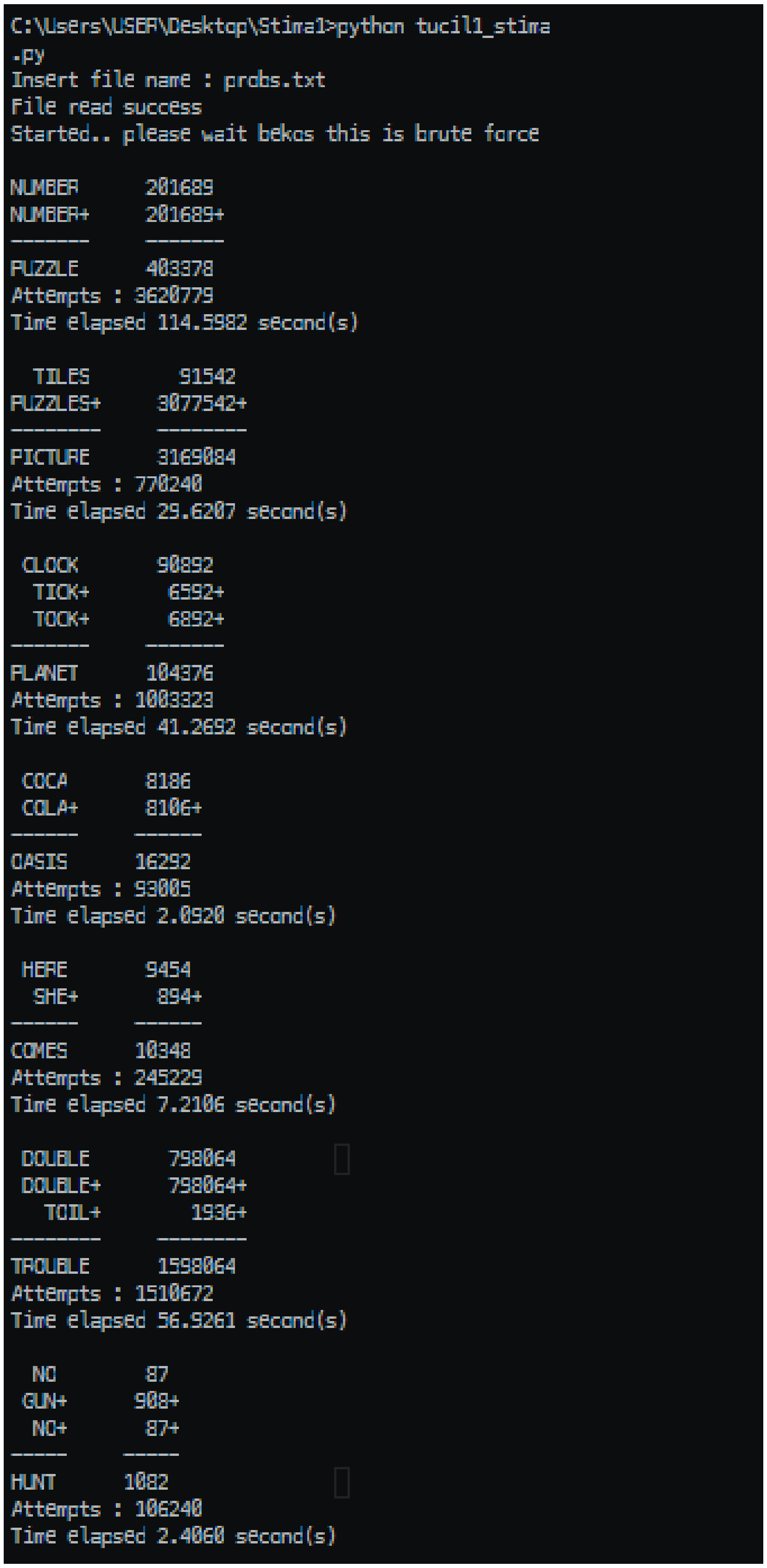
3. Skrinsut *input* dan *output*
    a.
    b. *Input*



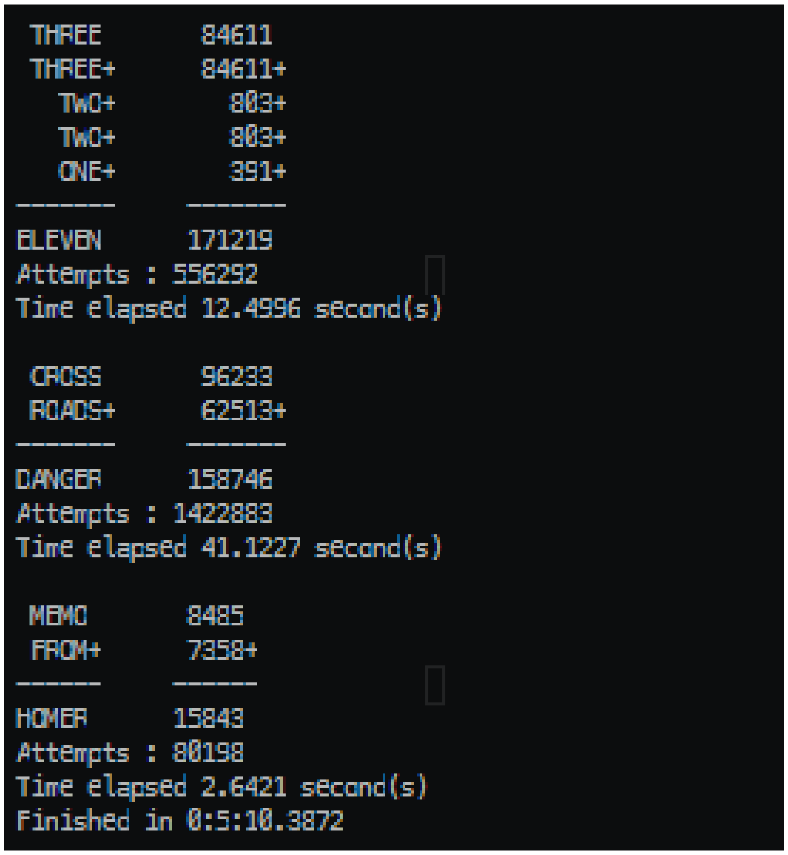
probs.txt

```
NUMBER
NUMBER+
-------
PUZZLE

TILES
PUZZLES+
--------
PICTURE

 CLOCK
  TICK
  TOCK+
-------
PLANET

 COCA
 COLA+
------
OASIS

 HERE
  SHE+
------
COMES

 DOUBLE
 DOUBLE+
   TOIL+
--------
TROUBLE

  NO
 GUN
  NO+
-----
HUNT
```

```
 THREE
 THREE+
   TWO+
   TWO+
   ONE+
-------
ELEVEN

 CROSS
 ROADS+
-------
DANGER

 MEMO
 FROM+
------
HOMER
```



```
C:\Users\USER\Desktop\Stima1>python tucil1_stima
.py
Insert file name : probs.txt
File read success
Started.. please wait bekos this is brute force

NUMBER      201689
NUMBER+     201689+
--------    --------
PUZZLE      403378
Attempts : 3620779
Time elapsed 114.5982 second(s)

  TILES       91542
PUZZLES+     3077542+
---------    ---------
PICTURE      3169084
Attempts : 770240
Time elapsed 29.6207 second(s)

 CLOCK       90892
  TICK+       6592+
  TOCK+       6892+
--------    --------
PLANET      104376
Attempts : 1003323
Time elapsed 41.2692 second(s)

 COCA       8186
 COLA+      8106+
------     ------
OASIS      16292
Attempts : 93005
Time elapsed 2.0920 second(s)

 HERE       9454
  SHE+       894+
------     ------
COMES      10348
Attempts : 245229
Time elapsed 7.2106 second(s)

 DOUBLE      798064
 DOUBLE+     798064+
   TOIL+       1936+
---------    ---------
TROUBLE      1598064
Attempts : 1510672
Time elapsed 56.9261 second(s)

  NO        87
 GUN+      908+
  NO+       87+
-----     -----
HUNT      1082
Attempts : 106240
Time elapsed 2.4060 second(s)
```



```
 THREE       84611
 THREE+      84611+
   TWO+        803+
   TWO+        803+
   ONE+        391+
-------     -------
ELEVEN      171219
Attempts : 556292
Time elapsed 12.4996 second(s)

 CROSS       96233
 ROADS+      62513+
-------     -------
DANGER      158746
Attempts : 1422883
Time elapsed 41.1227 second(s)

 MEMO       8485
 FROM+      7358+
------     ------
HOMER      15843
Attempts : 80198
Time elapsed 2.6421 second(s)
Finished in 0:5:10.3872
```

4. Alamat Google Drive
   https://drive.google.com/file/d/1ag8gfQ8e9oe9g7bUl1FQ62_tx2KclWk9/view?usp=sharing

Checklist

Poin	Ya	Tidak
1. Program berhasil dikompilasi tanpa kesalahan (no syntax error)	✓	
2. Program berhasil *running*	✓	
3. Program dapat membaca file masukan dan menuliskan luaran.	✓	
4. Solusi *cryptarithmetic* hanya benar untuk persoalan *cryptarithmetic* dengan dua buah *operand*.		✓
5. Solusi *cryptarithmetic* benar untuk persoalan *cryptarithmetic* lebih dari dua *operand*.	✓	