LAPORAN TUGAS KECIL 1
IF2211 - STRATEGI ALGORITMA
PENYELESAIAN CRYPTARITHMETIC DENGAN ALGORITMA BRUTE FORCE
SEMESTER II TAHUN 2020/2021

Oleh:
Aurelius Marcel Candra (13519198)

Prodi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung
2020

BAGIAN 1
PENJELASAN ALGORITMA

Urutan langkah-langkah yang dilakukan oleh program secara sederhana adalah sebagai berikut:

1. Dilakukan **pencatatan** terhadap semua **huruf** yang muncul pada persoalan. Secara lebih tepat, pada Langkah pertama dilakukan pembuatan himpunan yang berisi huruf-huruf yang ada dalam persoalan.
   Misalnya terdapat persoalan:

   ```
   JUNE
   JULY+
   --------
   APRIL
   ```

   Program akan mencatat: [‘J’, ‘U’, ‘N’, ‘E’, ‘L’, ‘Y’, ‘A’, ‘P’, ‘R’, ‘I’]

2. Dilakukan **perulangan** terhadap instruksi **permutasi** secara menyeluruh dari 10 digit, yaitu sejumlah 10! = 3.628.800.

3. Dari setiap hasil permutasi yang diperoleh, dibuat ***mapping*** dari himpunan huruf ke hasil permutasi tersebut.
   Misalnya:
   [‘J’ : 0, ‘U’ : 1, ‘N’ : 2, ‘E’ : 3, ‘L’ : 4, ‘Y’ : 5, ‘A’ : 6, ‘P’ : 7, ‘R’ : 8, ‘I’ : 9]

4. Berdasarkan mapping yang telah diperoleh, kemudian dilakukan **substitusi** terhadap setiap kata yang terlibat dalam persoalan. Terakhir, dilakukan **kalkulasi** untuk menentukan apakah penjumlahan operand (kata-kata sebelum garis pemisah penjumlahan) sama dengan hasil akhir (kata di baris terakhir)
   Misalnya, menurut mapping di atas:
   (0)*123 + (0)145 … 67894

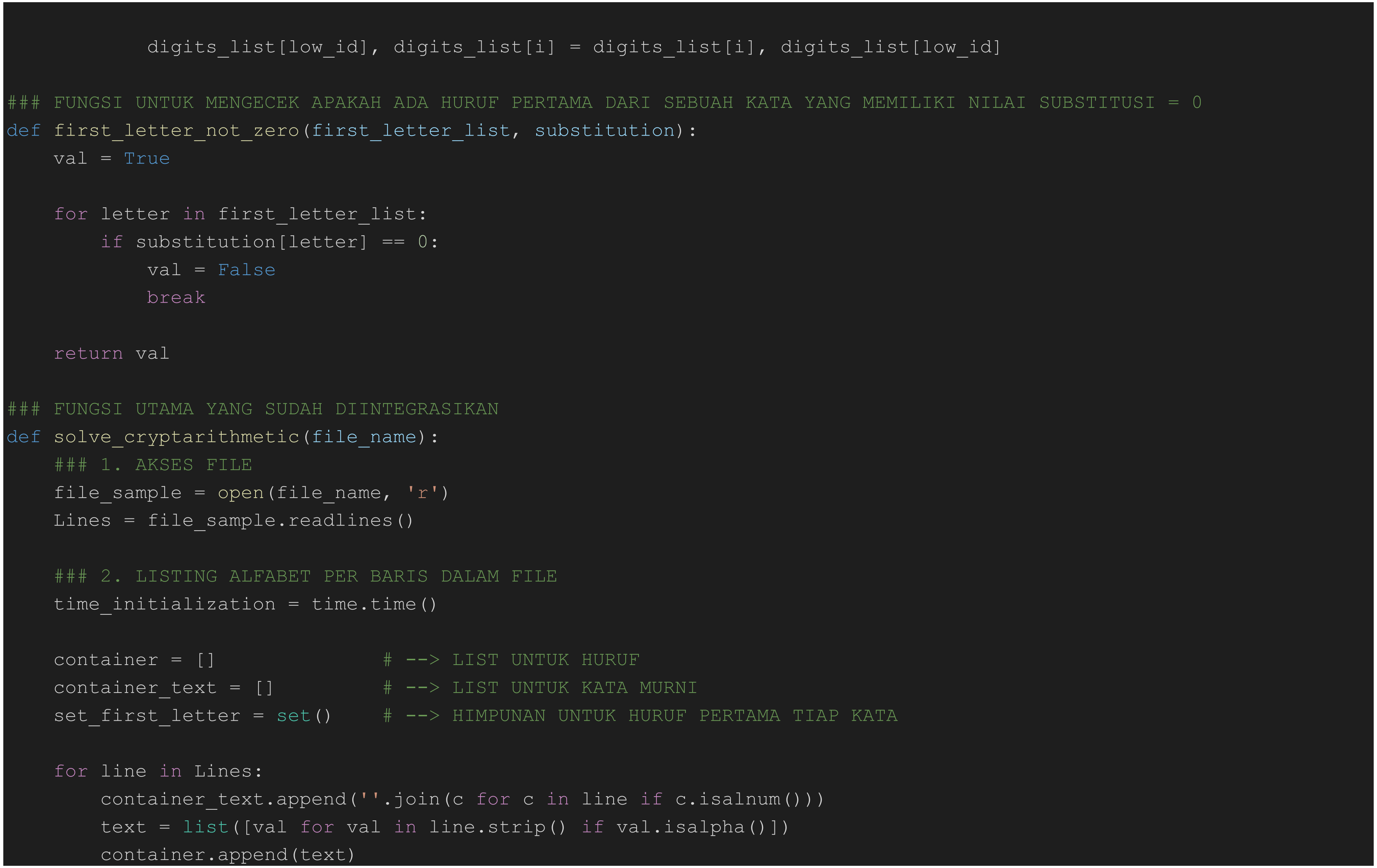
   Jika hasilnya berbeda seperti contoh, tidak ditampilkan apa-apa dan dilanjutkan ke bentuk permutasi berikutnya. Jika hasilnya sama, ditampilkan bukti penjumlahan yang benar beserta mapping yang digunakan dan juga ditampilkan informasi-informasi tambahan sesuai spesifikasi, seperti waktu dan total uji coba permutasi.

   *) Dalam program aslinya, dilakukan filter atau seleksi terhadap mapping yang memberikan nilai 0 terhadap huruf pertama dari suatu kata

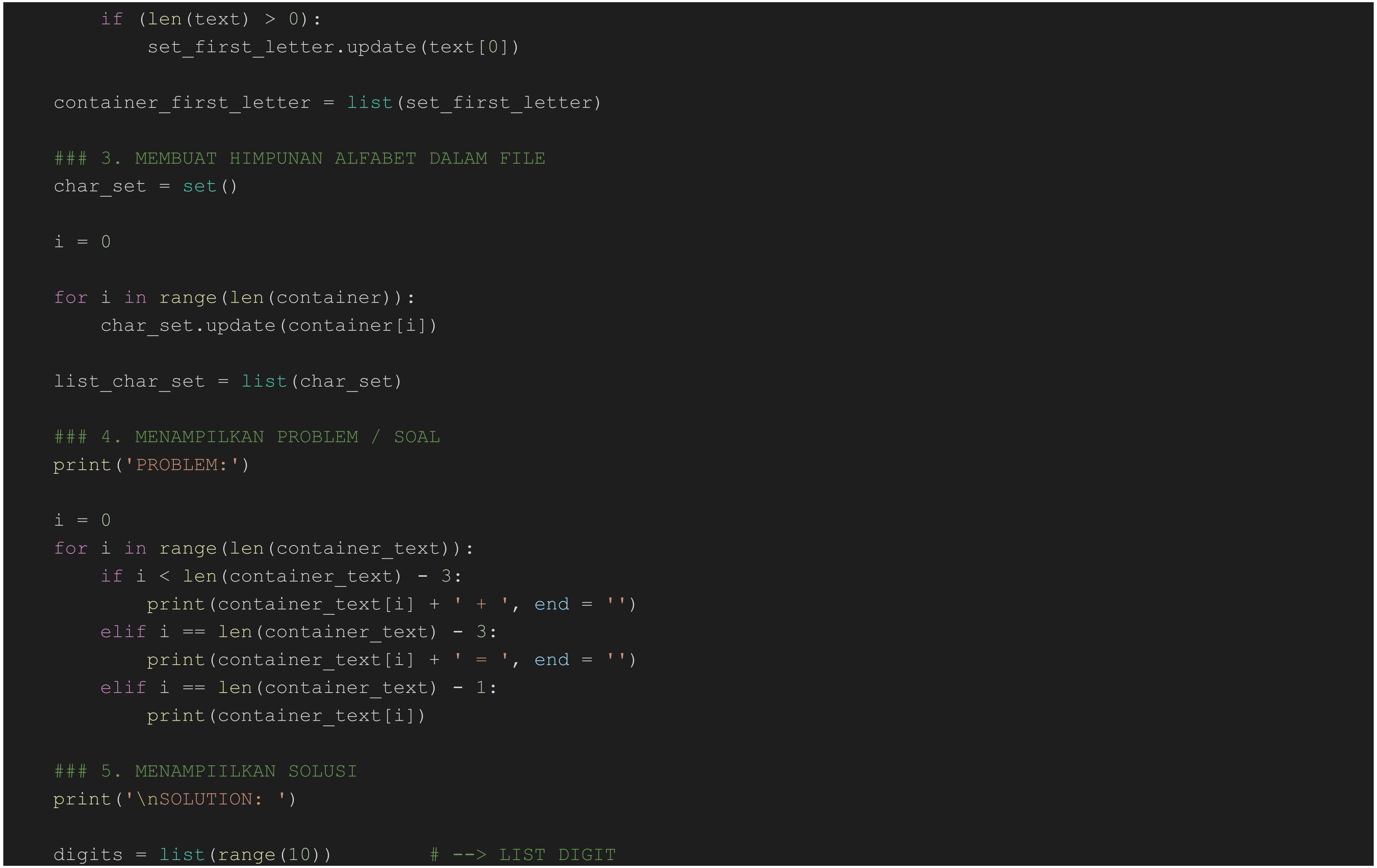
BAGIAN 2
SOURCE CODE

Program pencari solusi dari *cryptarithmetic* ini dibuat dalam bahasa Python dikarenakan kemudahan untuk melakukan manipulasi string dan sintaks yang sederhana tetapi efektif. *Source code* program adalah sebagai berikut:

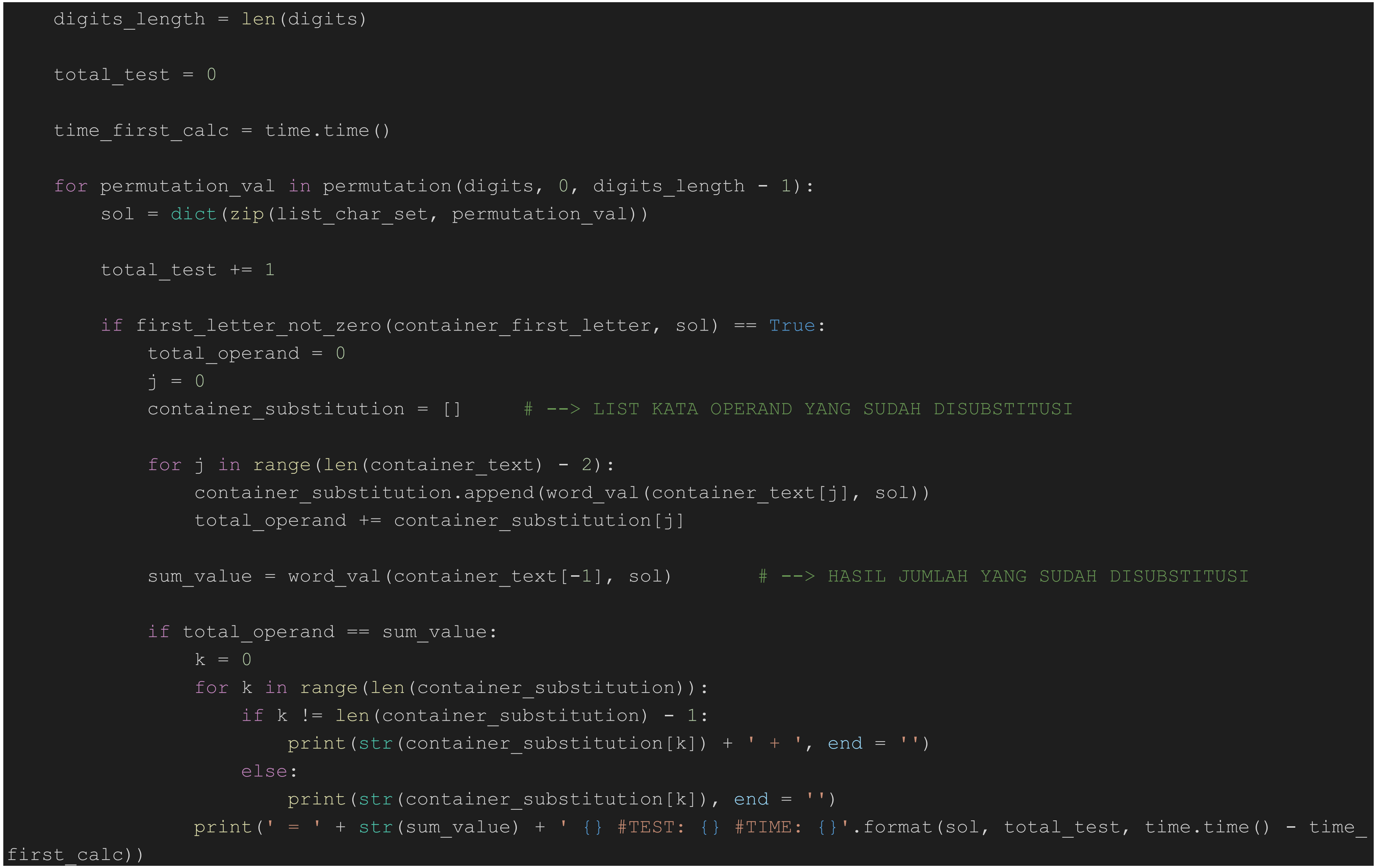
```
import os
import time

### FUNGSI UNTUK KONVERSI KATA MENJADI BILANGAN, INSPIRASI DARI: https://stackoverflow.com/questions/35975748/python3-cryptarithmetic-puzzle-generic-solution-in-python3
def word_val(word, letter_dict):
    total_val = 0
    factor = 1

    for letter in reversed(word):
        total_val += factor * letter_dict[letter]
        factor *= 10

    return total_val

### FUNGSI UNTUK MENGHASILKAN PERMUTASI DARI HIMPUNAN DIGIT, MODIFIKASI DARI: https://www.geeksforgeeks.org/write-a-c-program-to-print-all-permutations-of-a-given-string/
def permutation(digits_list, low_id, top_id):
    if low_id == top_id:
        yield digits_list
    else:
        for i in range(low_id, top_id + 1):
            digits_list[low_id], digits_list[i] = digits_list[i], digits_list[low_id]

            yield from permutation(digits_list, low_id + 1, top_id)
```



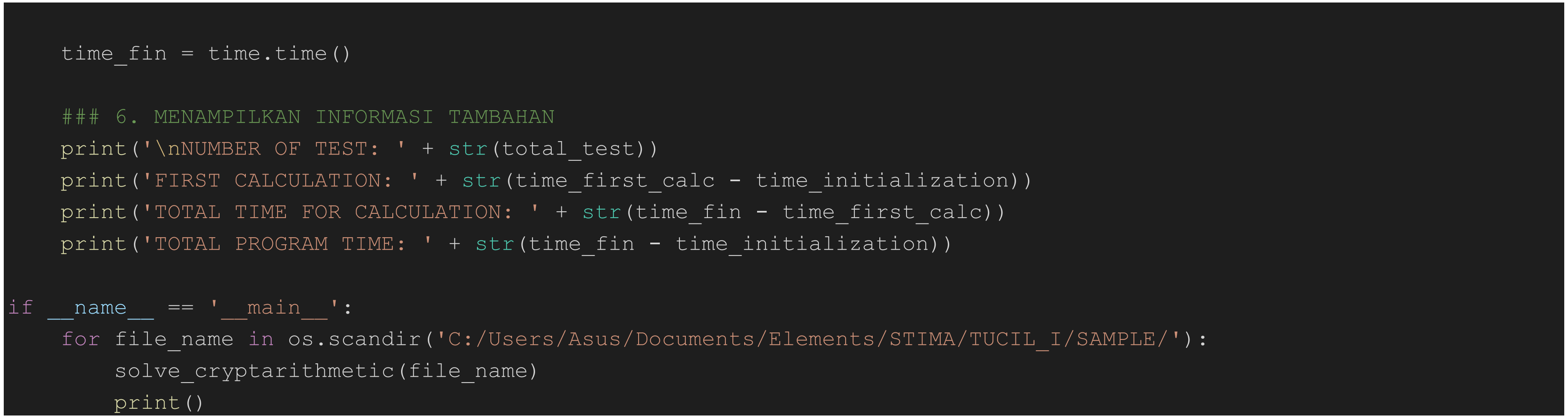
```
            digits_list[low_id], digits_list[i] = digits_list[i], digits_list[low_id]

### FUNGSI UNTUK MENGECEK APAKAH ADA HURUF PERTAMA DARI SEBUAH KATA YANG MEMILIKI NILAI SUBSTITUSI = 0
def first_letter_not_zero(first_letter_list, substitution):
    val = True

    for letter in first_letter_list:
        if substitution[letter] == 0:
            val = False
            break

    return val

### FUNGSI UTAMA YANG SUDAH DIINTEGRASIKAN
def solve_cryptarithmetic(file_name):
    ### 1. AKSES FILE
    file_sample = open(file_name, 'r')
    Lines = file_sample.readlines()

    ### 2. LISTING ALFABET PER BARIS DALAM FILE
    time_initialization = time.time()

    container = []              # --> LIST UNTUK HURUF
    container_text = []         # --> LIST UNTUK KATA MURNI
    set_first_letter = set()    # --> HIMPUNAN UNTUK HURUF PERTAMA TIAP KATA

    for line in Lines:
        container_text.append(''.join(c for c in line if c.isalnum()))
        text = list([val for val in line.strip() if val.isalpha()])
        container.append(text)
```



```
    if (len(text) > 0):
        set_first_letter.update(text[0])

container_first_letter = list(set_first_letter)

### 3. MEMBUAT HIMPUNAN ALFABET DALAM FILE
char_set = set()

i = 0

for i in range(len(container)):
    char_set.update(container[i])

list_char_set = list(char_set)

### 4. MENAMPILKAN PROBLEM / SOAL
print('PROBLEM:')

i = 0
for i in range(len(container_text)):
    if i < len(container_text) - 3:
        print(container_text[i] + ' + ', end = '')
    elif i == len(container_text) - 3:
        print(container_text[i] + ' = ', end = '')
    elif i == len(container_text) - 1:
        print(container_text[i])

### 5. MENAMPIILKAN SOLUSI
print('\nSOLUTION: ')

digits = list(range(10))        # --> LIST DIGIT
```



```
    digits_length = len(digits)

    total_test = 0

    time_first_calc = time.time()

    for permutation_val in permutation(digits, 0, digits_length - 1):
        sol = dict(zip(list_char_set, permutation_val))

        total_test += 1

        if first_letter_not_zero(container_first_letter, sol) == True:
            total_operand = 0
            j = 0
            container_substitution = []     # --> LIST KATA OPERAND YANG SUDAH DISUBSTITUSI

            for j in range(len(container_text) - 2):
                container_substitution.append(word_val(container_text[j], sol))
                total_operand += container_substitution[j]

            sum_value = word_val(container_text[-1], sol)       # --> HASIL JUMLAH YANG SUDAH DISUBSTITUSI

            if total_operand == sum_value:
                k = 0
                for k in range(len(container_substitution)):
                    if k != len(container_substitution) - 1:
                        print(str(container_substitution[k]) + ' + ', end = '')
                    else:
                        print(str(container_substitution[k]), end = '')
                print(' = ' + str(sum_value) + ' {} #TEST: {} #TIME: {}'.format(sol, total_test, time.time() - time_
first_calc))
```



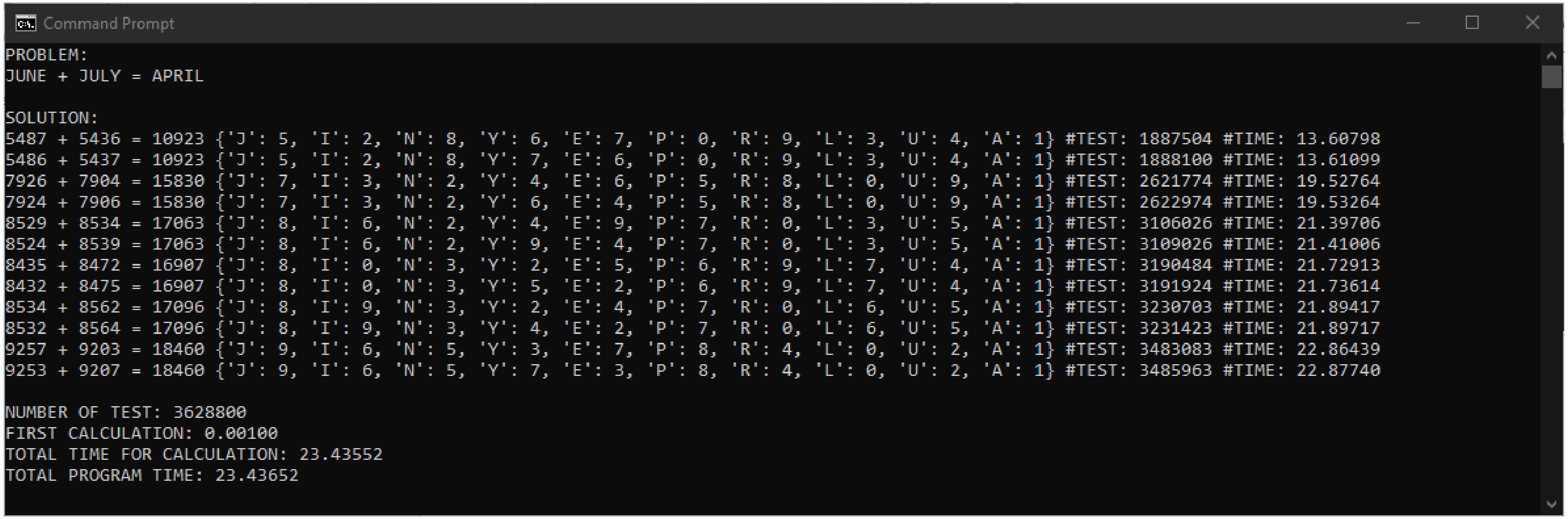
```
    time_fin = time.time()

    ### 6. MENAMPILKAN INFORMASI TAMBAHAN
    print('\nNUMBER OF TEST: ' + str(total_test))
    print('FIRST CALCULATION: ' + str(time_first_calc - time_initialization))
    print('TOTAL TIME FOR CALCULATION: ' + str(time_fin - time_first_calc))
    print('TOTAL PROGRAM TIME: ' + str(time_fin - time_initialization))

if __name__ == '__main__':
    for file_name in os.scandir('C:/Users/Asus/Documents/Elements/STIMA/TUCIL_I/SAMPLE/'):
        solve_cryptarithmetic(file_name)
        print()
```

BAGIAN 3
HASIL PERCOBAAN

INPUT: (sample0.txt)

```
JUNE
JULY+
-----
APRIL
```

OUTPUT:

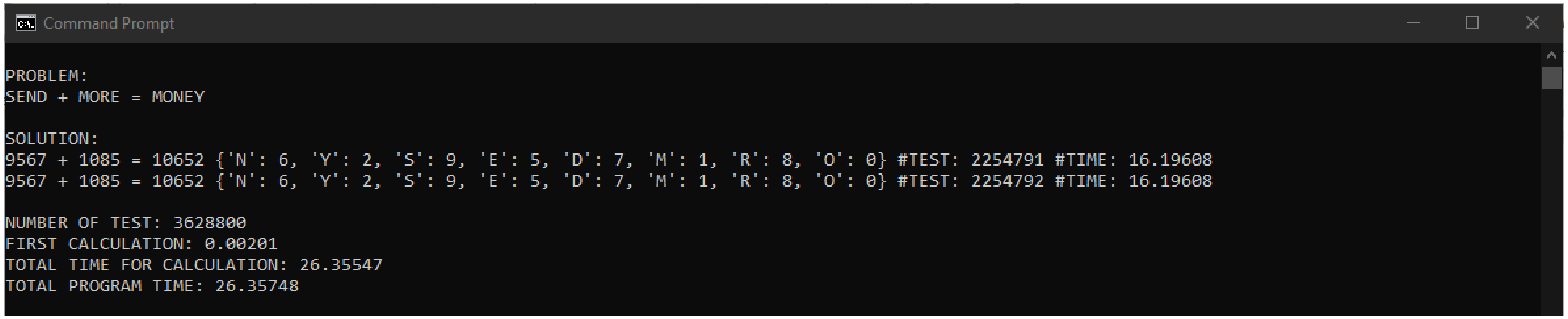


```
PROBLEM:
JUNE + JULY = APRIL

SOLUTION:
5487 + 5436 = 10923 {'J': 5, 'I': 2, 'N': 8, 'Y': 6, 'E': 7, 'P': 0, 'R': 9, 'L': 3, 'U': 4, 'A': 1} #TEST: 1887504 #TIME: 13.60798
5486 + 5437 = 10923 {'J': 5, 'I': 2, 'N': 8, 'Y': 7, 'E': 6, 'P': 0, 'R': 9, 'L': 3, 'U': 4, 'A': 1} #TEST: 1888100 #TIME: 13.61099
7926 + 7904 = 15830 {'J': 7, 'I': 3, 'N': 2, 'Y': 4, 'E': 6, 'P': 5, 'R': 8, 'L': 0, 'U': 9, 'A': 1} #TEST: 2621774 #TIME: 19.52764
7924 + 7906 = 15830 {'J': 7, 'I': 3, 'N': 2, 'Y': 6, 'E': 4, 'P': 5, 'R': 8, 'L': 0, 'U': 9, 'A': 1} #TEST: 2622974 #TIME: 19.53264
8529 + 8534 = 17063 {'J': 8, 'I': 6, 'N': 2, 'Y': 4, 'E': 9, 'P': 7, 'R': 0, 'L': 3, 'U': 5, 'A': 1} #TEST: 3106026 #TIME: 21.39706
8524 + 8539 = 17063 {'J': 8, 'I': 6, 'N': 2, 'Y': 9, 'E': 4, 'P': 7, 'R': 0, 'L': 3, 'U': 5, 'A': 1} #TEST: 3109026 #TIME: 21.41006
8435 + 8472 = 16907 {'J': 8, 'I': 0, 'N': 3, 'Y': 2, 'E': 5, 'P': 6, 'R': 9, 'L': 7, 'U': 4, 'A': 1} #TEST: 3190484 #TIME: 21.72913
8432 + 8475 = 16907 {'J': 8, 'I': 0, 'N': 3, 'Y': 5, 'E': 2, 'P': 6, 'R': 9, 'L': 7, 'U': 4, 'A': 1} #TEST: 3191924 #TIME: 21.73614
8534 + 8562 = 17096 {'J': 8, 'I': 9, 'N': 3, 'Y': 2, 'E': 4, 'P': 7, 'R': 0, 'L': 6, 'U': 5, 'A': 1} #TEST: 3230703 #TIME: 21.89417
8532 + 8564 = 17096 {'J': 8, 'I': 9, 'N': 3, 'Y': 4, 'E': 2, 'P': 7, 'R': 0, 'L': 6, 'U': 5, 'A': 1} #TEST: 3231423 #TIME: 21.89717
9257 + 9203 = 18460 {'J': 9, 'I': 6, 'N': 5, 'Y': 3, 'E': 7, 'P': 8, 'R': 4, 'L': 0, 'U': 2, 'A': 1} #TEST: 3483083 #TIME: 22.86439
9253 + 9207 = 18460 {'J': 9, 'I': 6, 'N': 5, 'Y': 7, 'E': 3, 'P': 8, 'R': 4, 'L': 0, 'U': 2, 'A': 1} #TEST: 3485963 #TIME: 22.87740

NUMBER OF TEST: 3628800
FIRST CALCULATION: 0.00100
TOTAL TIME FOR CALCULATION: 23.43552
TOTAL PROGRAM TIME: 23.43652
```

INPUT: (sample1.txt)

```
SEND
MORE+
-----
MONEY
```

OUTPUT:

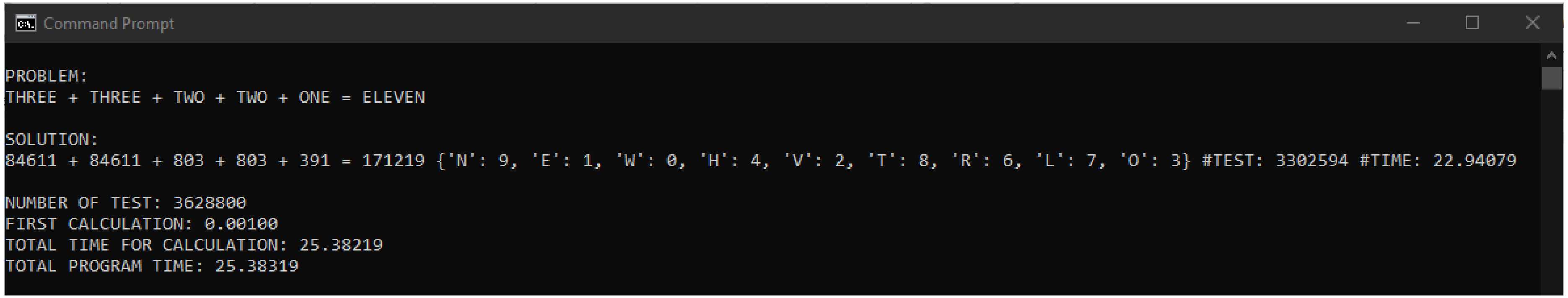


```
PROBLEM:
SEND + MORE = MONEY

SOLUTION:
9567 + 1085 = 10652 {'N': 6, 'Y': 2, 'S': 9, 'E': 5, 'D': 7, 'M': 1, 'R': 8, 'O': 0} #TEST: 2254791 #TIME: 16.19608
9567 + 1085 = 10652 {'N': 6, 'Y': 2, 'S': 9, 'E': 5, 'D': 7, 'M': 1, 'R': 8, 'O': 0} #TEST: 2254792 #TIME: 16.19608

NUMBER OF TEST: 3628800
FIRST CALCULATION: 0.00201
TOTAL TIME FOR CALCULATION: 26.35547
TOTAL PROGRAM TIME: 26.35748
```

INPUT: (sample2.txt)

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

OUTPUT:

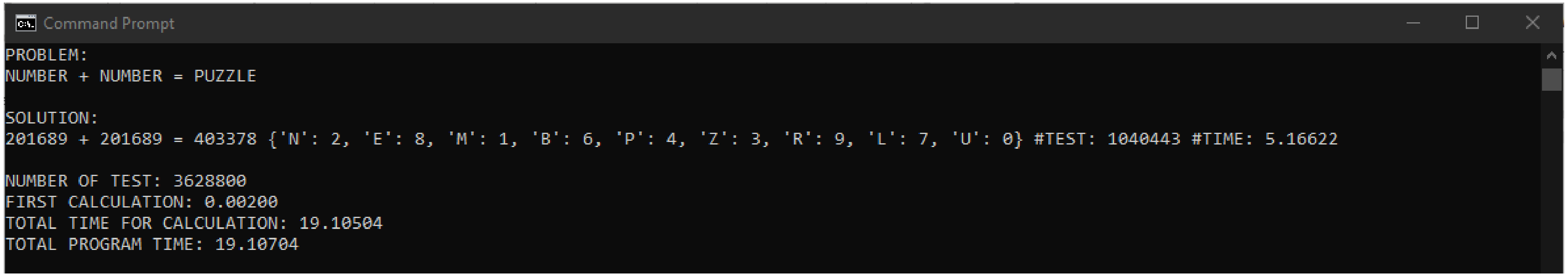


```
PROBLEM:
THREE + THREE + TWO + TWO + ONE = ELEVEN

SOLUTION:
84611 + 84611 + 803 + 803 + 391 = 171219 {'N': 9, 'E': 1, 'W': 0, 'H': 4, 'V': 2, 'T': 8, 'R': 6, 'L': 7, 'O': 3} #TEST: 3302594 #TIME: 22.94079

NUMBER OF TEST: 3628800
FIRST CALCULATION: 0.00100
TOTAL TIME FOR CALCULATION: 25.38219
TOTAL PROGRAM TIME: 25.38319
```

INPUT: (sample3.txt)

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

OUTPUT:

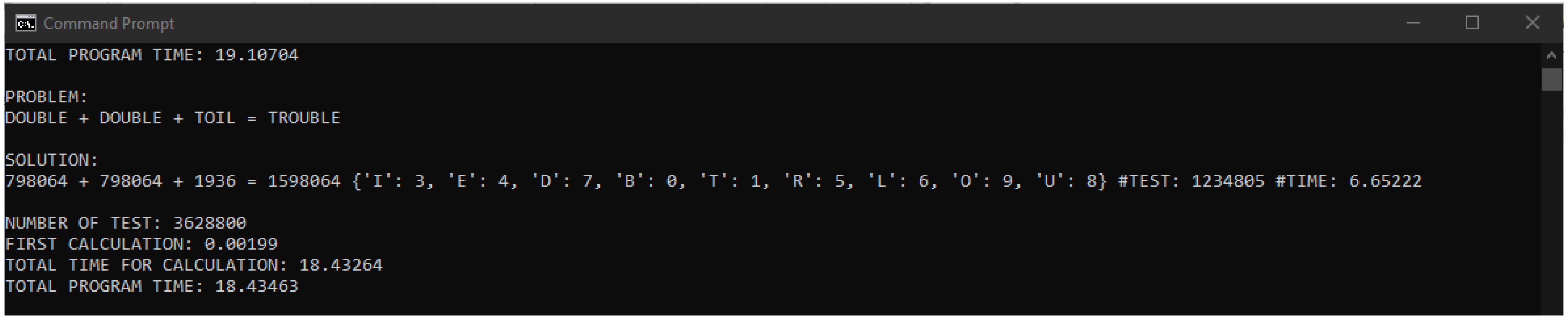


```
PROBLEM:
NUMBER + NUMBER = PUZZLE

SOLUTION:
201689 + 201689 = 403378 {'N': 2, 'E': 8, 'M': 1, 'B': 6, 'P': 4, 'Z': 3, 'R': 9, 'L': 7, 'U': 0} #TEST: 1040443 #TIME: 5.16622

NUMBER OF TEST: 3628800
FIRST CALCULATION: 0.00200
TOTAL TIME FOR CALCULATION: 19.10504
TOTAL PROGRAM TIME: 19.10704
```

INPUT: (sample4.txt)

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

OUTPUT:

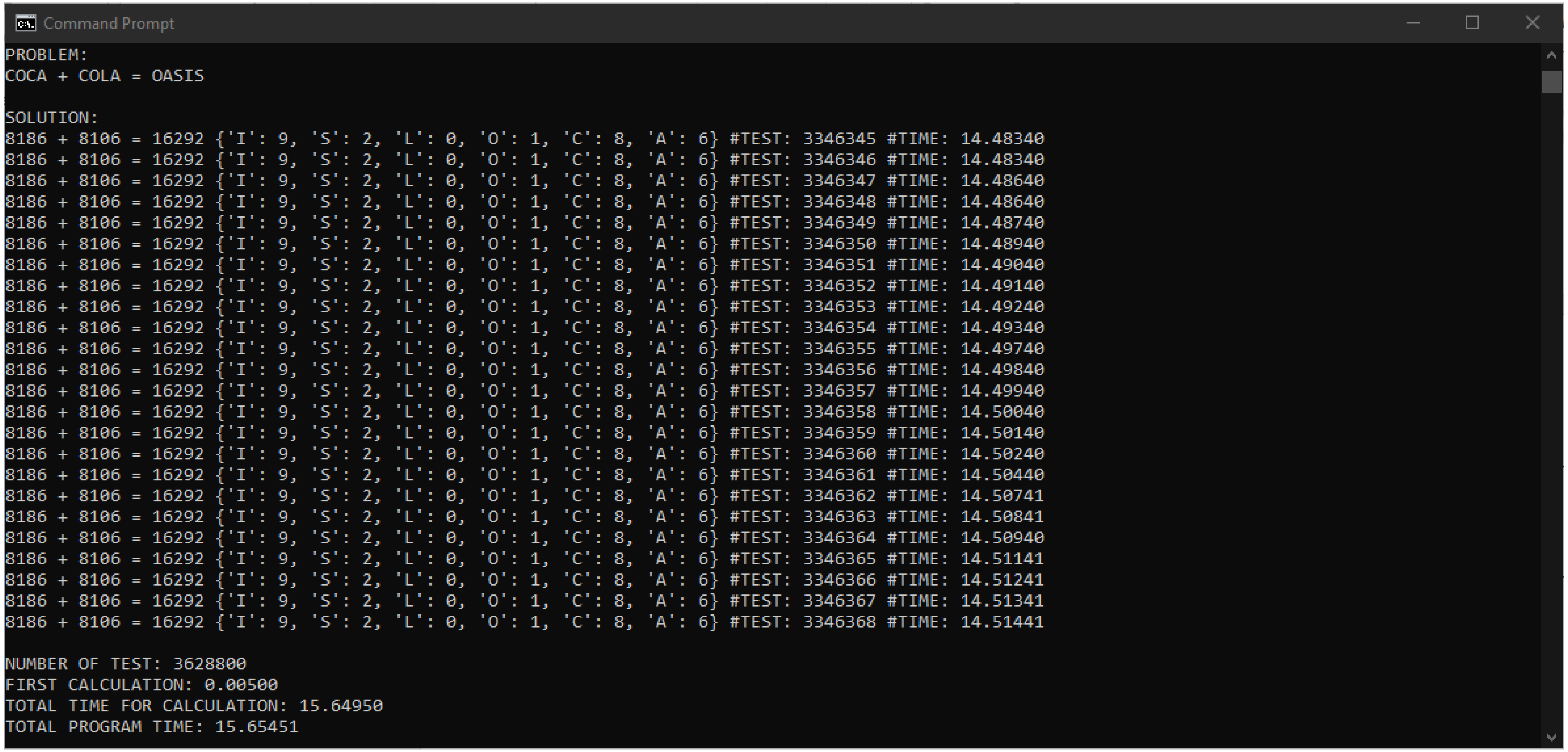


```
TOTAL PROGRAM TIME: 19.10704

PROBLEM:
DOUBLE + DOUBLE + TOIL = TROUBLE

SOLUTION:
798064 + 798064 + 1936 = 1598064 {'I': 3, 'E': 4, 'D': 7, 'B': 0, 'T': 1, 'R': 5, 'L': 6, 'O': 9, 'U': 8} #TEST: 1234805 #TIME: 6.65222

NUMBER OF TEST: 3628800
FIRST CALCULATION: 0.00199
TOTAL TIME FOR CALCULATION: 18.43264
TOTAL PROGRAM TIME: 18.43463
```

INPUT: (sample5.txt)

```
COCA
COLA+
-----
OASIS
```

OUTPUT:

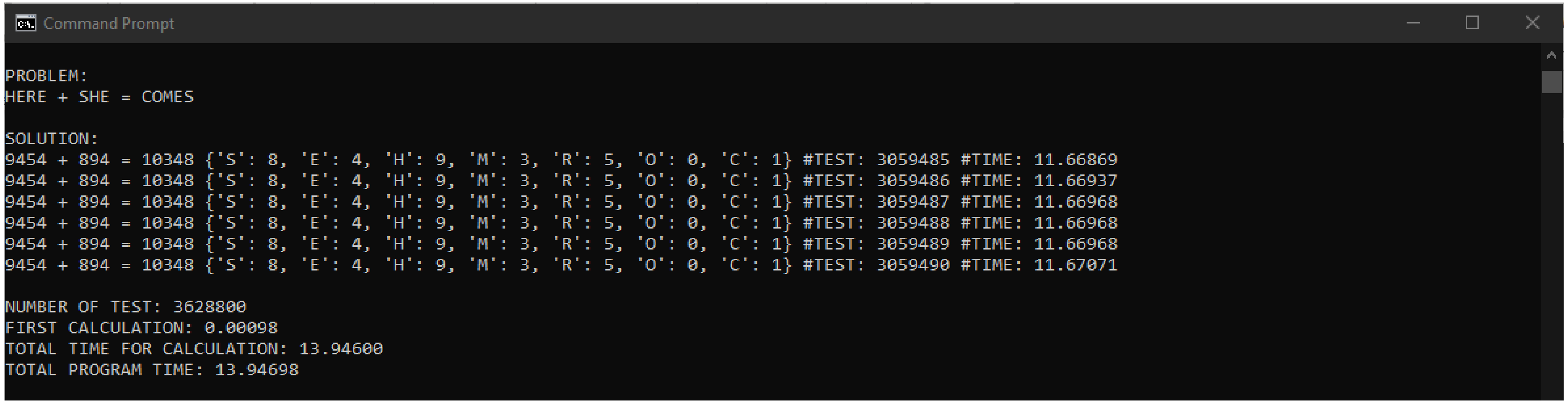


```
PROBLEM:
COCA + COLA = OASIS

SOLUTION:
8186 + 8106 = 16292 {'I': 9, 'S': 2, 'L': 0, 'O': 1, 'C': 8, 'A': 6} #TEST: 3346345 #TIME: 14.48340
8186 + 8106 = 16292 {'I': 9, 'S': 2, 'L': 0, 'O': 1, 'C': 8, 'A': 6} #TEST: 3346346 #TIME: 14.48340
8186 + 8106 = 16292 {'I': 9, 'S': 2, 'L': 0, 'O': 1, 'C': 8, 'A': 6} #TEST: 3346347 #TIME: 14.48640
8186 + 8106 = 16292 {'I': 9, 'S': 2, 'L': 0, 'O': 1, 'C': 8, 'A': 6} #TEST: 3346348 #TIME: 14.48640
8186 + 8106 = 16292 {'I': 9, 'S': 2, 'L': 0, 'O': 1, 'C': 8, 'A': 6} #TEST: 3346349 #TIME: 14.48740
8186 + 8106 = 16292 {'I': 9, 'S': 2, 'L': 0, 'O': 1, 'C': 8, 'A': 6} #TEST: 3346350 #TIME: 14.48940
8186 + 8106 = 16292 {'I': 9, 'S': 2, 'L': 0, 'O': 1, 'C': 8, 'A': 6} #TEST: 3346351 #TIME: 14.49040
8186 + 8106 = 16292 {'I': 9, 'S': 2, 'L': 0, 'O': 1, 'C': 8, 'A': 6} #TEST: 3346352 #TIME: 14.49140
8186 + 8106 = 16292 {'I': 9, 'S': 2, 'L': 0, 'O': 1, 'C': 8, 'A': 6} #TEST: 3346353 #TIME: 14.49240
8186 + 8106 = 16292 {'I': 9, 'S': 2, 'L': 0, 'O': 1, 'C': 8, 'A': 6} #TEST: 3346354 #TIME: 14.49340
8186 + 8106 = 16292 {'I': 9, 'S': 2, 'L': 0, 'O': 1, 'C': 8, 'A': 6} #TEST: 3346355 #TIME: 14.49740
8186 + 8106 = 16292 {'I': 9, 'S': 2, 'L': 0, 'O': 1, 'C': 8, 'A': 6} #TEST: 3346356 #TIME: 14.49840
8186 + 8106 = 16292 {'I': 9, 'S': 2, 'L': 0, 'O': 1, 'C': 8, 'A': 6} #TEST: 3346357 #TIME: 14.49940
8186 + 8106 = 16292 {'I': 9, 'S': 2, 'L': 0, 'O': 1, 'C': 8, 'A': 6} #TEST: 3346358 #TIME: 14.50040
8186 + 8106 = 16292 {'I': 9, 'S': 2, 'L': 0, 'O': 1, 'C': 8, 'A': 6} #TEST: 3346359 #TIME: 14.50140
8186 + 8106 = 16292 {'I': 9, 'S': 2, 'L': 0, 'O': 1, 'C': 8, 'A': 6} #TEST: 3346360 #TIME: 14.50240
8186 + 8106 = 16292 {'I': 9, 'S': 2, 'L': 0, 'O': 1, 'C': 8, 'A': 6} #TEST: 3346361 #TIME: 14.50440
8186 + 8106 = 16292 {'I': 9, 'S': 2, 'L': 0, 'O': 1, 'C': 8, 'A': 6} #TEST: 3346362 #TIME: 14.50741
8186 + 8106 = 16292 {'I': 9, 'S': 2, 'L': 0, 'O': 1, 'C': 8, 'A': 6} #TEST: 3346363 #TIME: 14.50841
8186 + 8106 = 16292 {'I': 9, 'S': 2, 'L': 0, 'O': 1, 'C': 8, 'A': 6} #TEST: 3346364 #TIME: 14.50940
8186 + 8106 = 16292 {'I': 9, 'S': 2, 'L': 0, 'O': 1, 'C': 8, 'A': 6} #TEST: 3346365 #TIME: 14.51141
8186 + 8106 = 16292 {'I': 9, 'S': 2, 'L': 0, 'O': 1, 'C': 8, 'A': 6} #TEST: 3346366 #TIME: 14.51241
8186 + 8106 = 16292 {'I': 9, 'S': 2, 'L': 0, 'O': 1, 'C': 8, 'A': 6} #TEST: 3346367 #TIME: 14.51341
8186 + 8106 = 16292 {'I': 9, 'S': 2, 'L': 0, 'O': 1, 'C': 8, 'A': 6} #TEST: 3346368 #TIME: 14.51441

NUMBER OF TEST: 3628800
FIRST CALCULATION: 0.00500
TOTAL TIME FOR CALCULATION: 15.64950
TOTAL PROGRAM TIME: 15.65451
```

INPUT: (sample6.txt)

```
SHERE
SHE+
-----
COMES
```

OUTPUT:

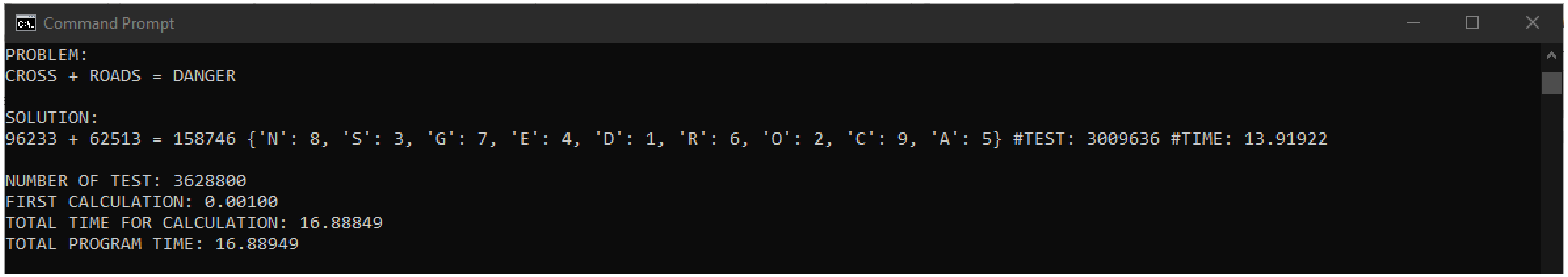


```
PROBLEM:
HERE + SHE = COMES

SOLUTION:
9454 + 894 = 10348 {'S': 8, 'E': 4, 'H': 9, 'M': 3, 'R': 5, 'O': 0, 'C': 1} #TEST: 3059485 #TIME: 11.66869
9454 + 894 = 10348 {'S': 8, 'E': 4, 'H': 9, 'M': 3, 'R': 5, 'O': 0, 'C': 1} #TEST: 3059486 #TIME: 11.66937
9454 + 894 = 10348 {'S': 8, 'E': 4, 'H': 9, 'M': 3, 'R': 5, 'O': 0, 'C': 1} #TEST: 3059487 #TIME: 11.66968
9454 + 894 = 10348 {'S': 8, 'E': 4, 'H': 9, 'M': 3, 'R': 5, 'O': 0, 'C': 1} #TEST: 3059488 #TIME: 11.66968
9454 + 894 = 10348 {'S': 8, 'E': 4, 'H': 9, 'M': 3, 'R': 5, 'O': 0, 'C': 1} #TEST: 3059489 #TIME: 11.66968
9454 + 894 = 10348 {'S': 8, 'E': 4, 'H': 9, 'M': 3, 'R': 5, 'O': 0, 'C': 1} #TEST: 3059490 #TIME: 11.67071

NUMBER OF TEST: 3628800
FIRST CALCULATION: 0.00098
TOTAL TIME FOR CALCULATION: 13.94600
TOTAL PROGRAM TIME: 13.94698
```

INPUT: (sample7.txt)

```
CROSS
ROADS+
-----
DANGER
```

OUTPUT:

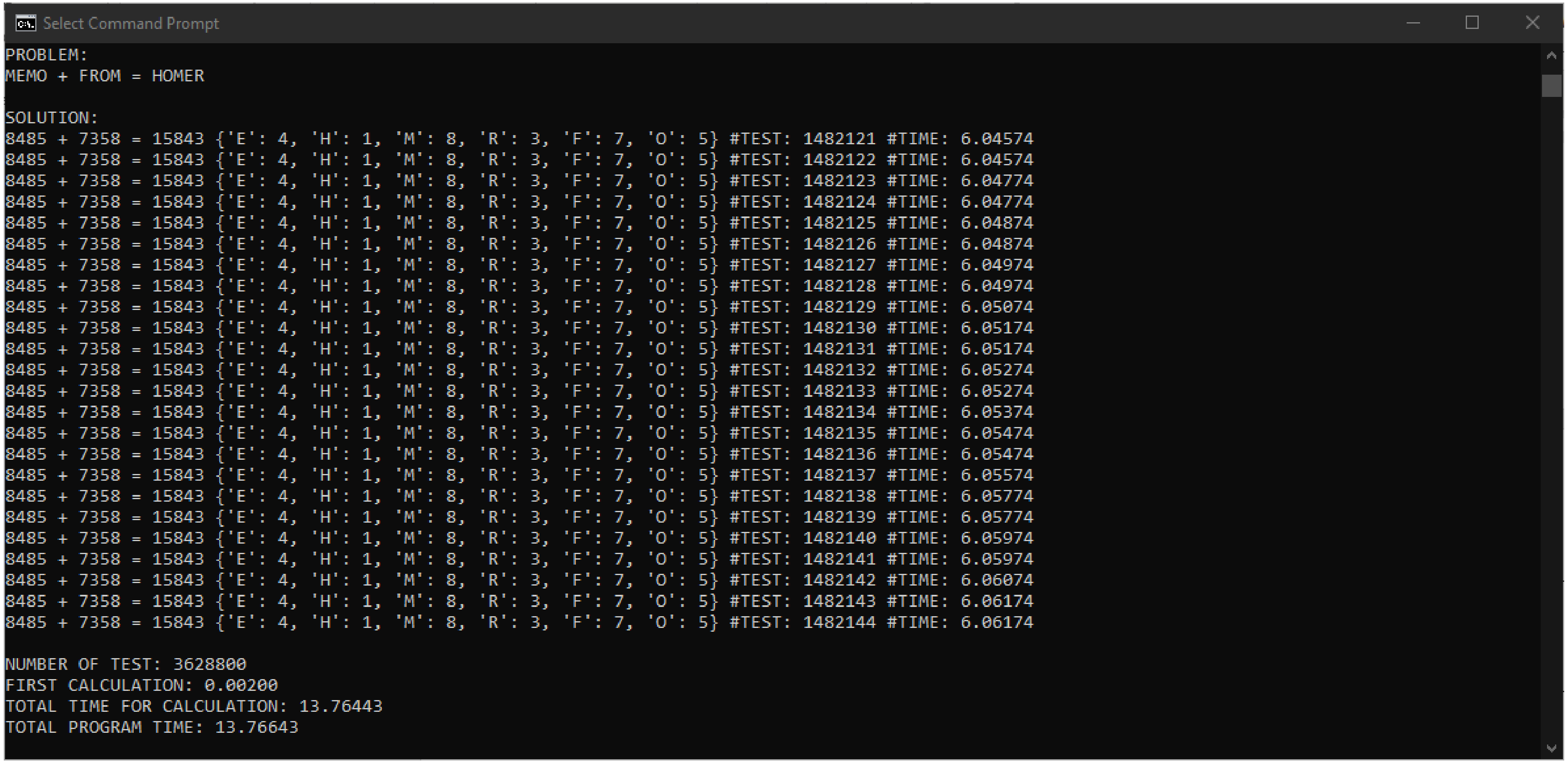


```
PROBLEM:
CROSS + ROADS = DANGER

SOLUTION:
96233 + 62513 = 158746 {'N': 8, 'S': 3, 'G': 7, 'E': 4, 'D': 1, 'R': 6, 'O': 2, 'C': 9, 'A': 5} #TEST: 3009636 #TIME: 13.91922

NUMBER OF TEST: 3628800
FIRST CALCULATION: 0.00100
TOTAL TIME FOR CALCULATION: 16.88849
TOTAL PROGRAM TIME: 16.88949
```

INPUT: (sample8.txt)

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

OUTPUT:

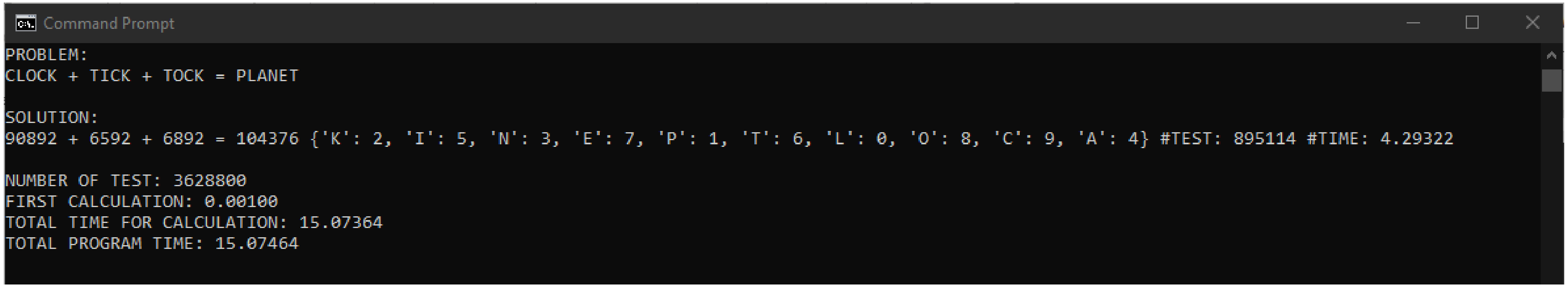


```
PROBLEM:
MEMO + FROM = HOMER

SOLUTION:
8485 + 7358 = 15843 {'E': 4, 'H': 1, 'M': 8, 'R': 3, 'F': 7, 'O': 5} #TEST: 1482121 #TIME: 6.04574
8485 + 7358 = 15843 {'E': 4, 'H': 1, 'M': 8, 'R': 3, 'F': 7, 'O': 5} #TEST: 1482122 #TIME: 6.04574
8485 + 7358 = 15843 {'E': 4, 'H': 1, 'M': 8, 'R': 3, 'F': 7, 'O': 5} #TEST: 1482123 #TIME: 6.04774
8485 + 7358 = 15843 {'E': 4, 'H': 1, 'M': 8, 'R': 3, 'F': 7, 'O': 5} #TEST: 1482124 #TIME: 6.04774
8485 + 7358 = 15843 {'E': 4, 'H': 1, 'M': 8, 'R': 3, 'F': 7, 'O': 5} #TEST: 1482125 #TIME: 6.04874
8485 + 7358 = 15843 {'E': 4, 'H': 1, 'M': 8, 'R': 3, 'F': 7, 'O': 5} #TEST: 1482126 #TIME: 6.04874
8485 + 7358 = 15843 {'E': 4, 'H': 1, 'M': 8, 'R': 3, 'F': 7, 'O': 5} #TEST: 1482127 #TIME: 6.04974
8485 + 7358 = 15843 {'E': 4, 'H': 1, 'M': 8, 'R': 3, 'F': 7, 'O': 5} #TEST: 1482128 #TIME: 6.04974
8485 + 7358 = 15843 {'E': 4, 'H': 1, 'M': 8, 'R': 3, 'F': 7, 'O': 5} #TEST: 1482129 #TIME: 6.05074
8485 + 7358 = 15843 {'E': 4, 'H': 1, 'M': 8, 'R': 3, 'F': 7, 'O': 5} #TEST: 1482130 #TIME: 6.05174
8485 + 7358 = 15843 {'E': 4, 'H': 1, 'M': 8, 'R': 3, 'F': 7, 'O': 5} #TEST: 1482131 #TIME: 6.05174
8485 + 7358 = 15843 {'E': 4, 'H': 1, 'M': 8, 'R': 3, 'F': 7, 'O': 5} #TEST: 1482132 #TIME: 6.05274
8485 + 7358 = 15843 {'E': 4, 'H': 1, 'M': 8, 'R': 3, 'F': 7, 'O': 5} #TEST: 1482133 #TIME: 6.05274
8485 + 7358 = 15843 {'E': 4, 'H': 1, 'M': 8, 'R': 3, 'F': 7, 'O': 5} #TEST: 1482134 #TIME: 6.05374
8485 + 7358 = 15843 {'E': 4, 'H': 1, 'M': 8, 'R': 3, 'F': 7, 'O': 5} #TEST: 1482135 #TIME: 6.05474
8485 + 7358 = 15843 {'E': 4, 'H': 1, 'M': 8, 'R': 3, 'F': 7, 'O': 5} #TEST: 1482136 #TIME: 6.05474
8485 + 7358 = 15843 {'E': 4, 'H': 1, 'M': 8, 'R': 3, 'F': 7, 'O': 5} #TEST: 1482137 #TIME: 6.05574
8485 + 7358 = 15843 {'E': 4, 'H': 1, 'M': 8, 'R': 3, 'F': 7, 'O': 5} #TEST: 1482138 #TIME: 6.05774
8485 + 7358 = 15843 {'E': 4, 'H': 1, 'M': 8, 'R': 3, 'F': 7, 'O': 5} #TEST: 1482139 #TIME: 6.05774
8485 + 7358 = 15843 {'E': 4, 'H': 1, 'M': 8, 'R': 3, 'F': 7, 'O': 5} #TEST: 1482140 #TIME: 6.05974
8485 + 7358 = 15843 {'E': 4, 'H': 1, 'M': 8, 'R': 3, 'F': 7, 'O': 5} #TEST: 1482141 #TIME: 6.05974
8485 + 7358 = 15843 {'E': 4, 'H': 1, 'M': 8, 'R': 3, 'F': 7, 'O': 5} #TEST: 1482142 #TIME: 6.06074
8485 + 7358 = 15843 {'E': 4, 'H': 1, 'M': 8, 'R': 3, 'F': 7, 'O': 5} #TEST: 1482143 #TIME: 6.06174
8485 + 7358 = 15843 {'E': 4, 'H': 1, 'M': 8, 'R': 3, 'F': 7, 'O': 5} #TEST: 1482144 #TIME: 6.06174

NUMBER OF TEST: 3628800
FIRST CALCULATION: 0.00200
TOTAL TIME FOR CALCULATION: 13.76443
TOTAL PROGRAM TIME: 13.76643
```

INPUT: (sample9.txt)

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

OUTPUT:



```
PROBLEM:
CLOCK + TICK + TOCK = PLANET

SOLUTION:
90892 + 6592 + 6892 = 104376 {'K': 2, 'I': 5, 'N': 3, 'E': 7, 'P': 1, 'T': 6, 'L': 0, 'O': 8, 'C': 9, 'A': 4} #TEST: 895114 #TIME: 4.29322

NUMBER OF TEST: 3628800
FIRST CALCULATION: 0.00100
TOTAL TIME FOR CALCULATION: 15.07364
TOTAL PROGRAM TIME: 15.07464
```

LINK SOURCE CODE: https://github.com/Ardovigus/TUCIL_1_IF2211_13519198

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 *cryptarihtmetic* dengan dua buah *operand*.	✓✓✓✓✓✓✓	
5. Solusi *cryptarithmetic* benar untuk persoalan *cryptarihtmetic* untuk lebih dari dua buah operand.	✓✓✓✓✓✓✓	

*Jika kompilasi diartikan membuat .exe, maka **TIDAK**, jika sebatas sintaks maka **YA**