

# **CRYPTHARITMETIC SOLVER**

**LAPORAN TUGAS KECIL 1**

**MATA KULIAH IF2211 STRATEGI ALGORITMA**



Disusun oleh: Aria Bachrul Ulum Berlian (13519115)

**Program Studi Teknik Informatika  
Sekolah Teknik Elektro dan Informatika  
Institut Teknologi Bandung  
Semester 2 Tahun 2020/2021**

## Algoritma Cryptarithmic

Program cryptarithmic ini menggunakan algoritma brute force. Program ini awalnya membaca file txt. Lalu, program ini akan membersihkan data tersebut dan memasukkannya ke dalam larik. Program akan mulai mencatat waktu. Dari array of character tadi, program membuat larik baru yang berisi huruf-huruf yang unik. Lalu, dibuat larik berisi angka dari 0 hingga 9. Dibuatlah sebuah kamus yang *key*-nya adalah elemen dari larik huruf unik dan *value*-nya adalah elemen dari larik berisi angka. Selanjutnya, larik data dari file akan diubah menurut kamus. Bila menurut aturan penjumlahan data ini sudah benar maka akan ditampilkan ke layar. Jika tidak, maka larik angka akan diacak lalu pembuatan kamus diulang lagi. Pengecekan dilakukan lagi dan seterusnya.

Pengacakan angka dilakukan dengan cara permutasi. Hal ini dilakukan agar tidak ada susunan angka yang berulang. Semua susunan akan dicoba satu per satu hingga solusi yang memenuhi persamaan aritmatikanya. Banyaknya percobaan maksimal, yakni  $nPr$ , untuk  $n$  adalah jumlah kemungkinan angka, dan  $r$  adalah jumlah huruf yang berbeda.

## SOURCE CODE

```
from time import time

def permutasi(arr):
    if len(arr) == 0:
        return []
    if len(arr) == 1:
        return [arr]
    l = []
    for i in range(len(arr)):
        N = arr[i]
        sisa = arr[:i] + arr[i+1:]
        for p in permutasi(sisa):
            l.append([N] + p)
    return l

def nilai(kata, dict):
    total = 0
    s = 1
    balikin = kata[::-1]
    for i in range(len(balikin)):
        total += dict[balikin[i]] * s
        s *= 10
    return total

def print_solusi(solusi):
    n = len(solusi)
    for i in range(n-2):
        print(solusi[i])
    print("{} {}".format(solusi[n-2]))
    print("-----")
    print(solusi[n-1])

soal = input("Masukkan nama file lengkap dengan direktorinya: ")

f = open(soal, "r")
isi = f.read()
data = isi.split("\n")
idxMax = len(data) - 1
data.pop(idxMax - 1)

for i in range(idxMax):
    data[i] = data[i].replace("+", " ").strip() #Bersihkan data
    data[i] = list(data[i])

f.close()

start = time() # Mulai hitung waktu
```

```

angka = list(range(10))    #Kemungkinan angka: 0-9

# Semua huruf yang ada dalam string
char_list = []
for i in range(idxMax):
    n = len(data[i])
    for j in range(n):
        char_list.append(data[i][j])
char_list = list(set(char_list))

coba = 0    # Menghitung jumlah percobaan

for tes in permutasi(angka):
    char_dict = dict(zip(char_list, tes[::-1]))    # {char:angka}

    nol = 0
    for i in range(len(data)):
        if(char_dict[data[i][0]] == 0) :
            nol += 1
    if nol != 0:
        continue

    else:
        hasil = 0
        jawaban = []

        for i in range(len(data)-1):
            hasil += nilai(data[i],char_dict)
            jawaban.append(nilai(data[i],char_dict))

        if (hasil == nilai(data[-1], char_dict)):
            jawaban.append(nilai(data[-1], char_dict))
            end = time()
            print(isi)
            print()
            print_solusi(jawaban)
            print()
            print("Jumlah percobaan :", coba)
            print("Waktu yang dibutuhkan :", end-start,"detik")
            break
    coba += 1

```

## PENGUJIAN

```
D:\Kuliahan\SMT 4\Strategi Algoritma\Tucil>python -u "d:\Kuliahan\SMT 4\Strategi Algoritma\Tucil\1\tugas_kecil.py"
NUMBER
NUMBER +
-----
PUZZLE

201689
201689 +
-----
403378

Jumlah percobaan : 1627618
Waktu yang dibutuhkan : 40.395331621170044 detik
```

```
D:\Kuliahan\SMT 4\Strategi Algoritma\Tucil>python -u "d:\Kuliahan\SMT 4\Strategi Algoritma\Tucil\1\tugas_kecil.py"
TILES
PUZZLES +
-----
PICTURE

91542
3077542 +
-----
3169084

Jumlah percobaan : 2682648
Waktu yang dibutuhkan : 53.763702392578125 detik
```

```
D:\Kuliahan\SMT 4\Strategi Algoritma\Tucil>python -u "d:\Kuliahan\SMT 4\Strategi Algoritma\Tucil\1\tugas_kecil.py"
CLOCK
TICK
TOCK +
-----
PLANET

90892
6592
6892 +
-----
104376

Jumlah percobaan : 1422544
Waktu yang dibutuhkan : 38.59207057952881 detik
```

```
D:\Kuliahan\SMT 4\Strategi Algoritma\Tucil>python -u "d:\Kuliahan\SMT 4\Strategi Algoritma\Tucil\1\tugas_kecil.py"
COCA
COLA +
-----
OASIS

8186
8106 +
-----
16292

Jumlah percobaan : 1042639
Waktu yang dibutuhkan : 29.254515171051025 detik
```

```
D:\Kuliahan\SMT 4\Strategi Algoritma\Tucil>python -u "d:\Kuliahan\SMT 4\Strategi Algoritma\Tucil\1\tugas_kecil.py"
HERE
SHE +
-----
COMES

9454
894 +
-----
10348

Jumlah percobaan : 763659
Waktu yang dibutuhkan : 25.833743810653687 detik
```

```
D:\Kuliah\SMT 4\Strategi Algoritma\Tucil>python -u "d:\Kuliah\SMT 4\Strategi Algoritma\Tucil\1\tugas_kecil.py"
DOUBLE
DOUBLE
TOIL +
-----
TROUBLE

798064
798064
1936 +
-----
1598064

Jumlah percobaan : 717026
Waktu yang dibutuhkan : 29.557260751724243 detik
```

```
D:\Kuliah\SMT 4\Strategi Algoritma\Tucil>python -u "d:\Kuliah\SMT 4\Strategi Algoritma\Tucil\1\tugas_kecil.py"
NO
GUN
NO +
-----
HUNT

87
908
87 +
-----
1082

Jumlah percobaan : 949960
Waktu yang dibutuhkan : 29.111649751663208 detik
```

```
D:\Kuliah\SMT 4\Strategi Algoritma\Tucil>python -u "d:\Kuliah\SMT 4\Strategi Algoritma\Tucil\1\tugas_kecil.py"
THREE
THREE
TWO
TWO
ONE +
-----
ELEVEN

84611
84611
803
803
391 +
-----
171219

Jumlah percobaan : 1504636
Waktu yang dibutuhkan : 49.76694321632385 detik
```

```
D:\Kuliah\SMT 4\Strategi Algoritma\Tucil>python -u "d:\Kuliah\SMT 4\Strategi Algoritma\Tucil\1\tugas_kecil.py"
CROSS
ROADS +
-----
DANGER

96233
62513 +
-----
158746

Jumlah percobaan : 168417
Waktu yang dibutuhkan : 20.546435117721558 detik
```

```
D:\Kuliah\SMT 4\Strategi Algoritma\Tucil>python -u "d:\Kuliah\SMT 4\Strategi Algoritma\Tucil\1\tugas_kecil.py"
MEMO
FROM +
-----
HOMER

8485
7358 +
-----
15843

Jumlah percobaan : 65455
Waktu yang dibutuhkan : 18.915944576263428 detik
```

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

<https://github.com/ariaberlian/cryptarithmic>