

**LAPORAN TUGAS KECIL I**  
**IF2211 STRATEGI ALGORITMA**  
**Semester II Tahun 2020/2021**



Disusun oleh:

<b>Nama</b>	<b>Arjuna Marcelino</b>
<b>NIM</b>	<b>13519021</b>
<b>Kelas</b>	<b>01</b>

Program Studi Teknik Informatika  
Sekolah Teknik Elektro dan Informatika  
Institut Teknologi Bandung  
2021

## Daftar Isi

<b>I. Deskripsi Masalah : Algoritma <i>Brute Force</i></b>	2
<b>II. <i>Source</i> Program</b>	2
<b>III. Eksperimen</b>	7
<b>IV. Komponen Penilaian</b>	15
<b>Lampiran</b>	16

## I. Deskripsi Masalah : Algoritma *Brute Force*

Sebuah persoalan *cryptarithmic* dapat diselesaikan dengan menggunakan algoritma *brute-force*. Suatu solusi dari *cryptarithmic* akan didapatkan dengan mengiterasi setiap kemungkinan dari kombinasi huruf.

Pengerjaan dengan algoritma *brute-force* adalah sebagai berikut.

1. Gabungkan setiap huruf yang muncul baik dalam soal maupun solusi tanpa terjadinya pengulangan ke dalam satu list.
2. Nyatakan setiap huruf tersebut sebagai suatu angka dengan cara melakukan permutasi dari 10 unsur (0-9).
3. Cek kebenaran dari penjumlahan dua buah kata tersebut dengan kata solusi yang ada.
4. Apabila hasil belum ditemukan (belum sesuai harapan), ulangi langkah sampai 3 dengan memilih hasil permutasi yang berbeda dengan yang telah dicoba sampai menemukan solusinya

Berikut langkah-langkah pengerjaan dengan contoh.

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

Pertama, setiap huruf yang muncul digabungkan ke dalam satu list dan huruf bersifat unik, yaitu SENDMORY.

Lalu, untuk setiap karakter inisialisasi dengan nilai hasil permutasi dari 10 unsur (0-9)

```
SENDMORY
0 1 2 3 4 5 6 7 8 9
```

Cek apakah penjumlahan dua buah kata SEND + MORE sudah benar atau belum.

```
0123
4561 +
-----
45217
```

Karena penjumlahan belum benar, maka dilakukan pengecekan terhadap representasi nilai karakter berikutnya

SENDMORY

0 1 2 3 4 5 6 8 7 9

Lakukan pengecekan kembali apakah penjumlahan angka sudah sesuai dengan solusi nya (nilai representasi baru).

Penjumlahan masih akan terus berlanjut mengikuti hasil dari permutasi dari 10 unsur (0-9) dan dilakukan iterasi penambahan representasi secara terus menerus hingga didapat nilai penjumlahan yang sesuai dan nilai yang mengikuti aturan yang sesuai juga, seperti tidak ada angka yang muncul lebih dari satu kali, representasi tidak diawali 0).

## II. Source Program

```
import time

# fungsi untuk mencari nilai sebuah permutasi
def permutasi (elemen_list):
    # best case mengembalikan elemen tunggal
    if len(elemen_list) == 1:
        return [elemen_list]
    # rekurens
    else :
        temp =[]
        # melanjutkan ke elemen berikutnya
        for i in range (len(elemen_list)):
            a = elemen_list[i]
            sisaList = elemen_list[:i] + elemen_list[i+1:]
            for j in permutasi (sisaList):
                temp.append([a]+j)
        return temp

# fungsi main
print("#####")
print("#          --      --          --          --          #")
print("#          |  |  |  |          |  |          |  |          #")
print("#          |  |  |  |          |  |          |  |          #")
print("#          |  --  |          |  |          |  |          #")
print("#          |  --  |          |  |          |  |          #")
print("#          |  |  |  |          |  |          --          #")
print("#          |  |  |  |          |  |          /\          #")
print("#          --      --          --          \/\          #")
print("#####")
print("#                                          #")
print("#          CRYPTARITHMETIC SOLVER          #")
print("#                                          #")
print("#          ARJUNA MARCELINO          13519021          K01          #")
print("#                                          #")
print("#####")
print("")
print("File yang diterima hanya dengan ekstensi \".txt\"")

# menerima masukan nama file
soal = input("Masukkan nama file teks (dengan ekstensi) : ")

# buka file
file_soal = open("../test/"+soal,"r")

# baca isi file
baca_soal = file_soal.read()
```

```
# mulai menghitung waktu
start = time.time()

# membersihkan karakter '+', '-', '\n'
car_bersih = baca_soal
car_bersih = car_bersih.replace('+','')
car_bersih = car_bersih.replace('-','')
car_bersih = car_bersih.replace('\n','')

# menyimpan kata
kata = car_bersih.split()

# menyimpan huruf berbeda
huruf = car_bersih.replace(' ','')
huruf = list(dict.fromkeys(huruf))

# menentukan banyaknya operan
n_operan = len(kata)-1

cek = 0
# mencari solusi yang benar dengan mencoba semua kemungkinan permutasi
found = False
data = list('1234567890')
for p in permutasi(data):
    sum = 0
    nilai_hasil = 0
    nilai_operan = [0 for i in range (n_operan)]
    nol = False
    for i in range (n_operan) :
        digit = len(kata[i])-1
        for curr in kata[i] :
            idx = huruf.index(curr)
            nilai_operan[i] += int(p[idx])*(10**digit)
            digit-=1
        sum += nilai_operan[i]
        if (int(p[huruf.index(kata[i][0]))]==0):
            nol = True

    digit_hasil = len(kata[n_operan])-1
    for curr in kata[n_operan] :
        indx = huruf.index(curr)
        nilai_hasil += int(p[indx])*(10**digit_hasil)
        digit_hasil-=1
    if (int(p[huruf.index(kata[n_operan][0]))]==0):
        nol = True
```

```
    if (sum == nilai_hasil) and (not (nol)) :
        found = True
        break

    cek +=1

end = time.time()
# menampilkan output
print("")
print(baca_soal)

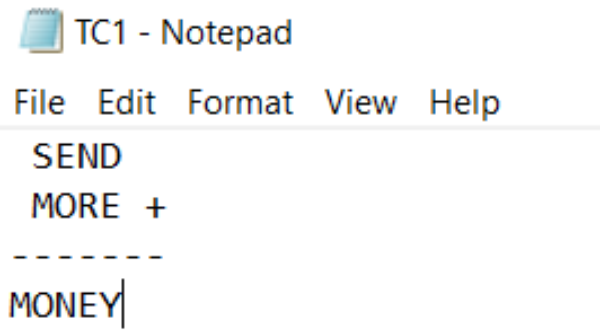
if (not(found)):
    print("\nTidak ada solusi untuk menyelesaikan permasalahan Cryptarithmic ini"
)
else :
    print("\nSolusi : \n")
    for i in range (n_operan-1):
        spasi = len(kata[n_operan])-len(kata[i])
        print(' '*spasi+str(nilai_operan[i]))

    spasi = len(kata[n_operan])-len(kata[n_operan-1])
    print(' '*spasi+str(nilai_operan[n_operan-1])+' +')
    print('-'*(len(kata[n_operan])+2))
    print(nilai_hasil)
    print("\nWaktu eksekusi program : "+ str(end - start),"detik")
    print("\nJumlah tes yang dilakukan untuk menemukan substitusi angka yang benar
untuk setiap huruf : ",cek,"kali")

# tutup file
file_soal.close()
```


### III. Eksperimen

#### Contoh 1


Input

Output
<pre>File yang diterima hanya dengan ekstensi ".txt" Masukkan nama file teks (dengan ekstensi) : TC1.txt  SEND MORE + ----- MONEY  Solusi :  9567 1085 + ----- 10652  Waktu eksekusi program : 38.51883625984192 detik  Jumlah tes yang dilakukan untuk menemukan substitusi angka yang benar untuk setiap huruf : 3087474 kali</pre>



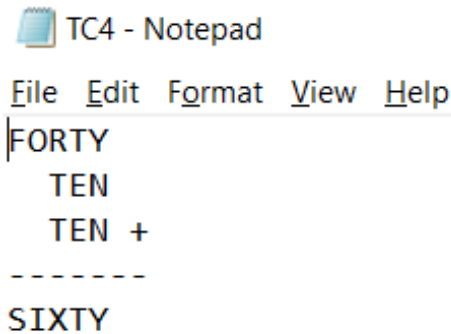
## Contoh 2

Input
 TC2 - Notepad File Edit Format View Help NUMBER NUMBER + ----- PUZZLE
Output
<pre>File yang diterima hanya dengan ekstensi ".txt" Masukkan nama file teks (dengan ekstensi) : TC2.txt  NUMBER NUMBER + ----- PUZZLE  Solusi :  201689 201689 + ----- 403378  Waktu eksekusi program : 19.685892343521118 detik  Jumlah tes yang dilakukan untuk menemukan substitusi angka yang benar untuk setiap huruf : 688183 kali</pre>


### Contoh 3

Input
 TC3 - Notepad File Edit Format View Help   NO GUN NO + ----- HUNT
Output
<pre>File yang diterima hanya dengan ekstensi ".txt" Masukkan nama file teks (dengan ekstensi) : TC3.txt  NO GUN NO + ----- HUNT  Solusi :  87 908 87 + ----- 1082  Waktu eksekusi program : 33.62595796585083 detik  Jumlah tes yang dilakukan untuk menemukan substitusi angka yang benar untuk setiap huruf : 2816640 kali</pre>


#### Contoh 4

Input	
	
Output	<pre>File yang diterima hanya dengan ekstensi ".txt" Masukkan nama file teks (dengan ekstensi) : TC4.txt  FORTY   TEN   TEN + ----- SIXTY  Solusi :  29786   850   850 + ----- 31486  Waktu eksekusi program : 19.83083987236023 detik  Jumlah tes yang dilakukan untuk menemukan substitusi angka yang benar untuk setiap huruf : 674492 kali</pre>


### Contoh 5

Input
 TC5 - Notepad <u>F</u> ile <u>E</u> dit <u>F</u> ormat <u>V</u> iew <u>H</u> elp DOUBLE DOUBLE TOIL + ----- TROUBLE
Output
<pre>File yang diterima hanya dengan ekstensi ".txt" Masukkan nama file teks (dengan ekstensi) : TC5.txt  DOUBLE DOUBLE TOIL + ----- TROUBLE  Solusi :  798064 798064 1936 + ----- 1598064  Waktu eksekusi program : 50.30888867378235 detik  Jumlah tes yang dilakukan untuk menemukan substitusi angka yang benar untuk setiap huruf : 2494755 kali</pre>


### Contoh 6

Input
 TC6 - Notepad <u>F</u> ile <u>E</u> dit <u>F</u> ormat <u>V</u> iew <u>H</u> elp TILES PUZZLES + ----- PICTURE
Output
<pre>File yang diterima hanya dengan ekstensi ".txt" Masukkan nama file teks (dengan ekstensi) : TC6.txt    TILES PUZZLES + ----- PICTURE  Solusi :    91542 3077542 + ----- 3169084  Waktu eksekusi program : 50.55449676513672 detik  Jumlah tes yang dilakukan untuk menemukan substitusi angka yang benar untuk setiap huruf : 2919620 kali</pre>

### Contoh 7

Input
 TC7 - Notepad <u>F</u> ile <u>E</u> dit <u>F</u> ormat <u>V</u> iew <u>H</u> elp HERE SHE + ----- COMES
Output
<pre>File yang diterima hanya dengan ekstensi ".txt" Masukkan nama file teks (dengan ekstensi) : TC7.txt  HERE SHE + ----- COMES  Solusi :  9454 894 + ----- 10348  Waktu eksekusi program : 35.97333002090454 detik  Jumlah tes yang dilakukan untuk menemukan substitusi angka yang benar untuk setiap huruf : 3042822 kali</pre>

### Contoh 8

Input
 TC8 - Notepad <u>F</u> ile <u>E</u> dit <u>F</u> ormat <u>V</u> iew <u>H</u> elp THREE THREE TWO TWO ONE + ----- ELEVEN
Output
<pre>File yang diterima hanya dengan ekstensi ".txt" Masukkan nama file teks (dengan ekstensi) : TC8.txt  THREE THREE   TWO   TWO ONE + ----- ELEVEN  Solusi :  84611 84611   803   803   391 + ----- 171219  Waktu eksekusi program : 57.28895139694214 detik  Jumlah tes yang dilakukan untuk menemukan substitusi angka yang benar untuk setiap huruf : 2681926 kali</pre>

## IV. Komponen Penilaian

Poin	Ya	Tidak
1. Program berhasil dikompilasi tanpa kesalahan ( <i>no syntax error</i> )	✓	
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 operand.		✓
5. Solusi <i>cryptarithmic</i> benar untuk persoalan <i>cryptarithmic</i> untuk lebih dari dua buah operand.	✓	



## **Lampiran**

File/repository dari Tugas Kecil 1 ini dapat juga diakses di pranala berikut ini :

[https://github.com/arjunamarcelino/Tucil1\\_13519021](https://github.com/arjunamarcelino/Tucil1_13519021)