

Laporan Tugas Kecil 1 IF2211 Strategi Algoritma  
Penyelesaian *Cryptarithmic* dengan Algoritma *Brute Force*



(Nama- NIM)  
Girvin Junod - 13519096

Semester II tahun 2020/2021

## Algoritma *Bruteforce*

Algoritma *brute force* ini sederhananya adalah mensubstitusikan tiap huruf dengan angka yang unik lalu mencoba memasukannya ke perhitungan soal. Jika substitusi angkanya semua operand yang lalu ditambahkan hasilnya cocok dengan substitusi hasil maka substitusi angka benar. Untuk substitusinya, karena menggunakan *brute force* maka akan dicoba semua kemungkinan substitusi angka sampai benar.

Pertama ada *input* dulu yaitu soal dibaca dari file .txt. Setelah *input* ada penyaringan *input* agar terbebas dari spasi dan '+' lalu disimpan dulu kata operand dalam array operand dan kata hasil dalam variabel hasil. Lalu dari *input* akan disimpan juga semua huruf unik ke array huruf unik. Dari array-array ini jadi diketahui jumlah huruf unik dan operand. Ada pengecekan jumlah huruf unik dan operand, jika huruf unik > 10 atau operand < 2 maka program akan berhenti karena soal tidak memenuhi spesifikasi.

Lalu, dibuat array substitusi angka untuk tiap huruf unik dengan jumlah elemen sebanyak jumlah huruf unik. Pada awalnya array bernilai 1 semua. Array substitusi angka ini elemennya adalah substitusi untuk huruf pada array huruf unik sesuai dengan urutan elemennya. Jadi jika pada array elemen pertamanya huruf 'R', maka nilai substitusi R adalah elemen pertama pada array substitusi angka dan begitu juga untuk tiap huruf dan angka substitusinya. Karena ini, algoritmanya tidak mangkus dan merupakan algoritma *brute force* yang *exhaustive*.

Lalu algoritma akan masuk ke loop yang lanjut sampai ditemukan solusi atau sudah habis semua kemungkinan substitusi angka. Untuk mendapatkan semua kemungkinan angka unik di array substitusi angka, dilakukan penambahan 1 ke elemen paling akhir array substitusi angka lalu di modulo dengan 10. Jadi jika sudah mencapai 10, elemen paling akhir ini akan kembali ke 0. Jika sudah mencapai 1 lagi, maka penambahan 1 ini akan dilakukan pada elemen sebelumnya dengan operasi yang sama. Jika elemen sebelum terakhir ini mencapai 1 lagi maka akan ditambahkan 1 ke elemen sebelumnya lagi dan seterusnya. Jika sudah kembali lagi ke array dengan elemen 1 semua lagi, berarti telah dicoba semua kemungkinan variasi angka sehingga tidak ada jawabannya dan program berhenti. Jadi misal,

Array substitusi awal = [1,1,1,1]  
Penambahan 1 lalu modulo 10 = [1,1,1,2]  
Jika ada [1,1,1,0], maka akan jadi [1,1,2,1]  
Jika ada [1,1,0,0], maka akan jadi [1,2,1,1]  
Dan seterusnya sampai mencapai [1,1,1,1] lagi

Tentunya banyak variasi dari array substitusi ini yang tidak memenuhi syarat nilai angka substitusi unik untuk tiap huruf. Oleh karena itu dilakukan juga pengecekan setelah itu untuk keunikan tiap elemen di array substitusi angka. Jadi, array [1,1,1,1] tidak akan lolos pengecekan ini namun array [1,2,3,4] akan lolos. Untuk array yang lolos akan, akan ada pencocokkan per kata operand terhadap array huruf dan array substitusi angka sehingga akan didapatkan nilai substitusi angka untuk kata operand itu. Lalu untuk mencegah adanya kasus didapatkan nilai substitusi yang berawal 0, dilakukan pengubahan string nilai substitusi dari string ke integer lalu ke string lagi. Jika panjang string ini berbeda dengan panjang string operand maka diketahui depannya 0 sehingga tidak lolos pengecekan. Contoh,

Didapat MONEY = 01234

01234 akan berubah menjadi 1234 ketika diubah menjadi integer

Jadi panjang string 1234  $\neq$  string MONEY sehingga tidak lolos

Substitusi kata operand menjadi nilai substitusi ini dilakukan per operand yang ditambahkan ke variabel hasil penambahan. Lalu, dilakukan juga substitusi kata hasil menjadi nilai substitusinya. Caranya sama dengan substitusi operand namun hanya dilakukan sekali karena hanya ada 1 kata hasil. Setelah itu, dilakukan pencocokkan nilai substitusi kata hasil dengan variabel hasil penambahan yang didapat dari penambahan nilai substitusi operand. Jika cocok, maka sudah ditemukan jawabannya dan loop dihentikan lalu dikeluarkan *output* sesuai spesifikasi. Jika tidak, maka akan kembali ke awal loop dan mencoba lagi dengan variasi array substitusi angka yang lain. Contoh yang benar adalah,

Array huruf unik = [S, E, N, D, M, O, R, Y]

Array substitusi angka = [9, 5, 6, 7, 1, 0, 8, 2]

Ada 2 operand yaitu SEND dan MORE

SEND = 9567

MORE = 1085

Hasil penambahannya = 10652

Kata hasil MONEY

MONEY = 10652

10652 = 10652 sehingga substitusi angka benar

Jika sudah mencoba semua kemungkinan substitusi angka dan gagal maka tidak ada solusi untuk soal. Jika menemukan solusinya maka solusi di-print sesuai format spesifikasi sehingga keluar *output*. Selain itu juga dilakukan perhitungan waktu eksekusi program dengan python dan jumlah tes untuk menemukan substitusi benar.

## Source Program (Python)

```
import time

f = open("soal2.txt").#baca input
read = f.read().split('\n').#diread dan pembersihan input agar bisa diolah
awalwaktu = time.perf_counter().#waktu mulai eksekusi program
wordcount = len(read) - 2.#jumlah operand
wordarray = ['* ' for i in range(wordcount)].#array operand
for i in range(len(read)):
    read[i] = read[i].replace(' ','').replace('+','').#membersihkan text input dari spasi dan +
for i in range(wordcount):
    wordarray[i] = read[i].#memasukkan operand ke array operand
result = read[len(read) - 1].#memasukkan kata hasil
letterarray = [].#array huruf unik awalnya kosong
for i in read:
    for j in i:
        if j != '-':
            if len(letterarray) == 0:
                letterarray.append(j).#jika array masih kosong langsung append
            else:
                unik = True
                for i in range(len(letterarray)):
                    if letterarray[i] == j: #pengecekan huruf yang sama
                        unik = False
                        break
                if unik:
                    letterarray.append(j).#memasukkan huruf unik ke array
lettercount = len(letterarray)

jumlahtotaltes = 0
found = False
count = 0.#buat break loop
lettervaluearray = [1 for i in range(lettercount)]
hasilarray = [0 for j in range(wordcount)]
maxlength = len(result).#karena panjang kata result pasti lebih besar dari yang lain, hanya untuk kebutuhan mempercantik output aja si ini
if len(letterarray) > 10: #batas huruf di operand 10
    found = True
    print('Jumlah huruf unik melebihi 10')
if wordcount < 2:
    found = True
    print('Jumlah operand kurang dari 2')
while found != True:
    n = lettercount-1.#digit terakhir array value huruf
    for i in range(lettercount):#bermula misal ada 3 digit 1 -> 111, akan ditambah 1 terus sampai habis semua kemungkinan value huruf
        lettervaluearray[n] = (lettervaluearray[n] + 1) % 10
        if lettervaluearray[n] == 1:
            n = n-1.#agar bisa manipulasi digit sebelum yang terakhir
        else:
            break #loopnya agar bisa manipulasi semua bagian array, kalau tidak perlu maka dibreak saja
    jumlahtotaltes+=1.#dianggap jumlah total tes mencangkupi tes yang tidak memenuhi syarat huruf unik atau huruf pertama 0
    if lettervaluearray[0] != 1:
        count = 1
    if count == 1 and lettervaluearray[0] == 1: #sudah menghabiskan semua kemungkinan
        print("Tidak ada hasil")
        break
    hasil = 0
    valid = True.#agar tidak perlu banyak perhitungan yang tidak perlu, dibuat pengecekan validitas
    for i in range(lettercount):
        for j in range(lettercount):
            if lettervaluearray[i] == lettervaluearray[j] and i!=j: #pengecekan keunikan value huruf
                valid = False
                break
    if valid: #jika semua nilai huruf unik
        for k in range(wordcount): #dibuat per operand
            stringnilai = ''
            for i in wordarray[k]:
```

```

        for j in range(lettercount):
            if i == letterarray[j]:
                stringnilai += str(lettervaluearray[j])
        stringnilai = int(stringnilai)
        stringnilai = str(stringnilai) #buat eliminasi yg awalnya 0
        if len(stringnilai) == len(wordarray[k]):
            hasil += int(stringnilai) #penambahan tiap operand
            hasilarray[k] = int(stringnilai) #nilai tiap operand disimpan
        else:
            valid = False
    if valid: #jika tidak valid tidak dilihat lagi
        stringresult = ''
        for i in result:
            for j in range(lettercount):
                if i == letterarray[j]:
                    stringresult += str(lettervaluearray[j])
        stringresult = int(stringresult)
        stringresult = str(stringresult)
        if (len(stringresult) == len(result)): #eliminasi yang awalnya 0
            if stringresult == str(hasil): #nilai cocok ketemu
                for l in range(wordcount): #formatting output
                    spasi = ' '
                    if l == wordcount-1:
                        spasi = '+ '
                    if len(wordarray[l]) < maxlength:
                        spasi += ' '*(maxlength-len(wordarray[l]))
                    print(spasi + wordarray[l], end=' ') #output soal
                    print(spasi + str(hasilarray[l]), end=' ') #output angka
                print(' ' + (maxlength+2)*'-' + ' ' + (maxlength+2)*'-' )
                print(' ' + result, end=' ')
                print(' ' + stringresult)
                found = True
                break

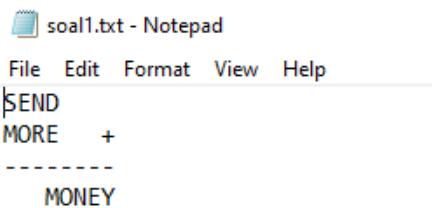
akhirwaktu = time.perf_counter() #program selesai waktu akhir program
print('') #tambah newline biar cantik yv
print("Waktu eksekusi program = " + str(akhirwaktu - awalwaktu) + ' detik') #hitung lama eksekusi program
print('Jumlah total tes = ' + str(jumlahtotaltes)) #hitung total tes angka

```

## Screenshot *input* dan *output*

1.

Input:

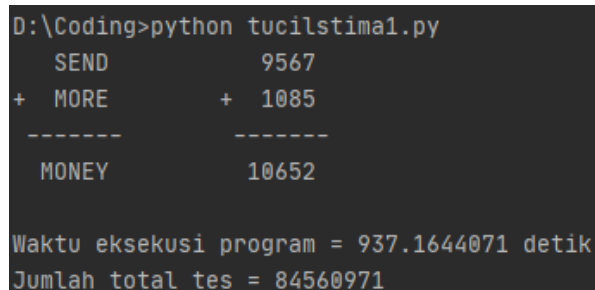


```

soal1.txt - Notepad
File Edit Format View Help
SEND
MORE +
-----
MONEY

```

Output:



```

D:\Coding>python tucilstima1.py
SEND          9567
+ MORE        + 1085
-----
MONEY         10652

Waktu eksekusi program = 937.1644071 detik
Jumlah total tes = 84560971

```

2.

Input:

```
soal2.txt - Notepad
File Edit Format View Help
      NO
      GUN
+     NO
-----
      HUNT|
```

Output:

```
D:\Coding>python tucilstima1.py
      NO          87
      GUN        908
+     NO          + 87
-----
      HUNT        1082

Waktu eksekusi program = 7.4582123000000005 detik
Jumlah total tes = 768901
```

3.

Input:

```
soal3.txt - Notepad
File Edit Format View Help
      MEMO
      FROM +
-----
      HOMER
```

Output:

```
D:\Coding>python tucilstima1.py
      MEMO        8485
+     FROM          + 7358
-----
      HOMER        15843

Waktu eksekusi program = 6.8046151 detik
Jumlah total tes = 734620
```

4.

Input:

soal4.txt - Notepad

File	Edit	Format	View	Help
HERE				
+ SHE				
-----				
COMES				

Output:

```
D:\Coding>python tucilstima1.py
  HERE          9454
+  SHE          +   894
-----
  COMES        10348

Waktu eksekusi program = 82.0090671 detik
Jumlah total tes = 8347092
```

5.

Input:

soal12.txt - Notepad

File	Edit	Format	View	Help
CRACK				
HACK +				
-----				
ERROR				

Output:

```
D:\Coding>python tucilstima1.py
  CRACK          42641
+  HACK          +   9641
-----
  ERROR        52282

Waktu eksekusi program = 29.344710499999998 detik
Jumlah total tes = 3150847
```

6. (Kasus tidak ada solusi karena melebihi 10 huruf unik)

Input:

```
soal6.txt - Notepad
File Edit Format View Help
DOUBLESOMERANDOMWORDSTOFILLSPACEZY
DOUBLE
TOIL +
-----
TROUBLE
```

Output:

```
D:\Coding>python tucilstima1.py
Jumlah huruf unik melebihi 10

Waktu eksekusi program = 0.00023590000000000416 detik
Jumlah total tes = 0
```

7.

Input:

```
soal7.txt - Notepad
File Edit Format View Help
COCA
+COLA
-----
OASIS|
```

Output:

```
D:\Coding>python tucilstima1.py
COCA          8186
+ COLA        + 8106
-----
OASIS          16292

Waktu eksekusi program = 6.861344 detik
Jumlah total tes = 705918
```



8.

Input:

```
soal8.txt - Notepad
File Edit Format View Help
EGG
EGG      +
-----
PAGE|
```

Output:

```
D:\Coding>python tucilstima1.py
  EGG          899
+  EGG      +  899
-----
  PAGE          1798

Waktu eksekusi program = 0.10114 detik
Jumlah total tes = 7806
```

9.

Input:

```
soal5.txt - Notepad
File Edit Format View Help
THREE
THREE
TWO
TWO
ONE +
-----
ELEVEN|
```

Output:

```
D:\Coding>python tucilstima1.py
  THREE          84611
  THREE          84611
    TWO           803
    TWO           803
+   ONE      +   391
-----
  ELEVEN          171219

Waktu eksekusi program = 9216.2645961 detik
Jumlah total tes = 735092861
```

10. (Kasus banyak solusi, hanya output 1 solusi)

Input:

```
soal10.txt - Notepad
File Edit Format View Help
A
A +
--
B|
```

Output:

```
D:\Coding>python tucilstima1.py
  A          1
+ A          + 1
---          ---
  B          2

Waktu eksekusi program = 0.0010880999999999946 detik
Jumlah total tes = 1
```

11.

Input:

```
soal11.txt - Notepad
File Edit Format View Help
EINS
EINS
EINS
+EINS
-----
VIER
```

Output:

```
D:\Coding>python tucilstima1.py
  EINS          1329
  EINS          1329
  EINS          1329
+ EINS          + 1329
-----          -----
  VIER          5316

Waktu eksekusi program = 0.1940535 detik
Jumlah total tes = 21845
```

12. (Kasus tidak ada solusi, karena hanya ada solusi kalau huruf depan bernilai 0)

Input:

```
soal9.txt - Notepad
File Edit Format View Help
SOD
IS +
-----
SOLD|
```

Output:

```
D:\Coding>python tucilstima1.py
Tidak ada hasil

Waktu eksekusi program = 0.9968109 detik
Jumlah total tes = 100000
```

13. (Kasus operand kurang dari 2)

Input:

```
soal13.txt - Notepad
File Edit Format View Help
ONE +
-----
ONE
```

Output:

```
D:\Coding>python tucilstima1.py
Jumlah operand kurang dari 2

Waktu eksekusi program = 0.00016649999999999998 detik
Jumlah total tes = 0
```

**Alamat Drive**

[Drive google](#) (pakai akun std)

[Repository Github](#)

**Check List**

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 cryptarithmic hanya benar untuk persoalan cryptarithmic dengan dua buah operand.		✓
5. Solusi <i>cryptarithmic</i> benar untuk persoalan <i>cryptarithmic</i> untuk lebih dari dua buah operand.	✓	

Note: Solusi benar untuk persoalan dengan  $2 \geq$  operand.