

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



Stefanus/13519101/K-02

PROGRAM STUDI TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG

2021

BAB 1

Deskripsi Masalah

Cryptarithmic (atau *cryptarithm*) adalah sebuah *puzzle* penjumlahan di dalam matematika di mana angka diganti dengan huruf. Setiap angka direpresentasikan dengan huruf yang berbeda. Deskripsi permainan ini adalah : diberikan sebuah penjumlahan huruf, carilah angka yang merepresentasikan huruf-huruf tersebut.

Contoh :

$$\begin{array}{r} \text{S E N D} \\ + \text{M O R E} \\ \hline \text{M O N E Y} \end{array}$$

Solusinya adalah:

$$\begin{array}{r} 9 \ 5 \ 6 \ 7 \\ + 1 \ 0 \ 8 \ 5 \\ \hline 1 \ 0 \ 6 \ 5 \ 2 \end{array}$$

Jadi, S = 9, E = 5, N = 6, D = 7, N = 1, O = 0, R = 8, Y = 2

Cara penyelesaian persoalan *cryptarithmic* yang umum adalah dapat dibaca pada laman: <https://www.basic-mathematics.com/cryptarithms.html>

Berdasarkan cara penyelesaian umum ini, dibuat sebuah program sederhana dengan algoritma *brute force* yang mana memiliki definisi gaya pemrograman primitif yang mana *programmer* bergantung pada kemampuan prosesor komputer daripada menggunakan kemampuan intelegensinya untuk membuat solusi yang simpel. Disebut juga sebagai pemrograman naif yang mana relatif bagus untuk masalah kecil, namun buruk untuk masalah besar.

Oleh sebab itu, untuk tugas kecil IF2211 ini, dibuatlah sebuah program sederhana yang menggunakan algoritma *brute force* untuk menyelesaikan *cryptarithmic* sesuai dengan definisi dari *brute force* maupun *cryptarithmic*.

BAB 2

Penjelasan Algoritma

Dalam program ini, digunakan pendekatan *brute force* dalam penyelesaian programnya. Program ini melakukan beberapa tahap sebelum mendapatkan jawaban yaitu :

1. Melakukan *listing* terhadap kata dan huruf, misalnya untuk persoalan

$$\begin{array}{r} \text{S E N D} \\ + \text{ M O R E} \\ \hline \text{M O N E Y} \end{array}$$

Akan menghasilkan list kata ['SEND','MORE','MONEY'] dan list huruf ['S','E','N','D','M','O','R','Y']

2. Menghitung jumlah set huruf (tidak ada perulangan), untuk kasus "SEND+MORE=MONEY" memiliki jumlah 8 huruf.
3. Memisahkan list kata menjadi soal dan jawaban dengan menganggap kata[0] sampai kata[x-2] sebagai soal dan kata[x-1] sebagai jawaban karena jawaban hanya satu.
4. Mengubah set kata menjadi *query*, dalam kasus ini menjadi *string* "SEND+MORE=MONEY".
5. Melakukan *looping* sebanyak jumlah huruf (semisal n), terhadap variabel l_1 sampai l_n yang berisi :
 - a. Melakukan *replace* dari *query* yaitu mengubah huruf[0] dengan l_1 , huruf[1] dengan l_2 , dan seterusnya hingga semua huruf di dalam *query* digantikan dengan angka
 - b. Melakukan pemisahan *query* menjadi integer, lalu dilakukan pengecekan apakah $query[0]+query[1]+\dots=query$ terakhir. Bila salah, maka akan kembali ke poin 5 (melanjutkan looping), namun bila benar, maka akan dilakukan pencetakan kata dan *query* serta spesifikasi lainnya (waktu, jumlah pergantian digit, dll) dan dilakukan *break* agar tidak perlu looping sebanyak 10^n kali.

BAB 3

Source Code

[illegible]

[illegible]


```

all_solutions = list()
for l1 in range(9, -1, -1):
    for l2 in range(9, -1, -1):
        for l3 in range(9, -1, -1):
            for l4 in range(9, -1, -1):
                for l5 in range(9, -1, -1):
                    counter=counter+1
                    temp = query.replace(list_huruf[0],str(l1))
                    temp = temp.replace(list_huruf[1],str(l2))
                    temp = temp.replace(list_huruf[2],str(l3))
                    temp = temp.replace(list_huruf[3],str(l4))
                    temp = temp.replace(list_huruf[4],str(l5))
                    temp = temp.split("+")
                    operan = len(temp)
                    count = 0
                    for i in temp:
                        count += int(i)
                    flag = [l1,l2,l3,l4,l5]
                    flagset = set(flag)

                    if ((count - (2*int(temp[operan-1]))==0) and len(flagset)==5):
                        query = query.split("+")
                        i = 0
                        for i in range (operan-1):
                            print(query[i])
                            print("_____+")
                            print(query[operan-1])
                            print()
                        i = 0
                        for i in range (operan-1):
                            print(temp[i])
                            print("_____+")
                            print(temp[operan-1])
                            print()
                        print("banyak tes untuk menemukan substitusi angka yang benar adalah", counter)
                        print("waktu yang dibutuhkan : --- %s seconds ---" % (time.time() -
start_time))

                        break

if (jumlah_huruf==4):
    all_solutions = list()
    for l1 in range(9, -1, -1):
        for l2 in range(9, -1, -1):
            for l3 in range(9, -1, -1):
                for l4 in range(9, -1, -1):
                    counter=counter+1
                    temp = query.replace(list_huruf[0],str(l1))
                    temp = temp.replace(list_huruf[1],str(l2))
                    temp = temp.replace(list_huruf[2],str(l3))
                    temp = temp.replace(list_huruf[3],str(l4))
                    temp = temp.split("+")
                    operan = len(temp)
                    count = 0
                    for i in temp:
                        count += int(i)
                    flag = [l1,l2,l3,l4]
                    flagset = set(flag)

                    if ((count - (2*int(temp[operan-1]))==0) and len(flagset)==4):
                        query = query.split("+")
                        i = 0
                        for i in range (operan-1):
                            print(query[i])
                            print("_____+")
                            print(query[operan-1])
                            print()
                        i = 0
                        for i in range (operan-1):
                            print(temp[i])
                            print("_____+")
                            print(temp[operan-1])
                            print()
                        print("banyak tes untuk menemukan substitusi angka yang benar adalah", counter)
                        print("waktu yang dibutuhkan : --- %s seconds ---" % (time.time() - start_time))
                        break

if (jumlah_huruf==3):
    all_solutions = list()
    for l1 in range(9, -1, -1):
        for l2 in range(9, -1, -1):
            for l3 in range(9, -1, -1):
                counter=counter+1
                temp = query.replace(list_huruf[0],str(l1))
                temp = temp.replace(list_huruf[1],str(l2))
                temp = temp.replace(list_huruf[2],str(l3))
                temp = temp.split("+")
                operan = len(temp)
                count = 0
                for i in temp:
                    count += int(i)
                flag = [l1,l2,l3]
                flagset = set(flag)

                if ((count - (2*int(temp[operan-1]))==0) and len(flagset)==3):
                    query = query.split("+")
                    i = 0
                    for i in range (operan-1):
                        print(query[i])

```

```

        print("_____+")
        print(query[operan-1])
        print()
        i = 0
        for i in range (operan-1):
            print(temp[i])
        print("_____+")
        print(temp[operan-1])
        print()
        print("banyak tes untuk menemukan substitusi angka yang benar adalah", counter)
        print("waktu yang dibutuhkan : --- %s seconds ---" % (time.time() - start_time))
        break

if (jumlah_huruf==2):
    all_solutions = list()
    for l1 in range(9, -1, -1):
        for l2 in range(9, -1, -1):
            counter=counter+1
            temp = query.replace(list_huruf[0],str(l1))
            temp = temp.replace(list_huruf[1],str(l2))
            temp = temp.split("+")
            operan = len(temp)
            count = 0
            for i in temp:
                count += int(i)
            flag = [l1,l2]
            flagset = set(flag)

            if ((count - (2*int(temp[operan-1]))==0) and len(flagset)==2):
                query = query.split("+")
                i = 0
                for i in range (operan-1):
                    print(query[i])
                print("_____+")
                print(query[operan-1])
                print()
                i = 0
                for i in range (operan-1):
                    print(temp[i])
                print("_____+")
                print(temp[operan-1])
                print()
                print("banyak tes untuk menemukan substitusi angka yang benar adalah", counter)
                print("waktu yang dibutuhkan : --- %s seconds ---" % (time.time() - start_time))
                break

```


BAB 4

Test Case

Notepad	Hasil
A B+ ----- CD	<pre> PS C:\Users\Stefanus\Desktop\tucil> python tucil.py masukkan nama file : test1 A B -----+ CD 9 8 -----+ 17 banyak tes untuk menemukan substitusi angka yang benar adalah 183 waktu yang dibutuhkan : --- 0.007999897003173828 seconds --- </pre>
AB CD+ ----- EFG	<pre> PS C:\Users\Stefanus\Desktop\tucil> python tucil.py masukkan nama file : test2 AB CD -----+ EFG 96 52 -----+ 148 banyak tes untuk menemukan substitusi angka yang benar adalah 347852 waktu yang dibutuhkan : --- 3.3642499446868896 seconds --- </pre>
COCA COLA+ ----- OASIS	<pre> PS C:\Users\Stefanus\Desktop\tucil> python tucil.py masukkan nama file : test3 COCA COLA -----+ OASIS 8186 8106 -----+ 16292 banyak tes untuk menemukan substitusi angka yang benar adalah 310988 waktu yang dibutuhkan : --- 2.842214345932007 seconds --- </pre>
A A+ ----- B	<pre> PS C:\Users\Stefanus\Desktop\tucil> python tucil.py masukkan nama file : test6 A A -----+ B 4 4 -----+ 8 banyak tes untuk menemukan substitusi angka yang benar adalah 52 waktu yang dibutuhkan : --- 0.012000083923339844 seconds --- </pre>

NO GUN NO+ ----- HUNT	PS C:\Users\Stefanus\Desktop\tucil> python tucil.py masukkan nama file : test4 NO GUN NO _____ + HUNT 87 908 87 _____ + 1082 banyak tes untuk menemukan substitusi angka yang benar adalah 81280 waktu yang dibutuhkan : --- 0.7750599384307861 seconds ---
MEMO FROM+ ----- HOMER	PS C:\Users\Stefanus\Desktop\tucil> python tucil.py masukkan nama file : test7 MEMO FROM _____ + HOMER 8485 7358 _____ + 15843 banyak tes untuk menemukan substitusi angka yang benar adalah 528147 waktu yang dibutuhkan : --- 4.810358285903931 seconds ---
TWO TWO+ ----- FOUR	PS C:\Users\Stefanus\Desktop\tucil> python tucil.py masukkan nama file : test10 TWO TWO _____ + FOUR 938 938 _____ + 1876 banyak tes untuk menemukan substitusi angka yang benar adalah 813027 waktu yang dibutuhkan : --- 8.454628944396973 seconds ---
UKT ITB+ ----- FULL	PS C:\Users\Stefanus\Desktop\tucil> python tucil.py masukkan nama file : test11 UKT ITB _____ + FULL 576 968 _____ + 1544 banyak tes untuk menemukan substitusi angka yang benar adalah 1802535 waktu yang dibutuhkan : --- 16.55252456665039 seconds ---

BAB 5

Cek Poin

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

Link github : https://github.com/stefanus-lamlo/Tucil1_IF2211