

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



Disusun Oleh:

Zaidan Naufal Sudrajat / 13518021

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

A. Algoritma Brute Force

1. Persiapan Data

Sebelum pencarian angka dilakukan akan disiapkan 2 tipe data yang diambil dari Input. Data tersebut yaitu :

- List of String Operand dan Hasil Penjumlahan
List ini berisi dua atau lebih string operand , dan sebuah string hasil penjumlahan yang ditempatkan pada end of list.

```
['THREE', 'THREE', 'TWO', 'TWO', 'ONE', 'ELEVEN']
```

- Dictionary Buffer
Dictionary berisi semua huruf berbeda yang terdapat pada soal Cryptarithmic , dan untuk buffer dipasangkan dengan char '0'

```
{'W': '0', 'R': '0', 'T': '0', 'E': '0', 'O': '0', 'L': '0', 'V': '0', 'N': '0', 'H': '0'}
```

2. Melakukan Pengulangan berdasarkan Permutasi

Pada setiap loop akan dilakukan:

- I. Program akan me-*generate* sebuah list angka sepanjang 10 (0-9) angka , list yang akan digunakan merupakan list sepanjang jumlah string berbeda pada soal dimulai dari belakang list angka, contoh apabila terdapat 8 huruf berbeda pada list soal, maka list angka yang digunakan dari list posisi 2 hingga 9 (pada loop selanjutnya akan digenerate permutasi dari string loop sebelumnya, atau string dengan angka berbeda).

```
[0, 1, 2, 3, 4, 5, 6, 7, 8]
```

- II. Mengganti dictionary dengan angka pada list string yang di-*generate*

```
{'L': '0', 'R': '1', 'V': '2', 'W': '3', 'H': '4', 'N': '5', 'O': '6', 'E': '7', 'T': '8'}
```

- III. Membuat list baru yang berisi hasil *mapping* list of string sesuai dictionary
Setiap string pada list operand akan di mapping sesuai dengan dictionary dan hasilnya dimasukkan ke list baru (pada loop selanjutnya list baru tersebut dikosongkan dan dimasukkan string angka baru)

```
['83722', '83722', '865', '865', '542', '202124']
```

- IV. Pengecekan char pertama dari string
Pengecekan dilakukan dengan mengecek semua string pada list , apabila char pertama pada string anggota list merupakan 0 maka loop dihentikan dan pencarian dilakukan pada loop berikutnya dengan angka berbeda.
- V. Pengecekan Penjumlahan
Setiap string operand (string yang berposisi pada list 0 hingga list sebelum terakhir) diubah menjadi integer lalu dijumlahkan , apabila hasil penjumlahan semua operand sama dengan integer “hasil penjumlahan”

(string yang berposisi pada list terakhir). Apabila sama maka loop selesai dan list dikeluarkan pada layar dan program selesai, sedangkan apabila berbeda akan dilanjutkan loop selanjutnya dengan angka berbeda.

B. Source Code dalam Bahasa Python

```
import timeit

def listtoint(l):
    i = len(l)-1
    integer = 0
    for x in l:
        integer = integer + x * 10**i
        i-=1
    return integer

def permutations(s,len):
    for i in range(len-2,-1,-1):
        if ( s[i] < s[i+1] ):
            break;
    if ( i < 0 ):
        return 0
    for j in range(len-1,i-1,-1):
        if ( s[j] > s[i] ):
            temp = s[i]
            s[i] = s[j]
            s[j] = temp
            break
    i = i + 1
    k = len-1
    while(k>i):
        temp = s[i]
        s[i] = s[k]
        s[k] = temp
        i+=1
        k-=1
    return s;

def listtostring(l):
    string = ""
    for x in l:
        string = string + x
    return string

def makedict(s):
    d = {}
    for i in range(len(s)):
        d[s[i]] = '0'
    return d

def changedictvalue(d,l):
    i = 0
    for key in d.keys():
        d[key] = str(l[i])
        i += 1
    return d

def changestringvalue(s,dict):
    for word, initial in dict.items():
        s = s.replace(word, initial)
```

```

    return s

def check(l):
    for i in l:
        if i[0] == '0':
            return False
    total = 0
    for i in l[::-1]:
        total = total + int(i)
    return total == int(l[-1])

def solution(dict, listsum):
    p = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
    tries = 1
    while int(listtoint(p)) != 9876543210:
        listsum_new = []
        dict = changedictvalue(dict, p[10-len(dict):])
        for j in range(len(listsum)):
            listsum_new.append(changestringvalue(listsum[j], dict))
        if check(listsum_new):
            listsum_new.append(str(tries))
            return listsum_new
        else:
            tries += 1
        p = permutations(p, 10)
    else:
        print("tidak ditemukan solusi")
        return []

def crypt(tekateki):
    print("Soal :")
    print(tekateki)
    print("\nSolusi :")
    tekateki = tekateki.replace(" ", "").replace('+', "").split('\n')
    tekateki.pop(-2)
    word = ".join(set(listtostring(tekateki)))
    if len(word) <= 10:
        worddict = makedict(word)
        listsolution = solution(worddict, tekateki)
        if listsolution:
            tries = listsolution[-1]
            listsolution = listsolution[:-1]
            whitespace = ''
            for i in listsolution[:-2]:
                wlenght = len(listsolution[-1]) - len(i)
                print(whitespace*wlenght+i)
            print(whitespace*(len(listsolution[-1]) - len(listsolution[-2])) + listsolution[-2] + '+')
            print('-----')
            print(listsolution[-1])
            print(f"\nTotal percobaan: {tries}")
        else:
            print("Tidak terdapat solusi dikarenakan jumlah huruf berbeda lebih dari 10")

if __name__ == "__main__":
    namafile = "tekateki.txt"
    f = open(f"./test/{namafile}", "r")
    tekateki = f.read()
    tekateki = tekateki.split('\n\n')
    for i in range(len(tekateki)):
        start=timeit.default_timer()

```

```
crypt(tekateki[i])
stop=timeit.default_timer()
print("Waktu diperlukan: " + "{:.2f}".format(stop-start) + " seconds\n")
```

C. Hasil *Screenshot* input dan output

1. SEND + MORE = MONEY

```
Soal :
SEND
MORE+
-----
MONEY

Solusi :
9567
1085+
-----
10652

Total percobaan: 1386418
Waktu diperlukan: 7.66 seconds
```

2. FORTY + TEN + TEN = SIXTY

```
Soal :
FORTY
TEN
TEN+
-----
SIXTY

Solusi :
29786
850
850+
-----
31486

Total percobaan: 1862973
Waktu diperlukan: 15.11 seconds
```

3. NUMBER + NUMBER = PUZZLE

```
Soal :  
NUMBER  
NUMBER+  
-----  
PUZZLE  
  
Solusi :  
201689  
201689+  
-----  
403378  
  
Total percobaan: 3165025  
waktu diperlukan: 24.52 seconds
```

4. TILES + PUZZLES = PICTURE

```
Soal :  
  TILES  
PUZZLES+  
-----  
PICTURE  
  
Solusi :  
  91542  
3077542+  
-----  
3169084  
  
Total percobaan: 1514613  
waktu diperlukan: 11.94 seconds
```

5. CLOCK + TICK + TOCK = PLANET

```

Soal :
CLOCK
TICK
TOCK+
-----
PLANET

Solusi :
90892
6592
6892+
-----
104376

Total percobaan: 1708218
Waktu diperlukan: 18.20 seconds

```

6. COCA + COLA = OASIS

```

Soal :
COCA
COLA+
-----
OASIS

Solusi :
8186
8106+
-----
16292

Total percobaan: 105631
Waktu diperlukan: 0.56 seconds

```

7. HERE + SHE = COMES

```
Soal :  
  HERE  
  SHE+  
-----  
COMES  
  
Solusi :  
  9454  
  894+  
-----  
10348  
  
Total percobaan: 251416  
Waktu diperlukan: 1.50 seconds
```

8. THREE + THREE + TWO + TWO + ONE = ELEVEN


```

Soal :
THREE
THREE
TWO
TWO
ONE+
-----
ELEVEN

Solusi :
84611
84611
803
803
391+
-----
171219

Total percobaan: 631243
waktu diperlukan: 6.18 seconds

```

D. Link Source Code

Github : https://github.com/zaidannaufal/Tucil_Stima/tree/main/tucil_1/Tucil1_13518021

E. Kotak Penilaian Asisten

| 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 cryptarithmic hanya benar untuk persoalan cryptarithmic dengan dua buah operand. | | ✓ |
| 5. Solusi cryptarithmic benar untuk persoalan cryptarithmic untuk lebih dari dua buah operand. | ✓ | |