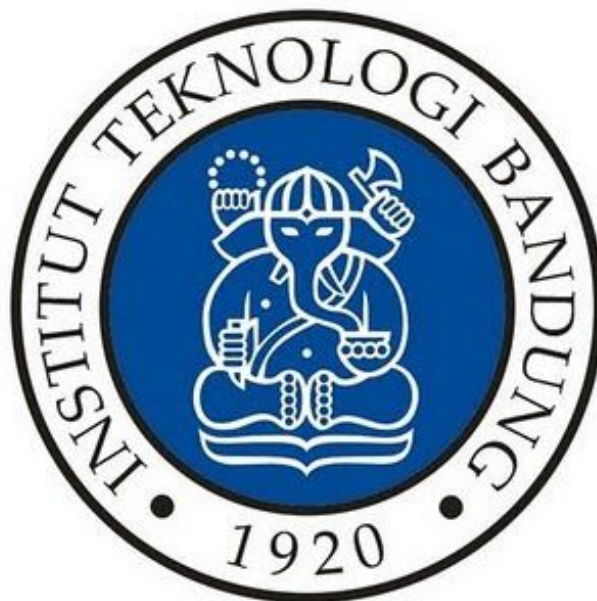


Laporan Tugas Kecil 1 IF2211 Strategi Algoritma

Penyelesaian *Cryptarithmic* dengan Algoritma *Brute Force*

Gayuh Tri Rahutami
13519192



**PROGRAM STUDI TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
2021**

I. Algoritma *Brute Force*

Langkah-langkah yang dilakukan untuk mendapatkan jawaban dari *cryptarithmic* yang dimasukkan oleh user adalah:

1. Menyimpan semua huruf yang terdapat di dalam kata-kata yang dimasukkan oleh user ke dalam sebuah *list*. Apabila jumlah huruf kurang dari 10, *string* kosong dimasukkan ke dalam *list* tersebut sehingga panjang *list* adalah 10. Indeks dari huruf pada list kemudian akan menjadi hasil translasi huruf ke angka.
2. Enumerasi seluruh kemungkinan solusi dengan mencari permutasi dari *list* yang didapatkan dari langkah 1.
3. Evaluasi tiap kemungkinan dengan mengkonversi kata menjadi bilangan dan mengecek apakah hasil penjumlahan *operand*-nya sama dengan hasil penjumlahan dari masukan. Jika sama, maka pencarian dihentikan. Jika berbeda, pencarian dilanjutkan.
4. Setelah solusi ditemukan, solusi dicetak ke layar *user*.

II. *Source Code*

```
# Tugas Kecil 1 Strategi Algoritma
# Penyelesaian Cryptarithmic dengan Algoritma Brute Force
# Nama: Gayuh Tri Rahutami
# NIM: 13519192

import time

# Fungsi untuk permutasi
def permutasi(li):
    listPermutasi = []

    if(len(li) < 2):
        return li
    else:
        for i in range (len(li)):
            temp = permutasi(li[0:i] + li[i+1:len(li)])

            if(len(li) > 2):
                for member in temp:
                    member.append(li[i])
                    listPermutasi.append(member)
            else:
                temp.append(li[i])
                listPermutasi.append(temp)

        return listPermutasi

# Fungsi untuk meng-convert string ke integer
def convert(str, listHuruf):
    converted = 0
    for huruf in str:
        converted *= 10
        converted += listHuruf.index(huruf)
    return converted

# Fungsi untuk mengecek apakah permutasi sudah benar
def check(listKata, listHuruf, result):
    sum = 0
```

```

convKata = []
i = 0
errorFound = False

while(not errorFound and i < len(listKata)):
    convKata.append(0)
    if (listHuruf.index(listKata[i][0]) == 0): # Apabila ditemukan
sebuah kata yang huruf pertamanya = 0 maka permutasi salah
        errorFound = True
    else:
        #meng-convert kata menjadi integer dan menjumlahkannya
        convKata[i] = convert(listKata[i], listHuruf)
        sum += convKata[i]
    i += 1

    if(sum == result and not errorFound): #Apabila jumlah operand-operand
= hasil, maka list berisi kata yang telah diconvert dikembalikan
        return convKata
    else: #Jika jumlah operand-operand != hasil, maka list kosong
dikembalikan
        return []

print("=====")
filename = input("Masukkan nama file:\n")
print("=====")
start_time = time.time()
file = open(filename, 'r')

listKata = [] # List untuk menyimpan kata-kata operand
listHuruf = [] # list untuk menyimpan huruf-huruf yang ada di operasi

# Pengambilan operand pertama
listKata.append(file.readline())
i = 0

# Membersihkan spasi dan newline
listKata[i] = listKata[i].replace(" ", "")
listKata[i] = listKata[i].replace("\n", "")

# Memasukkan huruf-huruf ke listHuruf
for huruf in listKata[i]:
    if not(huruf in listHuruf) and (huruf != "+") and (huruf != " ") and
(huruf != "\n"): # Jika huruf belum ada sebelumnya baru dimasukkan ke
list
        listHuruf.append(huruf)

# Membaca operand-operand selanjutnya
# Pembacaan berhenti apabila ditemukan baris yang mengandung tanda "+"
while not('+' in listKata[i]):
    i += 1

    # Pengambilan input ke-i
    listKata.append(file.readline())

    # Membersihkan spasi dan newline
    listKata[i] = listKata[i].replace(" ", "")
    listKata[i] = listKata[i].replace("\n", "")

    # Memasukkan huruf-huruf ke listHuruf

```

```

for huruf in listKata[i]:
    # Jika huruf belum ada sebelumnya baru dimasukkan ke list
    # Jika huruf = "+" tidak perlu dimasukkan
    if not(huruf in listHuruf) and (huruf != "+"):
        listHuruf.append(huruf)

# Menghilangkan tanda + dari operand terakhir
listKata[i] = listKata[i].replace("+", "")

file.readline() #Garis tidak perlu disimpan

result = file.readline() # Menyimpan hasil penjumlahan

#Membersihkan spasi dan newline pada string hasil
result = result.replace("\n", "")
result = result.replace(" ", "")

file.close()
i = 0

# Mencetak input
print("Input:\n")
for kata in listKata:
    if (i == len(listKata) - 1): #Jika kata bukan elemen terakhir pada
list
        space = len(result) - len(kata)
        print("+", end="")
    else:
        space = len(result) - len(kata) + 1
        print(space*" " + kata)
    i += 1

print((len(result)+1) * "-")
print(" " + str(result))

print("=====")

# Memasukkan huruf-huruf di string hasil ke listHuruf
for huruf in result:
    # Jika huruf belum ada sebelumnya baru dimasukkan ke list
    if not(huruf in listHuruf):
        listHuruf.append(huruf)

# Apabila total huruf < 10, maka tambahkan string kosong hingga panjang
list = 10
while(len(listHuruf) < 10):
    listHuruf.append("")

# Mencari Permutasi List
permutationList = permutasi(listHuruf)

found = False
i = 0
convKata = []

# Mengevaluasi setiap permutasi
while(convKata == []) and (i < len(permutationList)):
    convResult = convert(result, permutationList[i])
    convKata = check(listKata, permutationList[i], convResult)

```

```

        i += 1

# Mencetak hasil
if(convKata == []):
    print("Persoalan ini tidak memiliki penyelesaian.")
else:
    print("Result:\n")
    for kata in convKata:
        if(convKata.index(kata) == len(convKata) - 1):
            space = len(result) - len(str(kata))
            print("+", end="")
        else:
            space = len(result) - len(str(kata)) + 1
            print(space*" " + str(kata))

    print((len(result)+1) * "-")
    print(" " + str(convResult))

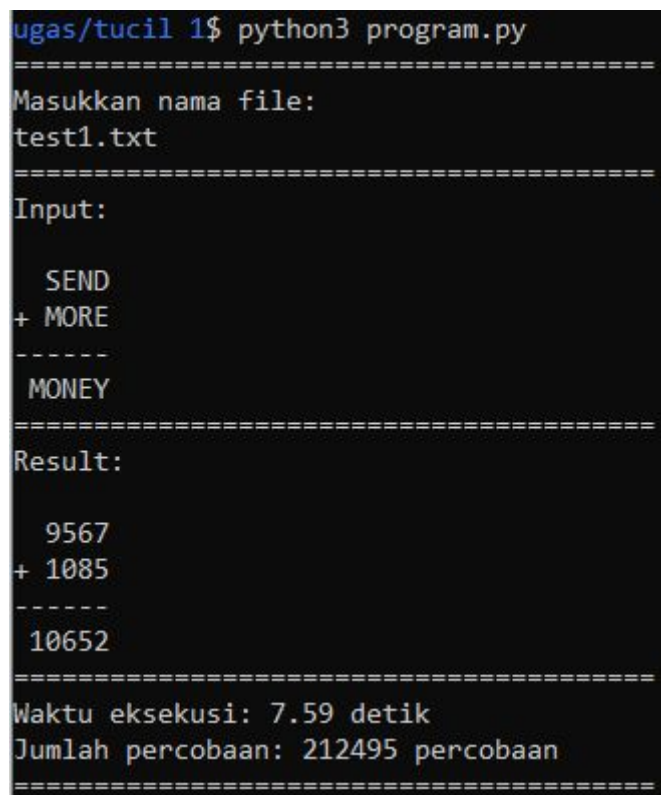
# Mencetak waktu eksekusi dan jumlah percobaan
print("=====")
print("Waktu eksekusi: %.2f detik" %(time.time()-start_time))
print("Jumlah percobaan: %d percobaan" %(i))
print("=====")

input()

```

III. *Screenshot* hasil program

1. Percobaan 1



```

lugas/tucil 1$ python3 program.py
=====
Masukkan nama file:
test1.txt
=====
Input:

    SEND
+ MORE
-----
    MONEY
=====
Result:

    9567
+ 1085
-----
    10652
=====
Waktu eksekusi: 7.59 detik
Jumlah percobaan: 212495 percobaan
=====

```

2. Percobaan 2

```
ugas/tucil 1$ python3 program.py
=====
Masukkan nama file:
test2.txt
=====
Input:

  NUMBER
+NUMBER
-----
  PUZZLE
=====
Result:

  201689
  201689
-----
  403378
=====
Waktu eksekusi: 13.60 detik
Jumlah percobaan: 2008772 percobaan
=====
```

3. Percobaan 3

```
ugas/tucil 1$ python3 program.py
=====
Masukkan nama file:
test3.txt
=====
Input:

  TILES
+PUZZLES
-----
  PICTURE
=====
Result:

  91542
+3077542
-----
  3169084
=====
Waktu eksekusi: 8.44 detik
Jumlah percobaan: 357279 percobaan
=====
```

4. Percobaan 4

```
agus/tucil 1$ python3 program.py
=====
Masukkan nama file:
test4.txt
=====
Input:
    CLOCK
    TICK
+   TOCK
-----
    PLANET
=====
Result:
    90892
    6592
+   6892
-----
    104376
=====
Waktu eksekusi: 7.57 detik
Jumlah percobaan: 77374 percobaan
=====
```

5. Percobaan 5

```
agus/tucil 1$ python3 program.py
=====
Masukkan nama file:
test5.txt
=====
Input:
    COCA
+   COLA
-----
    OASIS
=====
Result:
    8186
+   8106
-----
    16292
=====
Waktu eksekusi: 12.20 detik
Jumlah percobaan: 1835735 percobaan
=====
```

6. Percobaan 6

```
ugus/tucil 1$ python3 program.py
=====
Masukkan nama file:
test6.txt
=====
Input:
    DOUBLE
    DOUBLE
+   TOIL
-----
    TROUBLE
=====
Result:
    798064
    798064
+   1936
-----
    1598064
=====
Waktu eksekusi: 9.45 detik
Jumlah percobaan: 404442 percobaan
=====
```

7. Percobaan 7

```
=====
Masukkan nama file:
test7.txt
=====
Input:
    HERE
+   SHE
-----
    COMES
=====
Result:
    9454
+   894
-----
    10348
=====
Waktu eksekusi: 7.55 detik
Jumlah percobaan: 109577 percobaan
=====
```


8. Percobaan 8

```
=====
Masukkan nama file:
test8.txt
=====
Input:

  THREE
  THREE
    TWO
    TWO
+   ONE
-----
  ELEVEN
=====
Result:

  29700
  29700
    214
    214
+   480
-----
  60308
=====
Waktu eksekusi: 10.19 detik
Jumlah percobaan: 573092 percobaan
=====
```

IV. Alamat Drive untuk Kode Program

<https://github.com/wundersmith/tucil1-Cryptarithmic>

V. Cek 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 <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> dengan lebih dari dua buah <i>operand</i>	✓	