

Laporan Tugas Kecil
IF2211 Strategi Algoritma
Cryptarithmic



Rexy Gamaliel Rumahorbo (13519010)

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
2019

1. Algoritma Brute Force

a. Preparasi:

- Program membaca file
- Menghitung waktu awal
- Membuat list kata (word) "wordList" yang terdapat pada persamaan, dan dictionary "varDict" yang berisi pasangan variabel–nilai, serta menginisiasi list "solutions".
- Menampilkan persamaan pada soal.

b. Iterasi (brute force):

- Mengecek apakah konfigurasi nilai variabel-variabel pada varDict memenuhi, yakni tidak ada huruf inisial yang bernilai 0 dan masing-masing variabel bernilai unik. Jika ya, lanjut ke poin selanjutnya. Jika tidak, lanjut ke poin terakhir.
- Mengecek apakah untuk konfigurasi nilai variabel-variabel yang valid tersebut memenuhi persamaan cryptarithm-nya. Jika ya, menampilkan bahwa program menemukan sebuah solusi sementara, dan menambahkannya ke list solutions. Jika tidak, lanjut ke poin terakhir.
- Melakukan *increment* pada konfigurasi nilai variabel-variabel pada varDict, lanjut ke iterasi berikutnya. Iterasi dilakukan sebanyak 10^n kali, di mana n adalah jumlah variabel pada persamaan pada soal.

c. Solusi:

- Mengiterasi seluruh elemen list solutions dan menampilkan hasilnya
- Menampilkan waktu berjalannya program dan banyak tes yang dilakukan

2. Link: <https://github.com/Rexy-Gamaliel/Cryptarithmic>

3. Penilaian

Poin	Ya	Tidak
1. Program berhasil dikompilasi tanpa kesalahan (no syntax error)	v	
2. Program berhasil running	v	
3. Program dapat membaca file masukan dan menuliskan keluaran	v	
4. Solusi <i>Cryptarithmic</i> hanya benar untuk persoalan <i>cryptarithmic</i> dengan dua buah operan		v
5. Solusi <i>Cryptarithmic</i> benar untuk persoalan <i>cryptarithmic</i> untuk lebih dari dua buah operan	v	

Screenshoot Result

1) HERE + SHE = COMES

The screenshot shows a terminal window on the left and a Notepad window on the right. The terminal displays the execution of a program that solves the cryptarithm $HERE + SHE = COMES$. It shows the input, the program's output, and the solution found: $HERE = 8454$ and $SHE = 804$, which sum to $COMES = 18548$. The Notepad window shows the input text: $HERE + SHE = COMES$.

```
8454
8454
-----
18548

End time: Tue Jan 26 21:22:10 2021
Elapsed time: 0.12121221271 seconds
Total test run: 1000000 (10^6)

D:\Documents\College_Sophomore\Coolyash_Second\Strategi_Algoritma\tucilipy cryptarithmic.py
Start time: Tue Jan 26 21:24:48 2021
Persamaan cryptarithm yang akan dipecahkan:
  H E R E
    S H E
  -----
  C O M E S

=====Solution=====
Solution 1
  H E R E
    S H E
  -----
  C O M E S

End time: Tue Jan 26 21:25:48 2021
Elapsed time: 0.3645873969763 seconds
Total test run: 1000000 (10^6)

D:\Documents\College_Sophomore\Coolyash_Second\Strategi_Algoritma\tucilip>
```

input.txt - Notepad

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

2) MEMO + FROM = HOMER

The screenshot shows a terminal window on the left and a Notepad window on the right. The terminal displays the execution of a program that solves the cryptarithm $MEMO + FROM = HOMER$. It shows the input, the program's output, and the solution found: $MEMO = 8485$ and $FROM = 7329$, which sum to $HOMER = 15814$. The Notepad window shows the input text: $MEMO + FROM = HOMER$.

```
8485
7329
-----
15814

End time: Tue Jan 26 21:25:48 2021
Elapsed time: 5.37615079140748 seconds
Total test run: 1000000 (10^6)

D:\Documents\College_Sophomore\Coolyash_Second\Strategi_Algoritma\tucilip>
```

input.txt - Notepad

```
File Edit Format View Help
MEMO
FROM
-----
HOMER
```

3) NO + GUN + NO = HUNT

The screenshot shows a terminal window on the left and a Notepad window on the right. The terminal displays the execution of a program that solves the cryptarithm $NO + GUN + NO = HUNT$. It shows the input, the program's output, and the solution found: $NO = 80$, $GUN = 804$, and $NO = 80$, which sum to $HUNT = 1684$. The Notepad window shows the input text: $NO + GUN + NO = HUNT$.

```
80
804
80
-----
1684

End time: Tue Jan 26 21:35:16 2021
Elapsed time: 5.268327713052695 seconds
Total test run: 1000000 (10^6)

D:\Documents\College_Sophomore\Coolyash_Second\Strategi_Algoritma\tucilip>
```

input.txt - Notepad

```
File Edit Format View Help
NO
GUN
NO
-----
HUNT
```

4) OCRYPT + OCRYPT + OCRYPT = CRYPTO

```

D:\Documents\College_Sophomore\Conlyeah_Second\Strategi_Algoritma\tuillip cryptarithm.py
Start time: Tue Jan 26 21:41:47 2021
Permainan cryptarithm yang akan dipelajari:
O C R Y P T
O C R Y P T
O C R Y P T
-----
C R Y P T O

=====Solution=====
Solution 1
 2 8 5 7 1 4
 2 8 5 7 1 4
 2 8 5 7 1 4
-----
8 5 7 1 4 2

Solution 2
 1 4 2 8 5 7
 1 4 2 8 5 7
 1 4 2 8 5 7
-----
4 2 8 5 7 1

End time: Tue Jan 26 21:41:53 2021
Elapsed time: 8.578648909110285 seconds
Total test run: 1000000 (10^6)
D:\Documents\College_Sophomore\Conlyeah_Second\Strategi_Algoritma\tuillip:
  
```

5) REXY + IFITB + BERRY = RIBET

```

D:\Documents\College_Sophomore\Conlyeah_Second\Strategi_Algoritma\tuillip cryptarithm.py
Start time: Tue Jan 26 21:39:27 2021
Permainan cryptarithm yang akan dipelajari:
R X Y
I F I T B
B E R R Y
-----
R I B E T

=====Solution=====
Solution 1
 9 3 1 7
 5 1 0 0
 5 1 0 0
-----
0 3 1 7

End time: Tue Jan 26 21:39:46 2021
Elapsed time: 119.17751341796 seconds
Total test run: 1000000000 (10^9)
D:\Documents\College_Sophomore\Conlyeah_Second\Strategi_Algoritma\tuillip:
  
```

6) ALIAS + ASNI = AINUN

```

D:\Documents\College_Sophomore\Conlyeah_Second\Strategi_Algoritma\tuillip cryptarithm.py
Start time: Tue Jan 26 22:32:42 2021
Permainan cryptarithm yang akan dipelajari:
A L I A S
A S N I
-----
A I N U N

=====Solution=====
Solution 1
 1 0 7 1 3
 3 1 0 0
-----
1 7 0 1 0

Solution 2
 2 0 3 1 1
 3 1 0 0
-----
2 0 0 1 0

Solution 3
 3 1 4 2 0
 3 1 0 0
-----
3 2 0 1 0

Solution 4
 3 0 0 1 1
 3 1 0 0
-----
3 0 0 1 0

Solution 5
 3 1 0 1 1
 3 1 0 0
-----
2 0 1 1 1

End time: Tue Jan 26 22:32:44 2021
Elapsed time: 2.000000 seconds
Total test run: 2000000 (2*10^6)
D:\Documents\College_Sophomore\Conlyeah_Second\Strategi_Algoritma\tuillip:
  
```

```

C:\Users\user>python3 main.py
Solution 49
  2 4 0 2 1
  1 1 7 0
-----+
  2 4 2 0 2

Solution 50
  2 1 1 2 4
  1 4 7 3
-----+
  2 1 7 0 7

Solution 51
  1 1 0 1 2
  1 3 0 0
-----+
  1 4 0 0 4

Solution 52
  1 4 0 1 1
  1 3 0 0
-----+
  1 0 0 0 0

Solution 53
  1 1 1 1 1
  1 1 0 0
-----+
  1 1 0 0 0

End Time: Tue Jan 26 22:18:14 JKT
Elapsed time: 7.255821666666667 seconds
Total test run: 1000000 (10^6)

D:\Documents\College_Sophomore\College_Secund\Strategi_Algoritma\Facilitas

```

7) COCA + COLA = OASIS

```

D:\Documents\College_Sophomore\College_Secund\Strategi_Algoritma\Facilitas>python3 cryptarithmic.py
Start time: Tue Jan 26 23:13:33 JKT
Pernyataan cryptarithma yang akan dipecahkan:
  C O C A
+ C O L A
-----+
  O A S I S

=====Solution=====
Solution 1
  8 1 0 8
+ 8 1 0 8
-----+
  1 6 2 0 2

End Time: Tue Jan 26 23:13:38 JKT
Elapsed time: 8.221232222222222 seconds
Total test run: 1000000 (10^6)

D:\Documents\College_Sophomore\College_Secund\Strategi_Algoritma\Facilitas

```

8) EAT + THAT = APPLE

```

D:\Documents\College_Sophomore\College_Secund\Strategi_Algoritma\Facilitas>python3 cryptarithmic.py
Start time: Tue Jan 26 21:43:07 JKT
Pernyataan cryptarithma yang akan dipecahkan:
  E A T
+ T H A T
-----+
  A P P L E

=====Solution=====
Solution 1
  0 1 0
+ 0 2 1 0
-----+
  1 0 0 0 0

End Time: Tue Jan 26 21:43:07 JKT
Elapsed time: 0.447151184002071 seconds
Total test run: 1000000 (10^6)

D:\Documents\College_Sophomore\College_Secund\Strategi_Algoritma\Facilitas

```

4. Source Code

```
import time

def InitializeVarDict(wordList):
    # Menginisialisasi dictionary yang berisi pasangan seluruh variabel pada equati
on dan nilainya
    temp = {}
    for word in wordList:
        for char in word:
            temp.update({char: 0})
    return temp

def InterpretWord (word, varDict):
    # Menentukan nilai dari suatu Word berdasarkan kombinasi nilai variabel pada va
rDict
    # word      : string
    # varDict   : dictionary dengan pasangan <Var, Value>
    value = 0
    temp = varDict.copy()
    for currentChar in word:
        value *= 10
        value += temp.get(currentChar)
    return value

def IncrementVarConfiguration(varDict):
    # Meng-increment konfigurasi nilai variabel-variabel pada varDict
    tempVarDict = varDict.copy()
    for Var in tempVarDict:
        currentVal = tempVarDict.get(Var)
        if (currentVal == 9):
            tempVarDict.update({Var: 0})
        else:
            currentVal += 1
            tempVarDict.update({Var: currentVal})
            break
    return tempVarDict

def IsEquationValid(wordList, varDict):
    # Menentukan apakah suatu konfigurasi nilai variabel-variabel memenuhi equation
    tempVarDict = varDict.copy()
    tempSum = 0
    curWordVal = 0    # akan berguna saat menyimpan nilai dari word terakhir (hasil
penjumlahan dalam persamaan)
    for word in wordList:
```

```

    curWordVal = InterpretWord(word, tempVarDict)
    tempSum += curWordVal
    # curWordVal bernilai value dari word terakhir, tempSum bernilai hasil penjumlahan semua word dalam persamaan
    return (tempSum == 2*curWordVal)

def HasZeroAsInitial(wordInitials, varDict):
    # Menentukan apakah huruf inisial sebuah kata yang bernilai 0
    tempVarDict = varDict.copy()
    for Var in tempVarDict:
        if (tempVarDict.get(Var) == 0):
            if (Var in wordInitials):
                return True
    return False

def IsVarUnique(varDict):
    # Menentukan apakah nilai semua variabel unik
    tempVarDict = varDict.copy()
    unique = True
    tempVallist = []
    for Var in tempVarDict:
        tempVallist.append(tempVarDict.get(Var))
    # cek jika ada kemunculan lebih dari sekali
    return not(any([tempVallist.count(element) > 1 for element in tempVallist]))

def PrintEquation(wordList, varDict, interpret):
    # Mencetak persamaan dalam format penjumlahan ke bawah
    # Contoh:
    #   S E N D
    #   M O R E
    # ----- +
    #   M O N E Y

    numWord = len(wordList)
    i = numWord + 1    # banyak baris
    maxWordLength = max(len(word) for word in wordList)
    j = 2*maxWordLength + 3    # banyak kolom

    nthWord = 0
    for row in range(i):
        if (row == numWord-1): # baris garis
            for col in range(2*maxWordLength+1):
                print("-", end='')
            print(" ", end='')
            print("+", end='')

```

```

else:
    # baris berisi word
    curWord = wordList[nthWord]
    curWordLength = len(curWord)
    nthChar = 0
    for col in range(maxWordLength):
        print(" ", end='')
        if (col >= maxWordLength - curWordLength):
            if (interpret):
                print(varDict.get(curWord[nthChar]), end = '')
            else:
                print(curWord[nthChar], end = '')
            nthChar += 1
        else:
            print(" ", end = '')
    nthWord += 1
    print()

def UpdateWordDict(wordDict, varDict):
    # Memperbarui value dari setiap word dalam wordDict
    tempVarDict = varDict.copy()
    tempWordDict = wordDict.copy()
    for word in tempWordDict:
        newVal = InterpretWord(word, tempVarDict)
        tempWordDict.update({word: newVal})
    return tempWordDict

# Kamus
# wordList    List "kata-kata" apa saja yang ada dalam persamaan
# wordDict    Dictionary berisi pasangan key-
value <Word, Value>, Value bergantung pada interpretasi setiap karakter pada varD
ict
# varDict     Dictionary berisi semua variabel pada persamaan dan kombinasi nilai
nya
# solutions   List of dictionary kombinasi pasangan <Var, Value> yang memenuhi pe
rseamaan

# Enumerasi "kata" dan inisial kata yang ada pada persamaan
wordList = []
wordInitials = []

# Pembacaan file
with open("input.txt") as file:
    for row in file:
        if (row[0] != '-'):

```



```

        if (row[-1] == '\n'):
            wordList.append(row[:-1])
        else:
            wordList.append(row)
file.close()

# Waktu Awal
start = time.time()
print("Start time: ", time.ctime(start))

# Membuat list huruf inisial
wordInitials = [word[0] for word in wordList if word[0] not in wordInitials]

# Inisialisasi Dictionary <Word, Value>
wordDict = {}
wordDict = wordDict.fromkeys(wordList, 0)

# Inisialisasi Dictionary <Var, Val>
varDict = InitializeVarDict(wordList)

# Persamaan
print("Persamaan cryptarithm yang akan dipecahkan:")
PrintEquation(wordList, varDict, False)
print()

# Iterasi untuk setiap kombinasi nilai variabel yang mungkin (Brute Force part)
# Karena menggunakan Brute Force, jumlah kemungkinan yang dicoba itu fixed

solutions = []
varLength = len(varDict.keys()) # menentukan banyaknya iterasi yang harus dilakukan

for _ in range(10**varLength):
    if (not(HasZeroAsInitial(wordInitials, varDict)) and IsVarUnique(varDict)):
        if (IsEquationValid(wordList, varDict)):
            solutions.append(varDict)
        varDict = IncrementVarConfiguration(varDict)

print("#####Solutions#####")
i = 1
#print(solutions)
for item in solutions:
    #print("item", item)
    print("Solution", i)

```

```
PrintEquation(wordList, item, True)
print()
i += 1
print("- - - - -")
if not any(solutions):
    print("No solutions")

# Waktu Akhir
end = time.time()
print("End time: ", time.ctime(end))
print("Ellapsed time: ", (end-start), "seconds")

print("Total test run:", 10**varLength, "(10^" + str(varLength) + ")" )
```