

LAPORAN TUGAS KECIL 1 MATA KULIAH IF2211 STRATEGI ALGORITMA

MAKALAH

Disusun untuk Memenuhi Tugas Mata Kuliah IF2211 Strategi Algoritma

Nama

Prana Gusriana

NIM

(13519195)



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

DAFTAR ISI

Contents

DAFTAR ISI	2
BAB I	3
I. Abstraksi	3
BAB II	5
BAB III	7
BAB IV	19
BAB V	25

BAB I

DESKRIPSI MASALAH

I. Abstraksi

Cryptarithmic (atau *cryptarithm*) adalah sebuah *puzzle* penjumlahan di dalam matematika dimana angka diganti dengan huruf. Setiap angka dipresentasikan 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, M = 1, O = 0, R = 8, Y = 2

Contoh-contoh *cryptarithmic* dengan solusinya:

$$\begin{array}{r} \text{JUNE} \quad 7924 \\ + \text{JULY} \quad + 7906 \\ \hline \text{APRIL} \quad 15830 \end{array}$$

$$\begin{array}{r} \text{F O R T Y} \\ \quad \text{T E N} \\ + \quad \text{T E N} \\ \hline \text{S I X T Y} \end{array} \qquad \begin{array}{r} 2 \ 9 \ 7 \ 8 \ 6 \\ \quad \quad 8 \ 5 \ 0 \\ \quad \quad 8 \ 5 \ 0 \\ \hline 3 \ 1 \ 4 \ 8 \ 6 \end{array}$$

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

Tugas anda adalah merancang algoritma *brute force* untuk menyelesaikan persoalan *cryptarithmic*.

Spesifikasi Tugas Kecil 1:

- Tuliskan program kecil (sederhana) dalam Bahasa C/C++/Java/Python yang mengimplementasikan algoritma *Brute Force* untuk mencari solusi persoalan

cryptarithmic.

- Operasi aritmetika yang dipakai hanyalah tambah (+)
- Jumlah *operand* dalam operasi tambah minimal 2 buah. Misalnya SEND + MORE = MONEY, maka *operand* 1 adalah SEND dan *operand* 2 adalah MONEY
- Jumlah huruf di dalam *operand* paling banyak 10 buah
- Setiap huruf menyatakan angka yang unik
- Huruf pertama tidak boleh menyatakan angka 0
- **Input:** file teks yang berisi minimal empat baris. Baris pertama menyatakan *operand* pertama, baris kedua menyatakan *operand* kedua dan diakhiri (atau diawali) dengan tanda tambah (+), baris ketiga menyatakan garis -----, dan baris keempat adalah hasil penjumlahan.

Contoh isi file teks:

```
SEND
MORE+
-----
MONEY

FORTY
TEN
TEN+
-----
SIXTY
```

- **Output:** Tampilan di layar yang memuat:
 - a. Persoalan *cryptarithmic* dan solusinya dalam bentuk penjumlahan seperti contoh di bawah ini:

	S	E	N	D		9	5	6	7		
+	M	O	R	E		+	1	0	8	5	
	<hr/>						<hr/>				
	M	O	N	E	Y		1	0	6	5	2

- b. waktu eksekusi program (tidak termasuk waktu pembacaan file input).
- c. Jumlah total tes yang dilakukan untuk menemukan substitusi angka yang benar untuk setiap huruf

BAB II

PEMBAHASAN

1. Algoritma *Brute Force*

Algoritma *brute force* adalah suatu pendekatan yang lempeng (*straightforward*) untuk memecahkan suatu persoalan. Algoritma *brute force* biasanya didasarkan pada pernyataan persoalan (*problem statement*) dan definisi atau konsep yang dilibatkan. Algoritma ini memecahkan persoalan dengan sangat sederhana, langsung, dan jelas caranya.

Contoh-contoh algoritma *brute force*:

- a. Mencari elemen terbesar atau terkecil dari suatu senarai atau *list*:

Algoritma *brute force*: bandingkan setiap elemen senarai mulai dari a_1 sampai a_n untuk menemukan elemen terbesar atau terkecil

- b. Pencarian beruntun

Algoritma *brute force*: setiap elemen senarai dibandingkan dengan x (bilangan yang dicari) . Pencarian selesai jika x ditemukan atau seluruh elemen senarai sudah habis diperiksa

- c. Menghitung faktorial ($n!$)

Algoritma *brute force*: mengalikan bilangan dari n sampai 1 bersama-sama

2. Strategi Penyelesaian

Untuk menyelesaikan permainan *cryptarithmic* ini pada dasarnya adalah mencoba semua kemungkinan bilangan dari 1 sampai 10 kemudian mensubstitusikan kemungkinan angka tersebut pada setiap huruf dan membandingkan apakah substitusi angka tersebut sudah benar atau belum.

Berikut adalah strategi yang saya gunakan untuk menyelesaikan permainan *cryptarithmic* ini sesuai dengan spesifikasi:

1. Membaca input dari file txt
2. Mengolah hasil pembacaan input dari file txt tersebut sehingga siap untuk diselesaikan
3. Mensubstitusikan semua kemungkinan angka dari 1 sampai 10 pada setiap huruf kemudian dibandingkan apakah memenuhi kondisi atau tidak jika memenuhi kombinasi angka tersebut akan disimpan lalu akan diperiksa apakah ada kombinasi angka tersebut dengan substitusi pada setiap huruf pertama yang 0, jika ada

kemungkinan tersebut akan dihilangkan. Kombinasi angka tersebut dibuat berbeda setiap hurufnya (setiap huruf mempresentasikan angka yang berbeda) sehingga jumlah total kombinasi angka yang dites adalah $P(10, n)$, dengan n adalah total huruf pada operand.

4. Menampilkan semua solusinya dan menyimpan solusi tersebut pada file txt

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

BAB III

IMPLEMENTASI PROGRAM

Berikut adalah source code yang saya buat dengan menggunakan python. Source code, hasil kompilasi, dan juga dokumentasi dapat diakses di [github](#) dan [gdrive](#)

```
import time

"""
Nama: Prana Gusriana
NIM: 13519195
TUGAS KECIL 1 IF2211 STRATEGI ALGORITMA - CRYPTARITHMETIC 2020/2021

PROGRAM INI MERUPAKAN PROGRAM UNTUK MENYELESAIKAN PERSOALAN CRYPTARITHMETIC DE
NGAN MENGGUNAKAN ALGORITMA BRUTE FORCE
"""

# CATATAN DAN ASUMSI:
# ASUMSI HURUF PERTAMA TIDAK BOLEH NOL KECUALI PANJANG NYA CUMA 1
# CONTOH: AB + D = AB MAKA A TIDAK BOLEH 0 TAPI D VALID UNTUK 0
# UNTUK PEMBACAAN FILE TIDAK MEMINTA MASUKAN PADA USER TETAPI JIKA INGIN MEMBA
CA FILE
# SILAHKAN GANTI PARAMETER PEMANGGILAN FUNGSI readInputFromFile DAN saveFile D
I MAIN FUNCTION
# CONTOH: NAMA FILE YANG INGIN DIBACA input1.txt DAN INGIN MENYIMPAN FILE PADA
output1.txt
# MAKA UBAH PEMANGGILAN FUNGSI BACA FILE MENJADI readInputFromFile("input1.txt
") DAN saveFile(file_output[0], "output1.txt")

class Cryptarithmic():
    def __init__(self, char, first, currVal):
        self.char = char
        self.first = first
        self.currVal = currVal
        self.query = []
        self.ans = []
        self.posNumber = []

    def setQ(self, arrQ):
        self.query += arrQ

    def setA(self, arrA):
        self.ans += arrA

    def sumQ(self):
        sumq = 0
```

```

        for i in range(len(self.query)):
            sumq += self.currVal * (10**self.query[i])
        return sumq

    def sumA(self):
        suma = 0
        for i in range(len(self.ans)):
            suma += self.currVal * (10**self.ans[i])
        return suma

    def printClass(self):
        print(self.char, self.first, self.currVal, self.query, self.ans)

    def setCurrVal(self, integer):
        self.currVal = integer

    def setFirst(self, first):
        self.first = first

    def setPosNumber(self):
        self.posNumber += [self.currVal]

# FUNGSI UNTUK MEMBACA FILE INPUT DARI FILE
def readInputFromFile(namaFile):
    file = open("../test/"+namaFile, "r")
    file_input = file.readlines()
    i = 0
    while(i < len(file_input)):
        j = 0
        strRet = ""
        tempstr = file_input[i]
        while (j < len(tempstr)):
            if (tempstr[j] != "\n"):
                strRet += tempstr[j]
            j += 1
        file_input[i] = strRet
        i += 1
    file.close()
    return file_input

# FUNGSI UNTUK MEMISAHKAN ANTARA PERTANYAAN DAN JAWABAN
def distinguishOpr(query, ans, file_input):
    i = 0
    stop = False
    while(i < len(file_input) and not(stop)):
        if (file_input[i][0] != '-'):
            if (file_input[i][0] != '+'):
                query += [file_input[i]]

```



```

        else:
            strRet = ""
            p = 1
            while (p < len(file_input[i])):
                strRet += file_input[i][p]
                p += 1
            query += [strRet]
    else:
        stop = True
    i += 1
    ans += [file_input[i]]

# FUNGSI UNTUK MEMBUAT ARRAY CHARACTER UNTUK SUBSTITUSI ANGKA PADA SETIAP HURUF
F
def buildArrChar(arrChar, query, ans):
    for i in range(len(query)):
        for j in range(len(query[i])):
            if (j == 0 and len(query[i]) > 1):
                if (arrChar[ord(query[i][j])-65] == 0):
                    arrChar[ord(query[i][j])-
65] = Cryptarithmic(query[i][j], True, -1)
                else:
                    arrChar[ord(query[i][j])-65].setFirst(True)
            else:
                if (arrChar[ord(query[i][j])-65] == 0):
                    arrChar[ord(query[i][j])-
65] = Cryptarithmic(query[i][j], False, -1)
                    arrChar[ord(query[i][j])-65].setQ([len(query[i])-j-1])

        for i in range(len(ans[0])):
            if (i == 0 and len(ans[0]) > 1):
                if (arrChar[ord(ans[0][i])-65] == 0):
                    arrChar[ord(ans[0][i])-
65] = Cryptarithmic(ans[0][i], True, -1)
                else:
                    arrChar[ord(ans[0][i])-65].setFirst(True)
            else:
                if (arrChar[ord(ans[0][i])-65] == 0):
                    arrChar[ord(ans[0][i])-
65] = Cryptarithmic(ans[0][i], False, -1)
                    arrChar[ord(ans[0][i])-65].setA([len(ans[0])-i-1])

# FUNGSI UNTUK MENJUMLAHKAN TOTAL HURUF UNIK
def sumChar(arrChar):
    sumChr = 0
    for i in range(len(arrChar)):
        if (arrChar[i] != 0):
            sumChr += 1

```

```

        return sumChr

# FUNGSI UNTUK MEMENUHI SYARAT HURUF PERTAMA TIDAK BOLEH NOL
# CATATAN: UNTUK AB+D=AB ITU ASUMSI D NYA VALID UNTUK 0
def cleanFirstnol(arrChar, totest, temptotest):
    idxFirst = []
    idx = 0
    idxAr = []
    for i in range(len(arrChar)):
        if (arrChar[i] != 0):
            if (arrChar[i].first == True):
                idxFirst += [i]
                idxAr += [idx]
            idx += 1
    arr = []
    lenPos = len(arrChar[idxFirst[0]].posNumber)
    for i in range(lenPos):
        tempArr = []
        for j in range(len(arrChar)):
            if (arrChar[j] != 0):
                tempArr += [arrChar[j].posNumber[i]]
        arr += [tempArr]

    for i in range(len(idxFirst)):
        col = idxAr[i]
        tempSol = []
        temptest = []
        for j in range(len(arr)):
            if (arr[j][col] != 0):
                tempSol += [arr[j]]
                temptest += [tostest[j]]
        arr = tempSol
        totest = temptest
    temptotest += totest
    for j in range(len(arrChar)):
        if (arrChar[j] != 0):
            arrChar[j].posNumber = []

    for i in range(len(arr)):
        p = 0
        for j in range(len(arrChar)):
            if (arrChar[j] != 0):
                arrChar[j].posNumber += [arr[i][p]]
                p += 1

# FUNGSI UNTUK MENAMPILKAN SOLUSI
def printHasil(arrChar, query, ans, file_input, file_output):
    file_outputt = ""

```

```

for i in file_input:
    print(i)
    file_outputtt += i + "\n"
arrSol = []
arrOut = []
sumSol = 0
totChar = sumChar(arrChar)
p = 0
stopS = False
while (p < len(arrChar) and not(stopS)):
    if (arrChar[p] != 0):
        sumSol = len(arrChar[p].posNumber)
        stopS = True
    else:
        p+=1
if (sumSol != 0):
    print("=====\n" + "                                Terdapat " + str
(sumSol) + " Solusi dari permainan cryptarithmic ini, yaitu: " + "\n=====
=====")
    file_outputtt += "=====\n" + "                                Terdapat
at " + str(sumSol) + " Solusi dari permainan cryptarithmic ini, yaitu: " + "
\n=====
=====\\n"
    for i in range(sumSol):
        Sol = ""
        idxChar = 0
        for j in range(len(arrChar)):
            if (arrChar[j] != 0):
                Sol += ( str(arrChar[j].char) + " = " + str(arrChar[j].pos
Number[i] ) )
                if (idxChar < totChar - 1):
                    if (idxChar == totChar -2):
                        Sol += ", dan "
                    else:
                        Sol += ", "
                idxChar += 1
        arrSol += [Sol]
        tempArrout = []
        for p in range(len(query)):
            subsChar = ""
            for q in range(len(query[p])):
                subsChar += str(arrChar[ord(query[p][q])-65].posNumber[i])
            if (p != len(query)-1):
                tempArrout += [subsChar]
            else:

```

```

        tempArrout += [subsChar + " +"]
        tempArrout += ["-----"]
        tempans = ""
        for s in range(len(ans[0])):
            tempans += str(arrChar[ord(ans[0][s])-65].posNumber[i])
        tempArrout += [tempans]
        arrOut += [tempArrout]

    for i in range(sumSol):
        print("===== Solusi ke-
[" + str(i+1) + "] : " + arrSol[i] + " =====")
        file_outputt += "===== Solusi ke-
[" + str(i+1) + "] : " + arrSol[i] + " =====\n"
        for j in range(len(arrOut[i])):
            print(arrOut[i][j])
            file_outputt += arrOut[i][j] + "\n"
        print("")
        file_outputt += "\n"
    else:
        print("Tidak memiliki solusi")
        file_outputt += "Tidak memiliki solusi\n"
    file_output += [file_outputt]

# FUNGSI UNTUK MENGUPDATE SUBSTITUSI PADA SETIAP HURUF DAN PENGECEKANAN SOLUSI
def updatePosNumber(arr, arrChar, query, ans, tes, totest):
    idx = 0
    sumq = 0
    suma = 0
    for i in range(len(arrChar)):
        if (arrChar[i] != 0):
            arrChar[i].setCurrVal(arr[idx])
            sumq += arrChar[i].sumQ()
            suma += arrChar[i].sumA()
            idx += 1
    if (sumq == suma):
        totest += [tes]
        for i in range(len(arrChar)):
            if (arrChar[i] != 0):
                arrChar[i].setPosNumber()
                arrChar[i].setCurrVal(-1)
    else:
        for i in range(len(arrChar)):
            if (arrChar[i] != 0):
                arrChar[i].setCurrVal(-1)

# FUNGSI UNTUK MENGECEK ADA YANG MENJADI FIRST ATAU TIDAK (UNTUK DIHILANGKAN S
UBSTITUSI 0 PADA HURUF PERTAMA)
def isFirstExist(arrChar):

```

```

    First = 0
    for i in range(len(arrChar)):
        if (arrChar[i] != 0):
            if (arrChar[i].first == True):
                First += 1

    if (First == 0):
        return False
    else:
        return True

# FUNGSI UNTUK MENYELESAIKAN PERSOALAN CRYPTARITHMETIC
def Solve(arrChar, query, ans, file_input, file_output, tes, totest, tempt):
    totalChar = sumChar(arrChar)

    if (totalChar == 10):
        for first in range(10):
            for scnd in range(10):
                if (scnd != first):
                    for thrd in range(10):
                        if (thrd != first and thrd != scnd):
                            for fourth in range(10):
                                if (fourth != first and fourth != scnd and fourth != thrd):
                                    for fifth in range(10):
                                        if (fifth != first and fifth != scnd and fifth != thrd and fifth != fourth):
                                            for sixth in range(10):
                                                if (sixth != first and sixth != scnd and sixth != thrd and sixth != fourth and sixth != fifth):
                                                    for seventh in range(10):
                                                        if (seventh != first and seventh != scnd and seventh != thrd and seventh != fourth and seventh != fifth and seventh != sixth):
                                                            for eighth in range(10):
                                                                if (eighth != first and eighth != scnd and eighth != thrd and eighth != fourth and eighth != fifth and eighth != sixth and eighth != seventh):
                                                                    for ninth in range(10):
                                                                        if (ninth != first and ninth != scnd and ninth != thrd and ninth != fourth and ninth != fifth and ninth != sixth and ninth != seventh and ninth != eighth):
                                                                            for tenth in range(10):
                                                                                if (tenth != first and tenth != scnd and tenth != thrd and tenth != fourth and tenth != fifth and tenth != sixth and tenth != seventh and tenth != eighth and tenth != ninth):

```

```

nd tenth != fifth and tenth != sixth and tenth != seventh and tenth != eighth
and tenth != ninth):

    arr = [first, scnd, thrd, fourth, fifth, sixth, seventh, eighth, ninth,
tenth]

    updatePosNumber(arr, arrChar, query, ans, tes[0], totest)

    tes[0] += 1
elif (totalChar==9):
    for first in range(10):
        for scnd in range(10):
            if (scnd!=first):
                for thrd in range(10):
                    if (thrd != first and thrd != scnd):
                        for fourth in range (10):
                            if (fourth != first and fourth != scnd and fou
rth != thrd):
                                for fifth in range (10):
                                    if (fifth != first and fifth != scnd a
nd fifth != thrd and fifth != fourth):
                                        for sixth in range(10):
                                            if (sixth != first and sixth !=
scnd and sixth != thrd and sixth != fourth and sixth != fifth):
                                                for seventh in range(10):
                                                    if (seventh != first a
nd seventh != scnd and seventh != thrd and seventh != fourth and seventh != fi
fth and seventh != sixth):
                                                        for eighth in rang
e(10):
                                                            if (eighth !=
first and eighth != scnd and eighth != thrd and eighth != fourth and eighth !=
fifth and eighth != sixth and eighth != seventh):
                                                                for ninth
in range(10):
                                                                    if (ni
neth != first and ninth != scnd and ninth != thrd and ninth != fourth and n
ineth != fifth and ninth != sixth and ninth != seventh and ninth != eighth)
:
                                                                    ar
r = [first, scnd, thrd, fourth, fifth, sixth, seventh, eighth, ninth]
                                                                    up
datePosNumber(arr, arrChar, query, ans, tes[0], totest)
                                                                    te
s[0] += 1
elif (totalChar==8):
    for first in range(10):
        for scnd in range(10):

```

```

        if (scnd!=first):
            for thrd in range(10):
                if (thrd != first and thrd != scnd):
                    for fourth in range (10):
                        if (fourth != first and fourth != scnd and fourth != thrd):
                            for fifth in range (10):
                                if (fifth != first and fifth != scnd and fifth != thrd and fifth != fourth):
                                    for sixth in range(10):
                                        if (sixth != first and sixth != scnd and sixth != thrd and sixth != fourth and sixth != fifth):
                                            for seventh in range(10):
                                                if (seventh != first and seventh != scnd and seventh != thrd and seventh != fourth and seventh != fifth and seventh != sixth):
                                                    for eighth in range(10):
                                                        if (eighth != first and eighth != scnd and eighth != thrd and eighth != fourth and eighth != fifth and eighth != sixth and eighth != seventh):
                                                            arr = [first, scnd, thrd, fourth, fifth, sixth, seventh, eighth]
                                                            updatePosNumber(arr, arrChar, query, ans, tes[0], totest)
                                                            tes[0] += 1
            elif (totalChar==7):
                for first in range(10):
                    for scnd in range(10):
                        if (scnd!=first):
                            for thrd in range(10):
                                if (thrd != first and thrd != scnd):
                                    for fourth in range (10):
                                        if (fourth != first and fourth != scnd and fourth != thrd):
                                            for fifth in range (10):
                                                if (fifth != first and fifth != scnd and fifth != thrd and fifth != fourth):
                                                    for sixth in range(10):
                                                        if (sixth != first and sixth != scnd and sixth != thrd and sixth != fourth and sixth != fifth):
                                                            for seventh in range(10):
                                                                if (seventh != first and seventh != scnd and seventh != thrd and seventh != fourth and seventh != fifth and seventh != sixth):
                                                                    arr = [first, scnd, thrd, fourth, fifth, sixth, seventh]

```

```

updatePosNumber(arrChar, query, ans, tes[0], totest)
tes[0] += 1

elif (totalChar==6):
    for first in range(10):
        for scnd in range(10):
            if (scnd!=first):
                for thrd in range(10):
                    if (thrd != first and thrd != scnd):
                        for fourth in range (10):
                            if (fourth != first and fourth != scnd and fourth != thrd):
                                for fifth in range (10):
                                    if (fifth != first and fifth != scnd and fifth != thrd and fifth != fourth):
                                        for sixth in range(10):
                                            if (sixth != first and sixth != scnd and sixth != thrd and sixth != fourth and sixth != fifth):
                                                arr = [first, scnd, thrd, fourth, fifth, sixth]
                                                updatePosNumber(arr, arrChar, query, ans, tes[0], totest)
                                                tes[0] += 1

elif (totalChar==5):
    for first in range(10):
        for scnd in range(10):
            if (scnd!=first):
                for thrd in range(10):
                    if (thrd != first and thrd != scnd):
                        for fourth in range (10):
                            if (fourth != first and fourth != scnd and fourth != thrd):
                                for fifth in range (10):
                                    if (fifth != first and fifth != scnd and fifth != thrd and fifth != fourth):
                                        arr = [first, scnd, thrd, fourth, fifth]
                                        updatePosNumber(arr, arrChar, query, ans, tes[0], totest)
                                        tes[0] += 1

elif (totalChar==4):
    for first in range(10):
        for scnd in range(10):
            if (scnd!=first):
                for thrd in range(10):
                    if (thrd != first and thrd != scnd):
                        for fourth in range (10):

```



```

        if (fourth != first and fourth != scnd and fourth != thrd):
            arr = [first, scnd, thrd, fourth]
            updatePosNumber(arr, arrChar, query, ans, tes[0], totest)
            tes[0] += 1

    elif (totalChar==3):
        for first in range(10):
            for scnd in range(10):
                if (scnd!=first):
                    for thrd in range(10):
                        if (thrd != first and thrd != scnd):
                            arr = [first, scnd, thrd]
                            updatePosNumber(arr, arrChar, query, ans, tes[0], totest)
                            tes[0] += 1

    elif (totalChar==2):
        for first in range(10):
            for scnd in range(10):
                if (scnd!=first):
                    arr = [first, scnd]
                    updatePosNumber(arr, arrChar, query, ans, tes[0], totest)
                    tes[0] += 1

    elif (totalChar==1):
        for first in range(10):
            arr = [first]
            updatePosNumber(arr, arrChar, query, ans, tes[0], totest)
            tes[0] += 1

    else:
        file_outputt = ""
        for i in file_input:
            print(i)
            file_outputt += i + "\n"
        print("=====\n
        Maaf jumlah total huruf masukk
        annya lebih dari 10 huruf, tidak bisa diselesaikan :)\n=====
        \n")
        file_outputt += "=====\n
        Maaf jumlah total hu
        ruf masukkannya lebih dari 10 huruf, tidak bisa diselesaikan :)\n=====
        \n"
        file_output += [file_outputt]
    if (1<= totalChar <= 10):
        temptotest = []
        if (isFirstExist(arrChar)):

```

```

        cleanFirstnol(arrChar, totest, temptotest)
        totest = temptotest
        tempt += totest
        printHasil(arrChar, query, ans, file_input, file_output)

# FUNGSI UNTUK SAVE OUTPUT PADA FILE
def saveFile(file_output, namaFile):
    file = open("../test/"+namaFile, "w")
    file.write(file_output)
    file.close()

# MAIN
def Main():
    arrChar = [0 for i in range(26)]
    file_input = readInputFromFile("input10.txt")
    query = []
    ans = []
    file_output = []
    tes = [0]
    totest = []
    tempt = []
    distinguishOpr(query, ans, file_input)
    buildArrChar(arrChar, query, ans)
    start_time = time.time()
    Solve(arrChar, query, ans, file_input, file_output, tes, totest, tempt)
    print("=====
=====")
    print("----- %s seconds -----
---" %(time.time() - start_time))
    print("----- "+ str(tes[0]) + " total test -----
-----")
    file_output[0] += "=====
=====\\n" + "-----
---- %s seconds -----
\\n" %(time.time() - start_time) + "-----
- "+ str(tes[0]) + " total test -----\\n"
    for i in range(len(tempt)):
        print("Solusi ke-[" + str(i+1) + "] ditemukan pada test ke-
" + str(tempt[i]))
        file_output[0] += "Solusi ke-
[" + str(i+1) + "] ditemukan pada test ke-" + str(tempt[i]) + "\\n"
        saveFile(file_output[0], "output10.txt")

Main()

```

BAB IV

EKSPERIMEN

Untuk gambar atau hasil kompilasi dapat dilihat difolder gambar dan doc, pada file submisi, github, gdrive

1. Hasil kompilasi dengan input: JAMES + LYLY = HARRY

```
JAMES
+LYLY
-----
HARRY

=====
                Terdapat 32 Solusi dari permainan cryptarithmic ini, yaitu:
=====
===== Solusi ke-[1] : A = 1, E = 3, H = 6, J = 5, L = 9, M = 4, R = 2, S = 0, dan Y = 7 =====
51430
9797 +
-----
61227

===== Solusi ke-[2] : A = 1, E = 3, H = 6, J = 5, L = 9, M = 7, R = 2, S = 0, dan Y = 4 =====
51730
9494 +
-----
61224

===== Solusi ke-[3] : A = 1, E = 3, H = 8, J = 7, L = 9, M = 5, R = 2, S = 0, dan Y = 6 =====
71530
9696 +
-----
81226

===== Solusi ke-[4] : A = 1, E = 3, H = 8, J = 7, L = 9, M = 6, R = 2, S = 0, dan Y = 5 =====
71630
9595 +
-----
81225

===== Solusi ke-[5] : A = 1, E = 5, H = 3, J = 2, L = 9, M = 6, R = 4, S = 0, dan Y = 7 =====
21650
9797 +
-----
31447

===== Solusi ke-[6] : A = 1, E = 5, H = 3, J = 2, L = 9, M = 7, R = 4, S = 0, dan Y = 6 =====
21750
9696 +
-----
31446

===== Solusi ke-[32] : A = 8, E = 5, H = 3, J = 2, L = 9, M = 7, R = 4, S = 0, dan Y = 6 =====
28750
9696 +
-----
38446

=====
----- 267.0352156162262 seconds -----
----- 3628800 total test -----

Solusi ke-[1] ditemukan pada test ke-466494
Solusi ke-[2] ditemukan pada test ke-466518
Solusi ke-[3] ditemukan pada test ke-478039
Solusi ke-[4] ditemukan pada test ke-478063
Solusi ke-[5] ditemukan pada test ke-535614
Solusi ke-[6] ditemukan pada test ke-535638
Solusi ke-[7] ditemukan pada test ke-1203775
Solusi ke-[8] ditemukan pada test ke-1203822
Solusi ke-[9] ditemukan pada test ke-1261374
Solusi ke-[10] ditemukan pada test ke-1261398
Solusi ke-[11] ditemukan pada test ke-1555134
Solusi ke-[12] ditemukan pada test ke-1555158
Solusi ke-[13] ditemukan pada test ke-1607005
Solusi ke-[14] ditemukan pada test ke-1607029
Solusi ke-[15] ditemukan pada test ke-1929559
Solusi ke-[16] ditemukan pada test ke-1929583
Solusi ke-[17] ditemukan pada test ke-2280894
Solusi ke-[18] ditemukan pada test ke-2280918
Solusi ke-[19] ditemukan pada test ke-2350014
Solusi ke-[20] ditemukan pada test ke-2350038
Solusi ke-[21] ditemukan pada test ke-3006655
Solusi ke-[22] ditemukan pada test ke-3006702
Solusi ke-[23] ditemukan pada test ke-3012415
Solusi ke-[24] ditemukan pada test ke-3012462
Solusi ke-[25] ditemukan pada test ke-3052765
Solusi ke-[26] ditemukan pada test ke-3052789
Solusi ke-[27] ditemukan pada test ke-3075775
Solusi ke-[28] ditemukan pada test ke-3075822
Solusi ke-[29] ditemukan pada test ke-3116125
Solusi ke-[30] ditemukan pada test ke-3116149
Solusi ke-[31] ditemukan pada test ke-3121985
Solusi ke-[32] ditemukan pada test ke-3121909
```

2. Hasil kompilasi dengan input: SEND + MORE = MONEY

```
SEND
+MORE
-----
MONEY
=====
Terdapat 1 Solusi dari permainan cryptarithmic ini, yaitu:
===== Solusi ke-[1] : D = 7, E = 5, M = 1, N = 6, O = 0, R = 8, S = 9, dan Y = 2 =====
9567
1085 +
-----
10652
=====
----- 116.2174813747406 seconds -----
----- 1814400 total test -----
Solusi ke-[1] ditemukan pada test ke-1374885
```

3. Hasil kompilasi dengan input: THREE + THREE + TWO + TWO + ONE = ELEVEN

```
THREE
THREE
TWO
TWO
+ONE
-----
ELEVEN
=====
Terdapat 1 Solusi dari permainan cryptarithmic ini, yaitu:
===== Solusi ke-[1] : E = 1, H = 4, L = 7, N = 9, O = 3, R = 6, T = 8, V = 2, dan W = 0 =====
84611
84611
803
803
391 +
-----
171219
=====
----- 303.4698135852814 seconds -----
----- 3628800 total test -----
Solusi ke-[1] ditemukan pada test ke-513692
```

4. Hasil kompilasi dengan input: NO + GUN + NO = HUNT

```
NO
GUN
+NO
-----
HUNT
=====
Terdapat 1 Solusi dari permainan cryptarithmic ini, yaitu:
===== Solusi ke-[1] : G = 9, H = 1, N = 8, O = 7, T = 2, dan U = 0 =====
87
908
87 +
-----
1082
=====
----- 7.9057745933532715 seconds -----
----- 151200 total test -----
Solusi ke-[1] ditemukan pada test ke-139415
```

5. Hasil kompilasi dengan input: HERE + SHE = COMES

```
HERE
+SHE
-----
COMES

=====
Terdapat 1 Solusi dari permainan cryptarithmic ini, yaitu:
=====
===== Solusi ke-[1] : C = 1, E = 4, H = 9, M = 3, O = 0, R = 5, dan S = 8 =====
9454
894 +
-----
10348

=====
----- 35.31456661224365 seconds -----
----- 604800 total test -----
Solusi ke-[1] ditemukan pada test ke-86767
```

6. Hasil kompilasi dengan input: CLOCK + TICK + TOCK = PLANET

```
CLOCK
TICK
+TOCK
-----
PLANET

=====
Terdapat 1 Solusi dari permainan cryptarithmic ini, yaitu:
=====
===== Solusi ke-[1] : A = 4, C = 9, E = 7, I = 5, K = 2, L = 0, N = 3, O = 8, P = 1, dan T = 6 =====
90892
6592
6892 +
-----
104376

=====
----- 321.7078456878662 seconds -----
----- 3628800 total test -----
Solusi ke-[1] ditemukan pada test ke-1807450
```

7. Hasil kompilasi dengan input: $AB + D = AB$

```
AB
+D
-----
AB
=====
Terdapat 72 Solusi dari permainan cryptarithmic ini, yaitu:
===== Solusi ke-[1] : A = 1, B = 2, dan D = 0 =====
12
0 +
-----
12

===== Solusi ke-[2] : A = 1, B = 3, dan D = 0 =====
13
0 +
-----
13

===== Solusi ke-[3] : A = 1, B = 4, dan D = 0 =====
14
0 +
-----
14

===== Solusi ke-[4] : A = 1, B = 5, dan D = 0 =====
15
0 +
-----
15

===== Solusi ke-[5] : A = 1, B = 6, dan D = 0 =====
16
0 +
-----
16

===== Solusi ke-[6] : A = 1, B = 7, dan D = 0 =====
17
0 +
-----
17

-----
92

===== Solusi ke-[67] : A = 9, B = 3, dan D = 0 =====
93
0 +
-----
93

===== Solusi ke-[68] : A = 9, B = 4, dan D = 0 =====
94
0 +
-----
94

===== Solusi ke-[69] : A = 9, B = 5, dan D = 0 =====
95
0 +
-----
95

===== Solusi ke-[70] : A = 9, B = 6, dan D = 0 =====
96
0 +
-----
96

===== Solusi ke-[71] : A = 9, B = 7, dan D = 0 =====
97
0 +
-----
97

===== Solusi ke-[72] : A = 9, B = 8, dan D = 0 =====
98
0 +
-----
98

=====
----- 7.17163348197937 seconds -----
----- 720 total test -----
```

```

Solusi ke-[31] ditemukan pada test ke-344
Solusi ke-[32] ditemukan pada test ke-352
Solusi ke-[33] ditemukan pada test ke-368
Solusi ke-[34] ditemukan pada test ke-376
Solusi ke-[35] ditemukan pada test ke-384
Solusi ke-[36] ditemukan pada test ke-392
Solusi ke-[37] ditemukan pada test ke-400
Solusi ke-[38] ditemukan pada test ke-408
Solusi ke-[39] ditemukan pada test ke-416
Solusi ke-[40] ditemukan pada test ke-424
Solusi ke-[41] ditemukan pada test ke-440
Solusi ke-[42] ditemukan pada test ke-448
Solusi ke-[43] ditemukan pada test ke-456
Solusi ke-[44] ditemukan pada test ke-464
Solusi ke-[45] ditemukan pada test ke-472
Solusi ke-[46] ditemukan pada test ke-480
Solusi ke-[47] ditemukan pada test ke-488
Solusi ke-[48] ditemukan pada test ke-496
Solusi ke-[49] ditemukan pada test ke-512
Solusi ke-[50] ditemukan pada test ke-520
Solusi ke-[51] ditemukan pada test ke-528
Solusi ke-[52] ditemukan pada test ke-536
Solusi ke-[53] ditemukan pada test ke-544
Solusi ke-[54] ditemukan pada test ke-552
Solusi ke-[55] ditemukan pada test ke-560
Solusi ke-[56] ditemukan pada test ke-568
Solusi ke-[57] ditemukan pada test ke-584
Solusi ke-[58] ditemukan pada test ke-592
Solusi ke-[59] ditemukan pada test ke-600
Solusi ke-[60] ditemukan pada test ke-608
Solusi ke-[61] ditemukan pada test ke-616
Solusi ke-[62] ditemukan pada test ke-624
Solusi ke-[63] ditemukan pada test ke-632
Solusi ke-[64] ditemukan pada test ke-640
Solusi ke-[65] ditemukan pada test ke-656
Solusi ke-[66] ditemukan pada test ke-664
Solusi ke-[67] ditemukan pada test ke-672
Solusi ke-[68] ditemukan pada test ke-680
Solusi ke-[69] ditemukan pada test ke-688
Solusi ke-[70] ditemukan pada test ke-696
Solusi ke-[71] ditemukan pada test ke-704
Solusi ke-[72] ditemukan pada test ke-712

```

8. Hasil kompilasi dengan input: FORTY + TEN + TEN = SIXTY

```

FORTY
TEN
+TEN
-----
SIXTY
=====
Terdapat 1 Solusi dari permainan cryptarithmic ini, yaitu:
===== Solusi ke-[1] : E = 5, F = 2, I = 1, N = 0, O = 9, R = 7, S = 3, T = 8, X = 4, dan Y = 6 =====
29786
850
850 +
-----
31486

=====
----- 298.75089025497437 seconds -----
----- 3628800 total test -----
Solusi ke-[1] ditemukan pada test ke-1900756

```

9. Hasil kompilasi dengan input: HARRY + GINNY = ALBUS

```
HARRY
+GINNY
-----
ALBUS

=====
Maaf jumlah total huruf masukannya lebih dari 10 huruf, tidak bisa diselesaikan :)
=====

----- 0.12466907501220703 seconds -----
----- 0 total test -----
```

10. Hasil kompilasi dengan input: AKU + KAMU = KITA

```
AKU
+KAMU
-----
KITA

=====
Terdapat 75 Solusi dari permainan cryptarithmic ini, yaitu:
=====
===== Solusi ke-[1] : A = 2, I = 4, K = 1, M = 3, T = 5, dan U = 6 =====
216
1236 +
-----
1452

===== Solusi ke-[2] : A = 2, I = 4, K = 1, M = 5, T = 7, dan U = 6 =====
216
1256 +
-----
1472

===== Solusi ke-[3] : A = 2, I = 4, K = 1, M = 7, T = 9, dan U = 6 =====
216
1276 +
-----
1492

===== Solusi ke-[4] : A = 2, I = 4, K = 3, M = 1, T = 5, dan U = 6 =====
236
3216 +
-----
3452

===== Solusi ke-[5] : A = 2, I = 4, K = 3, M = 5, T = 8, dan U = 1 =====
231
3251 +
-----
3482

===== Solusi ke-[6] : A = 2, I = 4, K = 3, M = 5, T = 9, dan U = 6 =====
236
3256 +
-----
3492

===== Solusi ke-[71] : A = 4, I = 9, K = 8, M = 3, T = 1, dan U = 2 =====
482
8432 +
-----
8914

===== Solusi ke-[72] : A = 4, I = 9, K = 8, M = 3, T = 2, dan U = 7 =====
487
8437 +
-----
8924

===== Solusi ke-[73] : A = 4, I = 9, K = 8, M = 5, T = 3, dan U = 2 =====
482
8452 +
-----
8934

===== Solusi ke-[74] : A = 4, I = 9, K = 8, M = 6, T = 5, dan U = 7 =====
487
8467 +
-----
8954

===== Solusi ke-[75] : A = 4, I = 9, K = 8, M = 7, T = 5, dan U = 2 =====
482
8472 +
-----
8954

=====
----- 14.624305486679077 seconds -----
----- 151200 total test -----
Solusi ke-[1] ditemukan pada test ke-35526
Solusi ke-[2] ditemukan pada test ke-35567
Solusi ke-[3] ditemukan pada test ke-35638
Solusi ke-[4] ditemukan pada test ke-35736
Solusi ke-[5] ditemukan pada test ke-35781
Solusi ke-[6] ditemukan pada test ke-35787
Solusi ke-[7] ditemukan pada test ke-35816
Solusi ke-[8] ditemukan pada test ke-35957
```


BAB V

SARAN, KESIMPULAN, DAN REFLEKSI

1. Kesimpulan

Tugas kecil 1 mata kuliah IF2211 Strategi Algoritma ini sudah bisa diselesaikan sesuai spesifikasi.

2. Saran

Semoga tugas-tugas selanjutnya bisa dikerjakan dengan lebih baik lagi.

3. Refleksi

Jujur saja karena bingung dalam pengerjaan tugas kecil ini, saya mengerjakannya dengan cara yang sangat manual yaitu dengan mencoba semua kemungkinan angka dengan menggunakan looping sebanyak jumlah huruf unik sehingga waktu kompilasinya sangat lama (Bisa 320 detik). Semoga kedepannya lebih baik lagi .