**LAPORAN TUGAS KECIL**

Diajukan untuk memenuhi tugas kecil 1

IF2211 Strategi Algoritma Semester II tahun 2020/2021

Penyelesaian Cryptarithmetic dengan Algoritma Brute Force



Oleh

13518136 Muhammad Rizal Muhaimin

**PROGRAM STUDI TEKNIK INFORMATIKA**

**SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA**

**INSTITUT TEKNOLOGI BANDUNG 2021**

**ALGORITMA BRUTE FORCE**

Brute-force merupakan suatu jenis pendekatan penyelesaian masalah yang bersifat sederhana, jelas, dan juga langsung. Penyelesaian algoritma brute-force dengan mengunakan kombinatorik sering disebut sebagai algoritma exhaustive search. Di mana pencocokan dilakukan untuk semua kemungkinan kunci yang ada sehingga algoritma ini mampu menyelesaikan hampir seluruh persoalan. Namun, algoritma ini merupakan algoritma yang jarang menghasilkan algoritma yang mangkus dan sering tidak diterima karena lamanya pengerjaan.

Termasuk dalam persoalan cryptarithmetic dapat diselesaikan dengan menggunakan algoritma brute-force. Suatu solusi dari cryptarithmetic akan didapatkaan dengan mengiterasi setiap kemungkinan dari kombinasi huruf. Pengerjaan dengan algoritma brute-force adalah sebagai berikut.

1. Nyatakan setiap huruf sebagai suatu angka. Masing -masing angka harus berbeda
2. Cek kebenaran dari penjumlahan kata tersebut dengan kata solusi.
3. Apabila salah, representasikan ulang angka pada huruf. Tambahkan satu pada representasi huruf untuk mencegah pengulangan dan pastikan angka unik (kombinasi).
4. Ulangi langkat 1 sampai 3, sampai menemukan solusi.

Sebagai contoh:

SEND

MORE

----------+

MONEY

Pertama, untuk setiap huruf yang unik dibentuk string baru. SEND + MORE = MONEY menjadi string baru yaitu, SENDMORY. Lalu untuk setiap karakter inisialisasi nilai, maka:

SENDMORY

0 1 2 3 4 5 6 7

Cek apakah penjumlahan dengan angka tersebut sudah benar.

0123

4561

-------+

45217

Karena penjumlahan belum benar, maka dilakukan pengecekan terhadap representasi nilai karakter berikutnya.

SENDMORY

0 1 2 3 4 5 6 8

Lakukan pengecekan terhadap nilai representasi baru.

0123

4561

-------+

45218

Penjumlahan masih belum sesuai, lakukan terus penambahan nilai. Apabila nilai satuan sudah mencapai 9, tambahkan 1 pada nilai puluhan dan ubah nilai 9 menjadi 0, sama seperti penjumlahan biasa. Iterasi penambahan representasi dilakukan terus-menerus sampai terdapat nilai penjumlahan yang sesuai dan nilai mengikuti aturan yang sesuai juga (tidak ada angka duplikat, tidak diawali 0).

**SOURCE PROGRAM**

Dalam tugas penyelesaian cryptarithmetic dengan algoritma brute force kali ini menggunakan bahasa python dan berikut ini *source* programnya:

#Nama : Muhammad Rizal Muhaimin

#NIM  : 13519136

#Deskripsi: Penyelesaian Cryptarithmetic dengan Algoritma Brute Force

import timeit

import os

def Tampilan\_Awal():

    print(

    """

=========================================================

Penyelesaian Cryptarithmetic dengan Algoritma Brute Force

        Tugas Kecil 1 IF2211 Strategi Algoritma

                Semester II tahun 2020/2021

========================================================\n"""

            )

def String\_unik(S1,S2):

    #membuat kata unik dari inputan yang ada

    Str\_Unix =""

for huruf in S1:

if huruf not in Str\_Unix:

Str\_Unix += huruf

for huruf in S2:

if huruf not in Str\_Unix:

Str\_Unix += huruf

return Str\_Unix

def Str\_to\_angka(Str\_Input,Str\_Unik, List\_acuan):

    #mengembalikan kata angka kedalam integer

    angka = ''

    for huruf in Str\_Input:

        idx = 0

        while huruf != Str\_Unik[idx]:

            idx += 1

        angka += str(List\_acuan[idx])

    return int(angka)

def Awal\_Kata\_Not\_Nol(Str\_input, Str\_unik,List\_acuan):

    #Mengembalikan true ketika awal kata tidak direpresentasikan dengan nol (0)

    idx = 0

    while Str\_input[0] != Str\_unik[idx]:

        idx +=1

    return (List\_acuan[idx] != 0)

def Semua\_Awal\_Not\_Nol\_rekursif(L,n,s,Str\_Unik,List\_acuan):

    #memastikan tidak ada hufuf awal di repesentasikan dengan nol (0)

    if s == n:

        return Awal\_Kata\_Not\_Nol(L[s-1],Str\_Unik, List\_acuan)

    else:

        return Awal\_Kata\_Not\_Nol(L[s-1],Str\_Unik, List\_acuan) and Semua\_Awal\_Not\_Nol\_rekursif(L,n,s+1,Str\_Unik, List\_acuan)

def incramentList(L):

    #proses incrament permutasi

    j = L[len(L)-1]

    while j in L:

        j += 1

    L[len(L)-1] = j

    while int(10) in L[1:]:

        idx = 0

        while L[idx] != 10:

            idx += 1

        j = L[idx-1]

        while j in L[0:idx]:

            j += 1

        L[idx-1] = j

        j = 0

        while j in L[0:idx]:

            j += 1

        L[idx] = j

        if(idx +1 < len(L)):

            for i in range(idx+1,len(L)):

                if L[i] in L[0:i]:

                    j = 0

                    while j in L[0:i]:

                        j+=1

                    L[i] = j

def List\_file\_test ():

    #membaca list file yang ada di file test

    print("Daftar file test:\n")

    path = "../test"

    Dir\_test = os.listdir(path)

    for namefile in Dir\_test:

        print(namefile)

def Read\_file\_test():

    #membaca file dan menampilkan data file dan mengembalikan list string

    nfile = input("\nMasukkan Nama File\n>:")

    try:

        path = "../test/"+str(nfile)

        File = open(path, 'r')

        print("\nCryptarithmetic:")

        List\_kata =[]

        Line = File.readline()

        List\_kata.append(Line.strip())

        while Line:

            print(Line, end="")

            Line = File.readline()

            List\_kata.append(Line.strip())

        print("\n")

        List\_kata.pop()

        List\_kata[len(List\_kata)-2] = List\_kata[len(List\_kata)-1]

        List\_kata.pop()

List\_kata[len(List\_kata)-2] = List\_kata[len(List\_kata)-2][:len(List\_kata[len(List\_kata)-2])-1]

        return List\_kata

    except:

        print("Maaf Nama File salah")

def Cryptarithmetic\_dengan\_Algoritma\_Brute\_Force():

    #PROGRAM UTAMA

    List\_input = Read\_file\_test() #def

    S\_Unik = ""

    for i in range(len(List\_input)-1):

        if i == 0 :

            S\_Unik = String\_unik(List\_input[i], List\_input[i+1]) #def

        else:

            S\_Unik = String\_unik(S\_Unik,List\_input[i+1]) #def

    listAngka = [i for i in range(len(S\_Unik))] #representasi S\_unik dalam anggka

    start = timeit.default\_timer() # catat waktu mulai

    n\_permutasi = 1

    while listAngka[0] !=10: #konodisi berhenti

        found = False

        while (not found):

            List\_nilai =[]

            for kata in List\_input:

                List\_nilai.append( Str\_to\_angka(kata,S\_Unik, listAngka))

            if (sum(List\_nilai[0:len(List\_nilai)-1]) == List\_nilai[len(List\_nilai)-1] and Semua\_Awal\_Not\_Nol\_rekursif(List\_input,len(List\_input),1,S\_Unik,listAngka)):

                found = True #kondisi ditemukan

            else:

                incramentList(listAngka)

                n\_permutasi += 1

                if(listAngka[0] ==10): break

        if(listAngka[0] == 10): break

        stop = timeit.default\_timer() # catat waktu selesai

        lama\_eksekusi = stop - start # lama eksekusi dalam satuan detik

        print("Salah satu solusinya adalah:")

        Total = str(Str\_to\_angka(List\_input[len(List\_input)-1],S\_Unik, listAngka))

        for i in range(len(List\_input)):

            if i == len(List\_input)-1:

                print("-"\*len(Total)+"+")

            else:

                n = len(Total) - len(str(Str\_to\_angka(List\_input[i],S\_Unik, listAngka)))

                print(" "\*n,end='')

            print(Str\_to\_angka(List\_input[i],S\_Unik, listAngka))

        print("\nDengan",end=" ")

        for i in range(len(S\_Unik)):

            print(S\_Unik[i],":",listAngka[i],end=", ")

        print("\nLama eksekusi: ",lama\_eksekusi,"detik &",n\_permutasi,"permutasi.\n")

        incramentList(listAngka)

        n\_permutasi += 1

    print("""

=========================================================

                   Proses Selesai

=========================================================""")

# RUN PROGRAM

run = True

while run:

    Tampilan\_Awal()

    List\_file\_test ()

    Cryptarithmetic\_dengan\_Algoritma\_Brute\_Force()

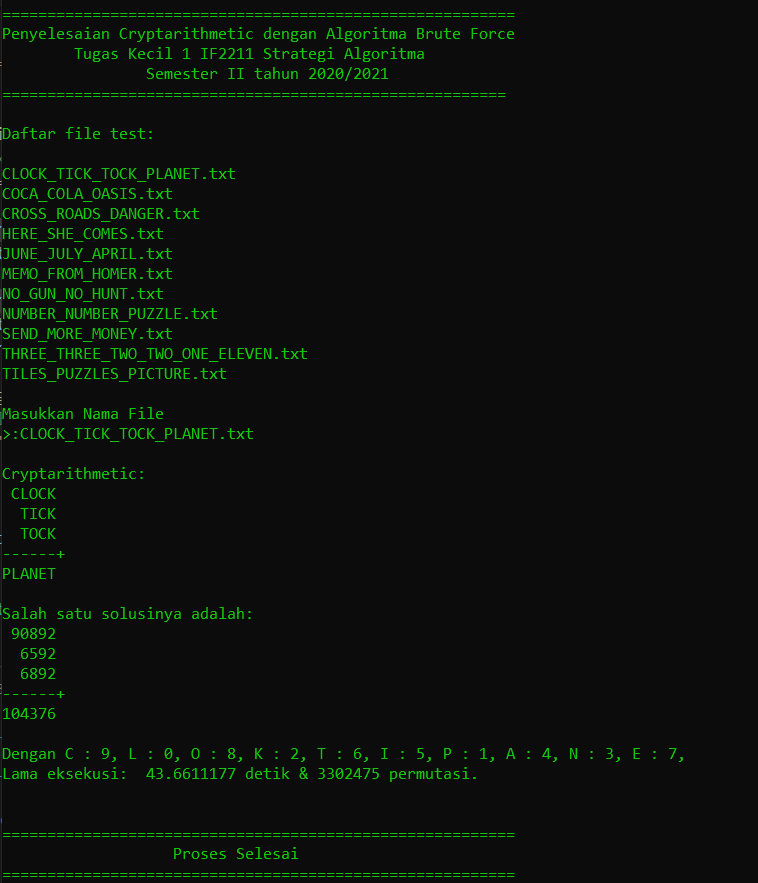
    N = input("Play Again? (y) :> ")

    if N != "y":

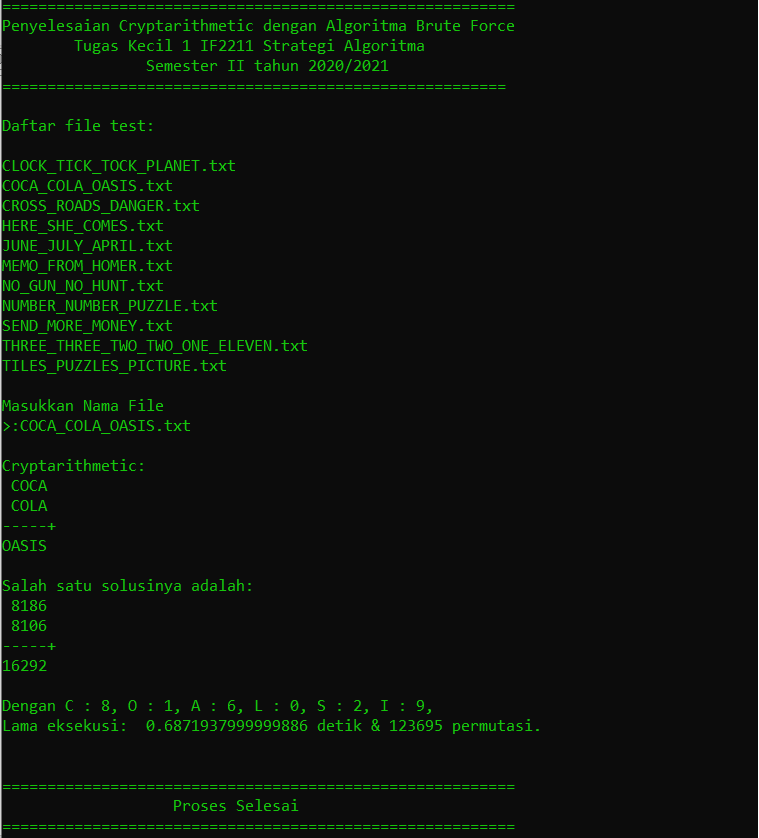
        run = False

**SKRINSHUT INPUT DAN OUTPUT**

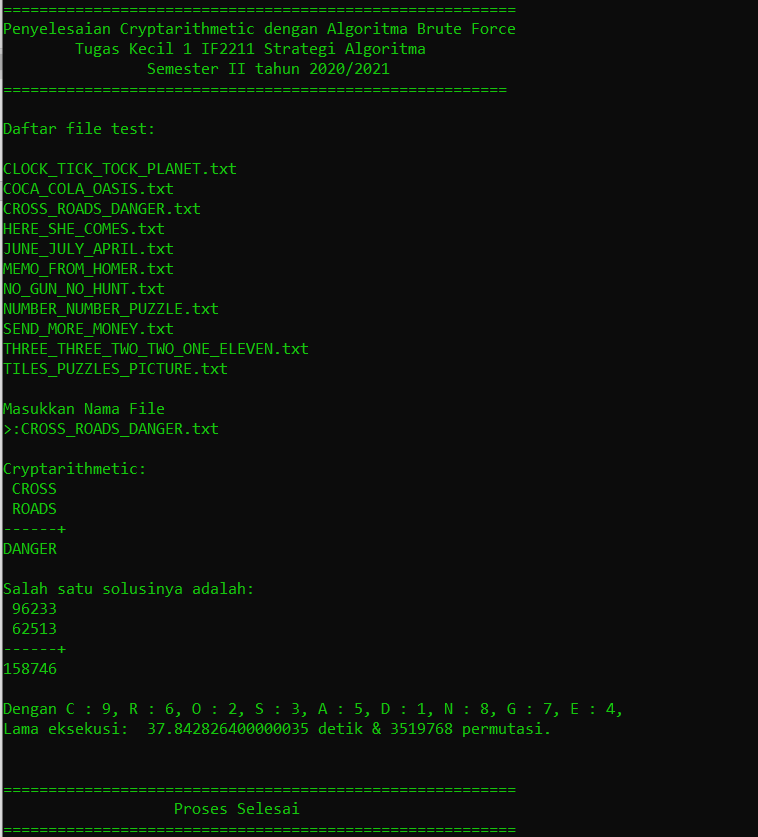
1. **CLOCK\_TICK\_TOCK\_PLANET.txt**

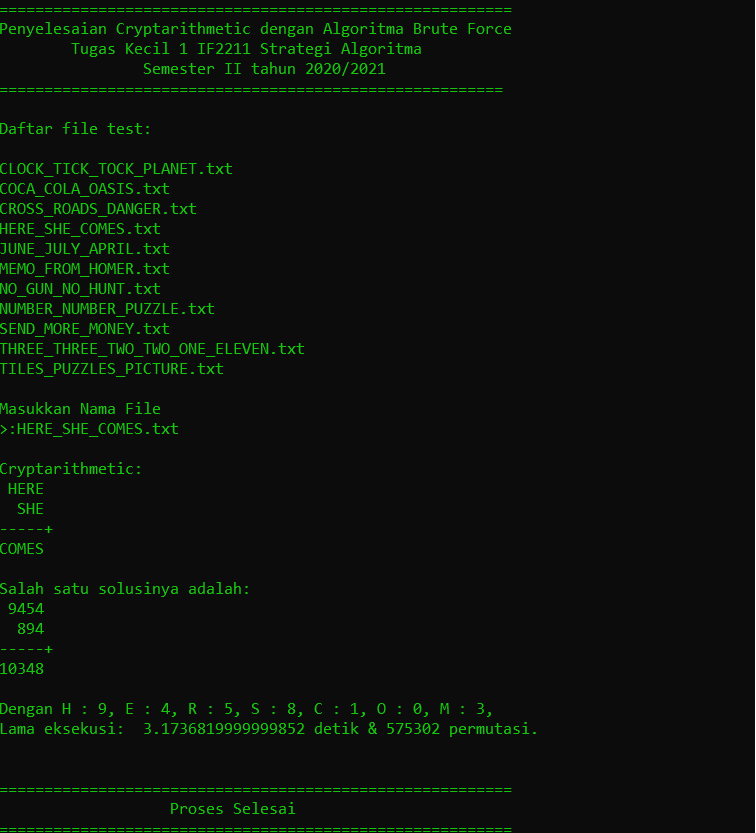
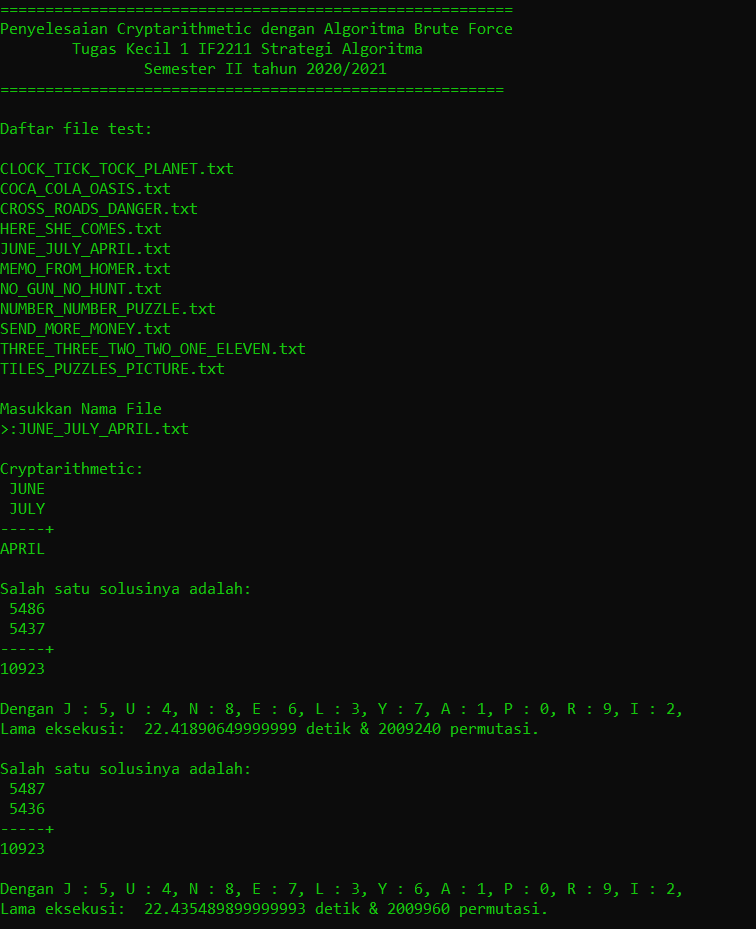


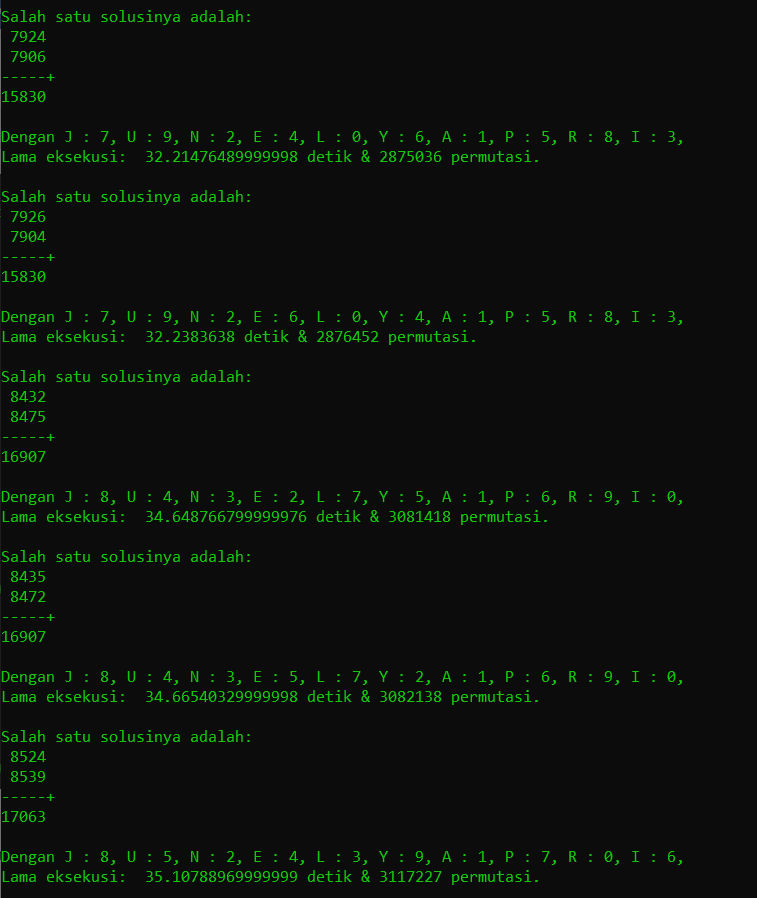
1. **COCA\_COLA\_OASIS.txt**

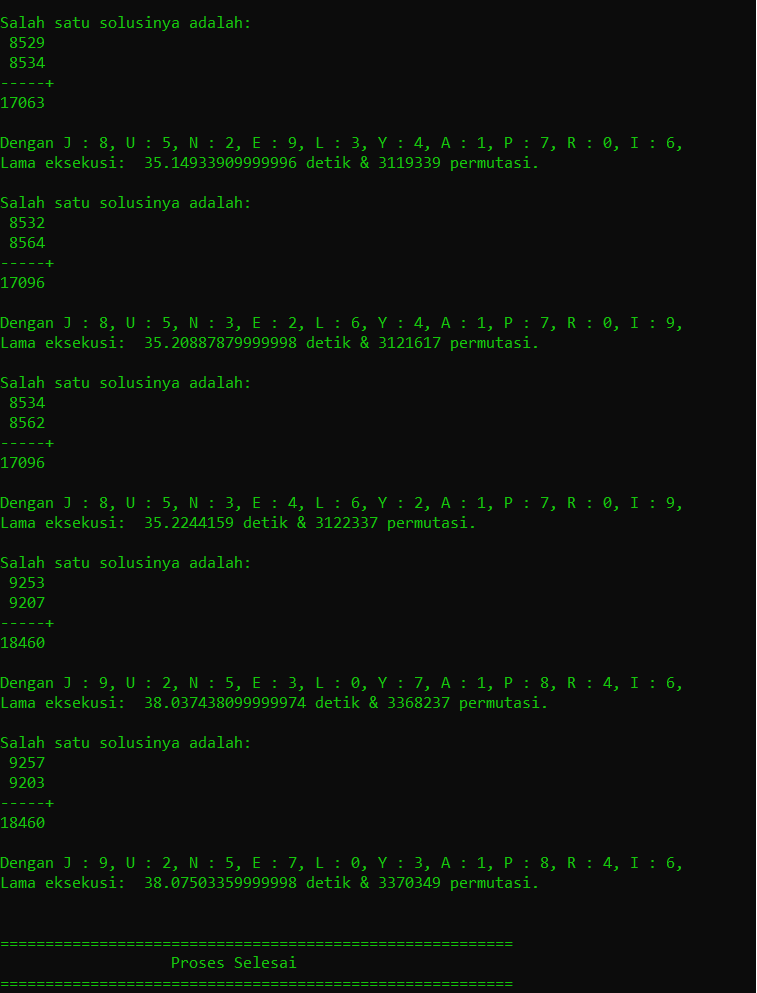


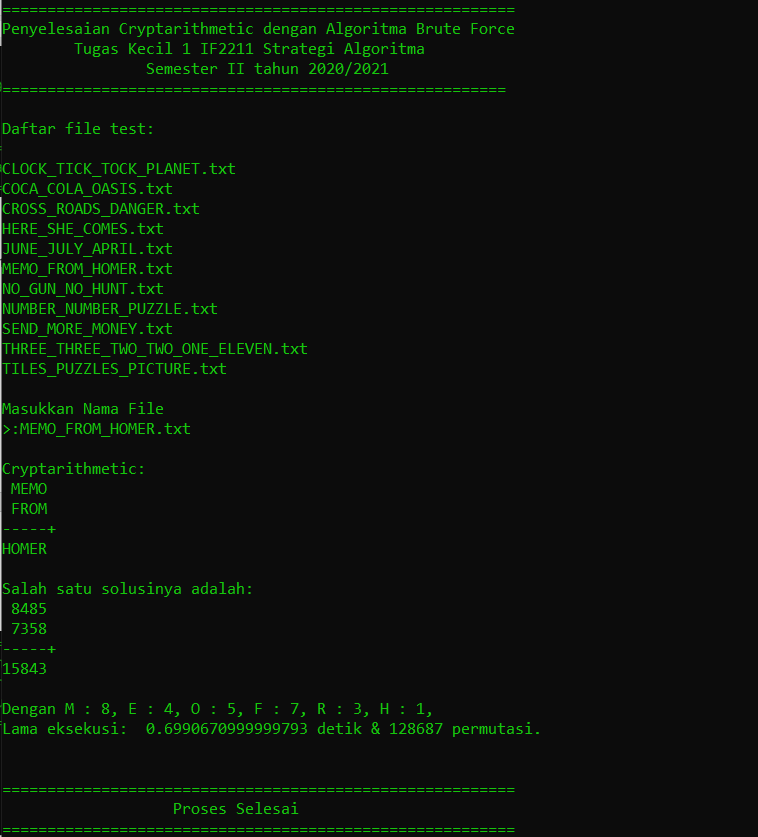
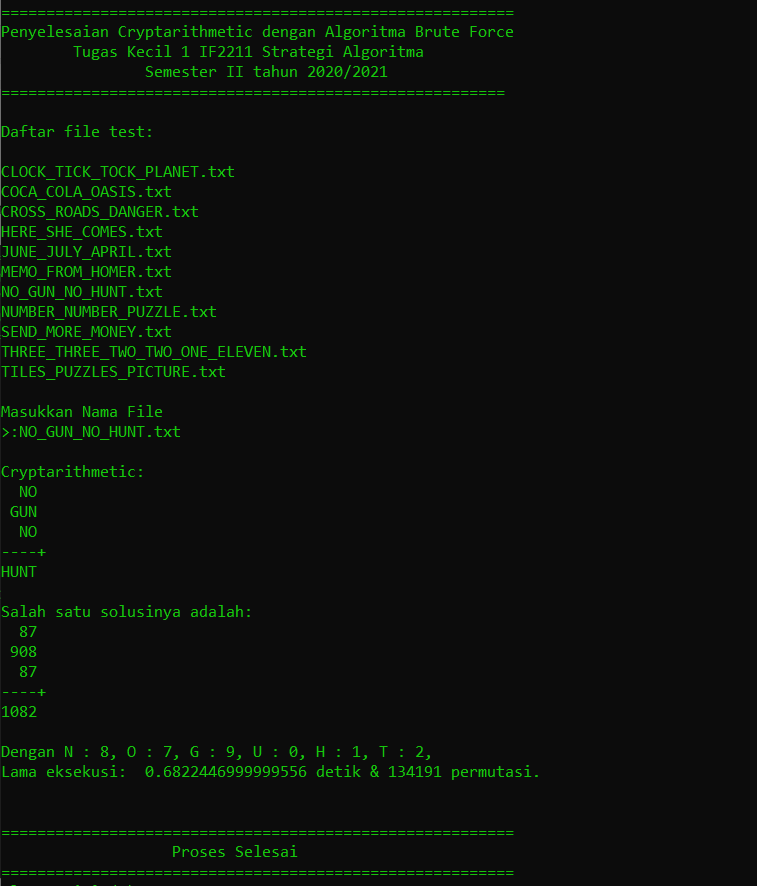
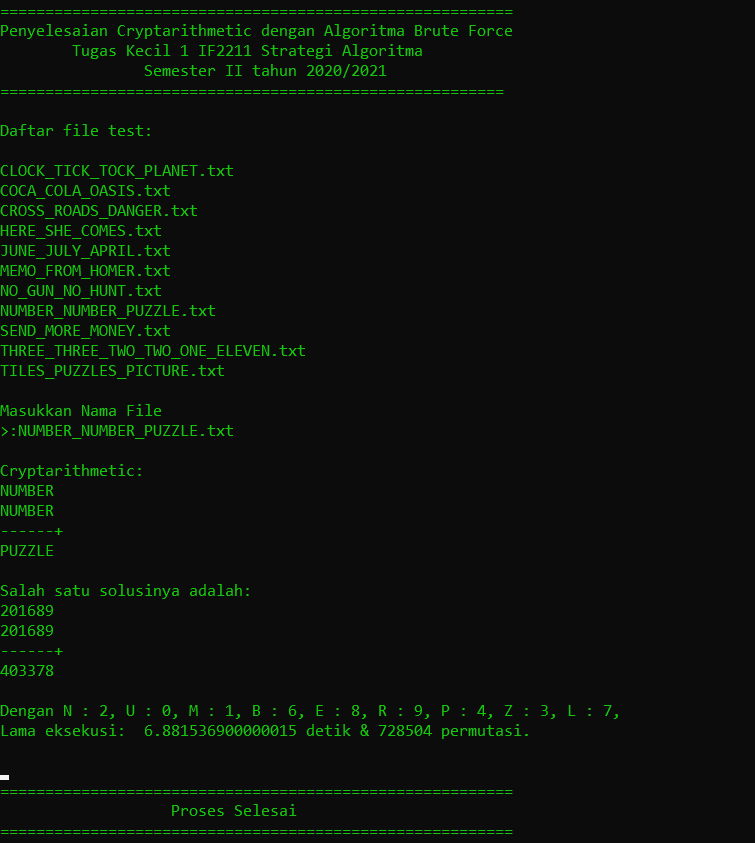
1. **CROSS\_ROADS\_DANGER.txt**

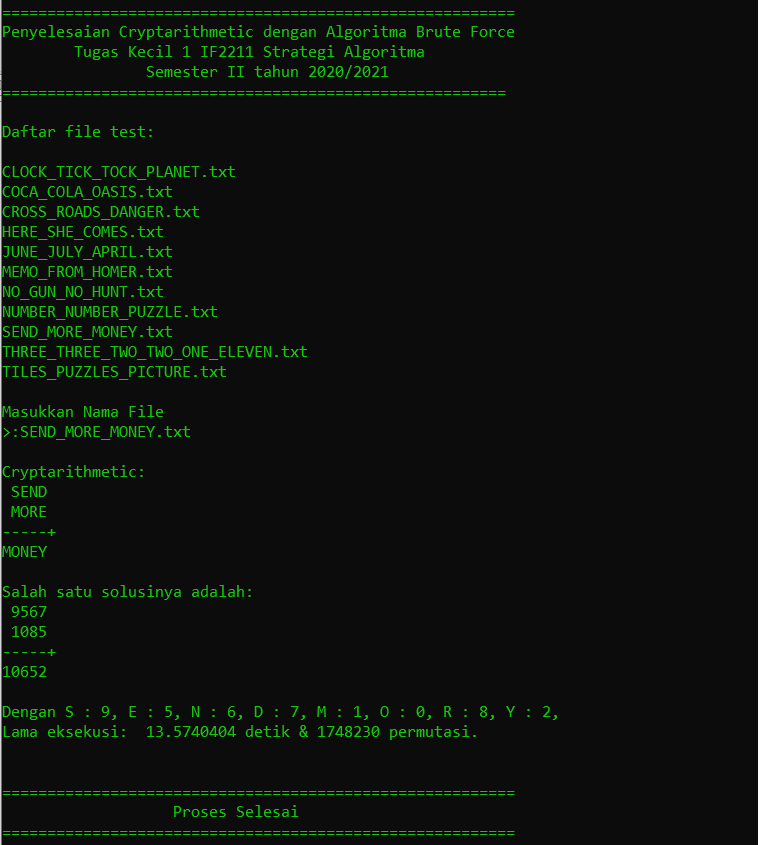


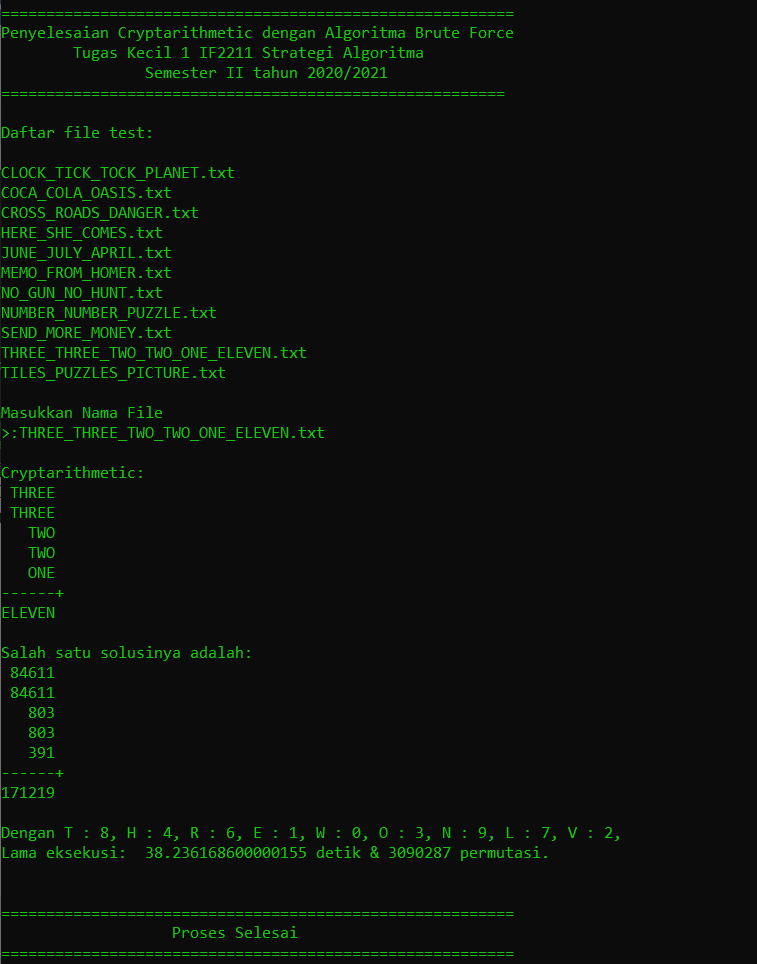
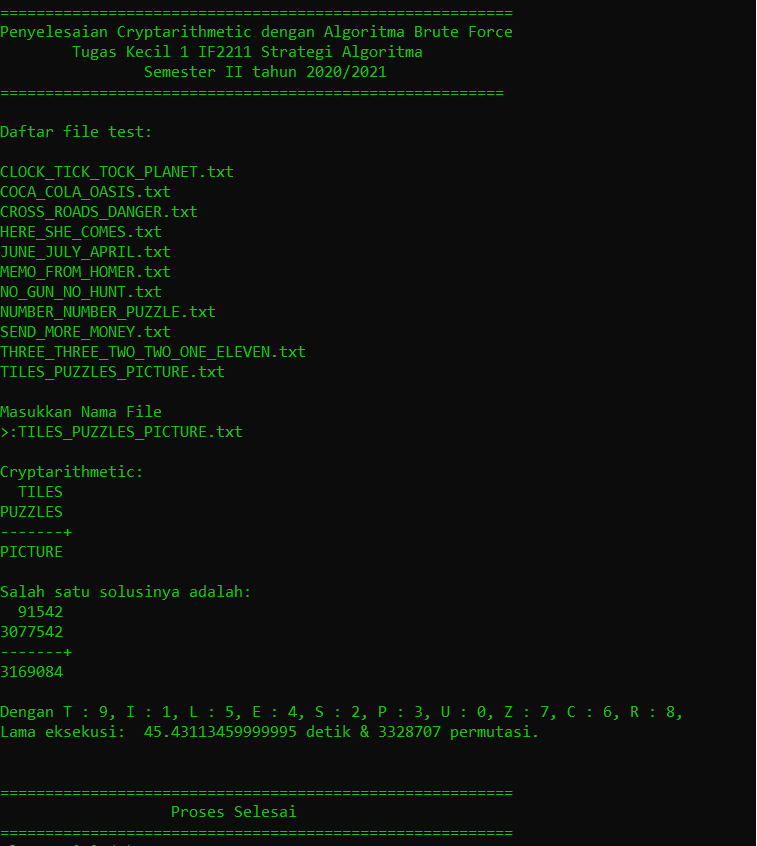
1. **HERE\_SHE\_COMES.txt** 
2. **JUNE\_JULY\_APRIL.txt** 





1. **MEMO\_FROM\_HOMER.txt** 
2. **NO\_GUN\_NO\_HUNT.txt** 
3. **NUMBER\_NUMBER\_PUZZLE.txt** 
4. **SEND\_MORE\_MONEY.txt**



1. **THREE\_THREE\_TWO\_TWO\_ONE\_ELEVEN.txt** 
2. **TILES\_PUZZLES\_PICTURE.txt** 

**CEK LIST PENILAIAN ASISTEN**

|  |  |  |
| --- | --- | --- |
| **Poin** | **Iya** | **Tidak** |
| 1. Program berhasil dikompilasi tanpa kesalahan (no syntax error) |  |  |
| 1. Program berhasil running |  |  |
| 1. Program dapat membaca file masukan dan menuliskan luaran. |  |  |
| 1. Solusi cryptarithmetic hanya benar untuk persoalan cryptarihtmetic dengan dua buah operand. |  |  |
| 1. Solusi cryptarithmetic benar untuk persoalan cryptarihtmetic untuk lebih dari dua buah operand. |  |  |