Tugas Kecil 1 IF2211 Strategi Algoritma

Penyelesaian Cryptarithmetic dengan Algoritma Brute Force

Nama: Rezda Abdullah Fachrezzi
NIM : 13519194

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

1. Deskripsi Singkat
   Cryptarithmetic (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:

```
   S E N D
+  M O R E
---------
  M O N E Y
```

   Solusinya adalah:

```
   9 5 6 7
+  1 0 8 5
---------
  1 0 6 5 2
```

   Jadi, S = 9, E = 5, N = 6, D = 7, M = 1, O = 0, R = 8, Y = 2

   Pada laporan ini akan dijelaskan cara kerja program sederhana untuk menyelesaikan persoalan cryptarithmetic.

2. Penjelasan Algoritma Brute Force yang digunakan.
   Algoritma yang digunakan menggunakan metode *brute force* dengan memanfaatkan permutasi (*exhaustive search*).
   Langkah-langkah program menyelesaikan permasalahan tersebut adalah sebagai berikut:
   1. Program meminta nama fail kepada pengguna.
   2. Program akan membaca fail yang di-input apabila tidak ada masalah.
   3. Program melakukan *parsing* terhadap isi fail, lalu menuliskan ke layar apabila terdapat format yang salah.
   4. Apabila tidak ada masalah dengan isi fail, isi fail tersebut akan diproses oleh program, contohnya adalah sebagai berikut:
      Apabila inputnya adalah

```
  SEND
 MORE +
MONEY
```

      Maka, setiap kata pada setiap baris akan diubah menjadi *list of letter* berukuran 10 seperti berikut

```
listletter = [ "S", "E", "N", "D", "M", "O", "R", "Y", "-", "-" ]
```

"-" merupakan simbol *blank* atau kosong, posisi tiap-tiap huruf pada list tersebut mewakilkan angka dari huruf tersebut, pada list tersebut berarti S = 0, E = 1, dan seterusnya.

5. Dari list tersebut, kemudian akan dibentuk permutasinya dan akan dimasukkan ke dalam set.
6. Dari set tadi, akan diloop sampai menemukan kombinasi yang tepat.
7. Apabila terdapat kombinasi yang tepat, program akan menuliskan kembali isi fail ke layar, solusi dari masukan, waktu yang dihabiskan untuk mencari kombinasi, dan banyaknya tes yang dilakukan.

3. Source Code Program.
   Program yang dibuat menggunakan bahasa pemrograman Python, program ini memanfaatkan Algoritma Brute Force dengan menggunakan permutasi. Terdapat 3 fail pada program ini, yaitu **main.py**, **lib_tugas.py**, dan **lib_tugas2.py**.

   a. Fail main.py
   Program utama untuk dijalankan.

```
'''
Rezda Abdullah Fachrezzi
13519194
K4
Strategi Algoritma - Tugas Kecil 1
21-01-2021

Algoritma: lib_tugas.py
'''
import lib_tugas

# sort merupakan opsi mengurutkan hasil permutasi
# opsi ini memengaruhi kecepatan pencarian solusi
# apabila True, maka kecepatan akan kurang lebih konstan dan jumlah tes selalu sama
# apabila False, kecepatan akan bergantung dari jumlah tes (acak)
# default:= True

sort = True
lib_tugas.start(sort)
```

   b. Fail lib_tugas.py
   Fail tempat algoritma diletakkan.

```
'''
Library time digunakan untuk menghasilkan waktu sekarang
lib_tugas2 tempat fungsi-fungsi bantuan
'''
from time import time
from lib_tugas2 import permutations, stringtonum, makeliststringnum, filterliststring,
filltoten, isfirstnotzero, sumallin, print_output

'''
DISCLAIMER:

Fail = File
```

```
Fail merupakan File dalam Bahasa Indonesia menurut KBBI
'''

def intro():
  '''
  Intro
  '''
  print("-------------------------------------------------------------------------")
  print("| Selamat datang di Tucil 1 Stima 2020/2021                             |")
  print("| Penerapan Algoritma Brute Force untuk menyelesaikan CryptArithmetic |")
  print("| Oleh: Rezda A.F | 13519194 | K4                                       |")
  print("-------------------------------------------------------------------------")
  print("\n")

def parse(file):
  '''
  Parsing isi fail input dan error handler
  Output berupa boolean, list of letter, dan list of lines

  Asumsi: Spasi pada fail tidak dihiraukan, dianggap selalu rata kanan setiap baris
  (asumsi format penulisan selalu benar tanpa mempedulikan posisi kata).
  '''

  line = file.readlines()
  line_total = len(line)

  if (line_total < 4):
    print("Format isi fail salah! (Minimal empat baris)")
    return False, [], []

  for i in range(line_total):
    line[i] = line[i].replace(" ", "").replace("\n","")

  if(line[line_total-2]!="------"):
    print("Format isi fail salah! (Baris kedua dari akhir harus memuat string ------)")
    return False, [], []

  if(line[line_total-3][len(line[line_total-3])-1]!="+"):
    print("Format isi fail salah! (Baris ketiga dari akhir harus memuat tanda tambah
(+) pada akhir kata)")
    return False, [], []

  # Menghapus tanda tambah
  line[line_total-3] = line[line_total-3][:-1]

  # Pengecekan karakter lain selain alfabet
  huruf_valid = True
  huruf = []
  for data in line:
    if(data == "------"): continue
    if(not data.isalpha()):
      huruf_valid = False
      break
    for letter in data:
      try:
        huruf.index(letter)
      except:
        huruf.append(letter)

  if(not huruf_valid):
```

```
    print("Format isi fail salah! (Terdapat karakter lain selain alfabet -->
"+data+")")
    return False, [], []

  if(len(huruf)>10):
    print("Format isi fail salah! (Jumlah alfabet unik lebih dari 10)")
    return False, [], []

  return True, huruf, line

def start(sort=True):
  '''
  Memulai program
  '''
  intro()

  nama_file = input("Masukkan nama fail dengan ekstensinya: ")
  err = False

  try:
    file = open(nama_file, "r")
  except:
    err = True

  # Mengecek apakah ada error pada pembacaan fail
  if (err):
    return print("Terjadi galat! (Fail tidak ditemukan atau lainnya)")

  print("Mohon tunggu..")

  # Waktu sebelum memulai brute force
  timebefore = time()

  # Parsing fail
  check = parse(file)
  if(not check[0]):
    return

  # Assign data yang diterima dari fungsi parse
  # line = list of lines (fitlered)
  # permutasi = set of permutations of tuple of letter
  # ^^^^^^^^^-> filltoten: mengisi list of letter sehingga berjumlah 10,
  #             permutations : menghasilkan permutasi dari list tadi dan diubah
bentuknya menjadi set of tuple,
  #             sorted: mengurutkan isi set

  line = filterliststring(check[2])
  permutasi = permutations(filltoten(check[1]))
  if sort:
    permutasi = sorted(permutasi)

  # assign jumlah tes yang akan dihitung, penanda hasil yg ditemukan, dan hasil
permutasi yang benar
  tes = 0
  ada = False
  outnum = []

  # proses pencarian dengan metode brute force
  for data in permutasi:
    listnum = makeliststringnum(line, data) # menghasilkan list of num (string)
```

```
    tes += 1
    if(isfirstnotzero(listnum)): # mengecek apakah dimulai dengan 0 atau tidak, jika
tidak maka lanjut
      penjumlah = sumallin(listnum)
      hasil = int(listnum[len(listnum)-1])
      if penjumlah == hasil: # mengecek hasil penjumlah dengan hasilnya, apabila ada
maka loop langsung dihentikan
        ada = True
        outnum = data
        break

  # membuka kembali fail untuk diambil bentuk penjumlahannya utk ditampilkan ke layar
  file =  open(nama_file, "r")
  unedited_lines = file.readlines()
  file.close()

  # penampilan hasil ke layar
  print("\nInput:")
  print_output(unedited_lines)
  print("\n\nOutput:")

  if(not ada):
    print("Tidak ditemukan hasil untuk input tersebut..")
    return
  else:
    print_output(unedited_lines, True, outnum)
    # menghitung selisih waktu
    deltatime = time()-timebefore
    print("\n\nWaktu yang dihabiskan: %.2f"%deltatime+" detik")
    print("Jumlah tes:", tes, "kali")
    return
```

c. Fail lib_tugas2.py

Fail tempat fungsi-fungsi yang membantu algoritma diletakkan.

```
'''
Berisi fungsi-fungsi yang membantu proses brute force
Bikin ini biar ga menuh2in fail satunya aja hehe
'''

def swap(arr, i, j):
    '''
    swap array
    '''
    arr[i], arr[j] = arr[j], arr[i]
    return arr

def permutations(arr):
    '''
    Mencari permutasi pada array dengan Algoritma Heap's
    Referensi:
https://en.wikipedia.org/wiki/Heap%27s_algorithm#Details_of_the_algorithm
    output berupa set (agar tidak ada duplikat dari hasil permutasi)
    '''
    l = len(arr)
    out = set()
    out.add(tuple(arr))
    c = []
    for i in range(l):
```

```
        c.append(0)
    i = 0
    while i < l:
        if(c[i]<i):
            if(i%2==0):
                swap(arr, 0, i)
            else:
                swap(arr, c[i], i)
            out.add(tuple(arr))
            c[i] += 1
            i = 0
        else:
            c[i] = 0
            i+=1
    return out

def stringtonum(string, listnum):
  '''
  Mengubah string input menjadi angka sesuai listnum (dalam bentuk integer dan string)
  Contoh listnum = ["b", "c"] yang berarti "b" bernilai 0 dan "c" bernilai 1.
  '''
  out = ""
  for letter in string:
    if(letter == "-"): continue
    out += str(listnum.index(letter))
  return int(out), out

def makeliststringnum(liststring, listnum):
  '''
  Membuat list of string yang berisi nilai dari masing-masing string pada liststring
  '''
  out = []
  for data in liststring:
    out.append(stringtonum(data, listnum)[1])
  return out

def filterliststring(listnya):
  '''
  Menghapus "------" dari list of string
  '''
  out = []
  for data in listnya:
    if data == "------": continue
    out.append(data)
  return out

def filltoten(listnya):
  '''
  Mengisi list sampai berjumlah 10 (diisi dengan simbol -, dianggap kosong)
  '''
  arr2 = listnya
  pjg = len(listnya)
  minus = 10-pjg
  for i in range(minus):
    arr2.append("-")
  return arr2

def isfirstnotzero(listnum):
  '''
  Mengecek letter pertama pada string dari listnum apakah merupakan 0 atau bukan
```

```
  '''
  for num in listnum:
    if int(num[0]) == 0:
      return False
  return True

def sumallin(listnum):
  '''
  Menjumlahkan seluruh penjumlah kecuali hasil (nilai terakhir) dari listnum
  '''
  out = 0
  for i in range(len(listnum)):
    if i < len(listnum)-1:
      out += int(listnum[i])
  return out

def print_output(lines, replace=False, listnum=[]):
  '''
  Menulis bentuk penjumlahan pada layar
  '''
  for i in range(len(lines)):
    if (replace):
      for letter in lines[i]:
        if (letter.isalpha()):
          lines[i] = lines[i].replace(letter, str(listnum.index(letter)))
    print(lines[i],end="")
```

4. Beberapa contoh dan tangkapan layar pada saat program dijalankan.
   a. Ketika program pertama kali dijalankan

```
--------------------------------------------------------------------------
| Selamat datang di Tucil 1 Stima 2020/2021                              |
| Penerapan Algoritma Brute Force untuk menyelesaikan CryptArithmetic    |
| Oleh: Rezda A.F | 13519194 | K4                                        |
--------------------------------------------------------------------------


Masukkan nama fail dengan ekstensinya: tes.txt
```

   b. Beberapa contoh persoalan (soal ditulis di bawah tulisan input)

1.

```
input:
 SEND
 MORE+
------
MONEY

output:
 9567
 1085+
------
10652

Waktu yang dihabiskan: 14.47 detik
Jumlah tes: 1186951 kali
```

2.

```
input:
 ONE
 ONE+
------
 TWO

output:
 467
 467+
------
 934

Waktu yang dihabiskan: 0.33 detik
Jumlah tes: 2415 kali
```

3.

```
input:
  NO
 GUN
  NO+
------
HUNT

output:
  87
 908
  87+
------
1082

Waktu yang dihabiskan: 1.94 detik
Jumlah tes: 145956 kali
```

4.

```
input:
 NUMBER
 NUMBER+
------
 PUZZLE

output:
 201689
 201689+
------
 403378

Waktu yang dihabiskan: 42.97 detik
Jumlah tes: 3089283 kali
```

5.

```
input:
 THREE
 THREE
   TWO
   TWO
   ONE+
------
ELEVEN

output:
 84611
 84611
   803
   803
   391+
------
171219

Waktu yang dihabiskan: 67.01 detik
Jumlah tes: 3344534 kali
```

6.

```
Input:
 TUCIL
 TUBES+
------
STRESS

Output:
 96870
 96541+
------
193411

Waktu yang dihabiskan: 39.85 detik
Jumlah tes: 2059461 kali
```

7.

```
input:
      TEN
      TEN
    SEVEN
 FOURTEEN+
------
 FORTYONE

output:
      482
      482
    78082
 19564882+
------
 19643928

Waktu yang dihabiskan: 54.94 detik
Jumlah tes: 2953313 kali
```

8.

```
Input:
  WATER
   FIRE
  EARTH
    AIR+
------
 NATURE

Output:
  97364
   8046
  67432
    704+
------
 173546

Waktu yang dihabiskan: 37.37 detik
Jumlah tes: 1631239 kali
```

5. Alamat unduh program.
   Program dapat diunduh melalui alamat berikut.
   https://github.com/raf555/cryptarithmetic-solver

6. Tabel Keberjalanan Program

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 *cryptarithmetic* hanya benar untuk persoalan *cryptarihtmetic* dengan dua buah *operand*.		✓
5. Solusi *cryptarithmetic* benar untuk persoalan *cryptarihtmetic* untuk lebih dari dua buah operand.	✓	

7. Kesimpulan
   Program dapat dijalankan tanpa ada kesalahan compile (syntax error / gagal compile), program juga dapat menyelesaikan persoalan cryptarithmetic lebih dari dua operan. Menurut pembuat, satu-satunya kelemahan pada program ini hanyalah program tidak mengecek posisi kata pada tiap baris; diasumsikan isi fail selalu benar, yaitu posisi tiap kata pada tiap baris rata kanan.