**Laporan Tugas Kecil 1 Strategi Algoritma IF2211**

**Penyelesaian *Cryptaritmethic* dengan Algoritma *Brute Force***



Disusun oleh:

Nama: Alvin WIlta

NIM: 13519163

**PROGRAM STUDI TEKNIK INFORMATIKA**

**SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA**

**INSTITUT TEKNOLOGI BANDUNG**

**2021**

1. Algoritma Brute Force

Algoritma brute force adalah sebuah algoritma yang simpel dan mudah untuk dibuat, akan tetapi sangat memakan waktu dan tenaga dalam pencarian solusinya. Umumnya algoritma ini berpusat pada pencarian solusi dengan mencoba satu per satu kemungkinan solusi yang ada dari permasalahan tersebut.

Algoritma untuk mencari solusi dari Cryptaritmethic Puzzle juga dilakukan dengan mencoba satu per satu solusi yang ada dengan menggunakan prinsip permutasi. Untuk algoritma dari pencarian solusi adalah:

1. Mencari permutasi dari list angka [0,1,2,3,4,5,6,7,8,9] secara berurutan sebanyak huruf unik dari soal
2. Memasangkan angka-angka yang sudah ditentukan dari list angka ke masing-masing huruf unik ke dalam dictionary (misal: {(“K”:”2”),(“L”:”5”),...} )
3. Memasangkan huruf unik tersebut sesuai dengan operand dan hasil yang diberi pada soal
4. Menghitung jumlah dari operand yang sudah berupa angka dan dibandingkan dengan hasil seharusnya. (jika sesuai maju ke tahap 5, jika tidak kembali ke tahap 1)
5. Mengecek apakah terdapat angka 0 pada angka digit terbesar/pertama dari masing-masing operand maupun hasil. (jika terdapat kembali ke tahap 1, jika tidak maju ke tahap 6)
6. Solusi disimpan

Sedangkan algortma program tersebut adalah:

1. Membaca semua file yang terdapat pada direktori “../test/” dan menjabarkan nama semua file yang terdapat pada direktori tersebut
2. Memberikan pilihan kepada user untuk memilih file mana yang ingin dicari solusi cryptaritmethic
3. Program membaca file dan mengubahnya menjadi bentuk list
4. List dibersihkan dari symbol-simbol yang tidak digunakan
5. List dicek jumlah hurufnya, jika melebihi 10 maka program akan menolak file tersebut (jika diterima maka lanjut, jika ditolak maka kembali ke tahap 1)
6. Program mencatat waktu sekarang dan mencatat berapa kali iterasi permutasi dilakukan
7. Program menjalankan prosedur mencari permutasi (sudah tertera di atas)
8. Program mengeluarkan output berupa soal yang diberi dan angka yang merupakan solusinya, serta mengeluarkan berapa kali permutasi dilakukan
9. User diberikan pilihan untuk melanjutkan program atau tidak (jika tidak maka program selesai, jika iya kembali ke tahap 1)
10. Source Code Program (Python)

import os

import time

def loadfile(): #memilih file mana yang akan diparse

    entries = os.listdir("../test")

    i = 0

    for entry in entries:

        print(str(i) + ". " + entry)

        i+=1

    idx = input("ketik nomor teks yang ingin diparse: ")

    namafile = entries[int(idx)]

    return namafile

def parser(namafile,lines): #membaca file txt dan mengubahnya menjadi list

    with open("../test/"+namafile,"r") as f:

        lines = f.read().splitlines()

    return lines

def strip(lines): #membersihkan list dari karakter yang tidak digunakan

    del lines[len(lines)-2]

    i=0

    for line in lines:

        line = line.replace("+","")

        line = line.replace("-","")

        line = line.replace(" ","")

        lines[i] = line

        i += 1

    return lines

def cek\_string(list\_string): #mengecek apakah terdapat string >10 karakter

    cek = True

    for string in list\_string:

        if len(string) > 10:

            cek = False

            break

    return cek

def ulang\_program():

    x = input("Apakah anda masih inggin menggunakan program? (0/1)")

    if x==str(0):

        quit()

    elif x==str(1):

        main()

    else:

        print("Tolong masukkan dengan benar, 0 untuk tidak dan 1 untuk iya.")

        ulang\_program()

def permutasi(xlist,r):

    ylist = tuple(xlist)

    n = len(ylist)

    indeks = list(range(n))

    siklus = list(range(n, n-r, -1))

    yield tuple(ylist[i] for i in indeks[:r])

    while n:

        for i in reversed(range(r)):

            siklus[i] -= 1

            if siklus[i] == 0:

                indeks[i:] = indeks[i+1:] + indeks[i:i+1]

                siklus[i] = n - i

            else:

                j = siklus[i]

                indeks[i], indeks[-j] = indeks[-j], indeks[i]

                yield tuple(ylist[i] for i in indeks[:r])

                break

        else:

            return

def unique\_is(xlist): #mengeluarkan unique i pada string di dalam list

    unique = []

    for string in xlist:

        for i in range(len(string)):

            if (string[i] not in unique):

                unique.append(string[i])

    return unique

def jml\_list(xlist):

    jml = 0

    i=0

    for i in range(len(xlist)-1):

        jml += int(xlist[i])

    return jml

def main():

    tebakan=[1,0,2,3,4,5,6,7,8,9]

    namafile = str("")

    lines = []

    namafile = loadfile()

    lines = parser(namafile,lines)

    lines = strip(lines) # stripped list

    if not cek\_string(lines):

        print('Soal terkait memiliki operand yang melebihi 10 karakter dan tidak akan diproses.\n')

        ulang\_program()

    else:

        start\_time = time.time()

        tries = 0

        unique = unique\_is(lines)

        for perm in permutasi(tebakan,len(unique)):

            kamus = dict(zip(unique,perm))

            operasi = []

            tdknol = True #melakukan pengecualian pada 0 di depan

            for string in lines:

                operand = []

                for i in string:

                    operand.append(str(kamus[i]))

                temp = "".join(operand)

                operasi.append(temp)

            #print(operasi)

            jumlah = jml\_list(operasi)

            tries += 1

            if ((jumlah == int(operasi[len(operasi)-1]))): #mengecek apakah ada 0 di depan

                for i in range(len(operasi)):

                    if (int(operasi[i][0]) == 0):

                        tdknol = False

                        break

                if tdknol:

                    for j in range(len(operasi)):

                        if (j == len(operasi)-1):

                            print("------------")

                        print(lines[j]," : ",end="")

                        print(operasi[j])

                    break

        print("Jumlah Kasus yang diuji : ",tries)

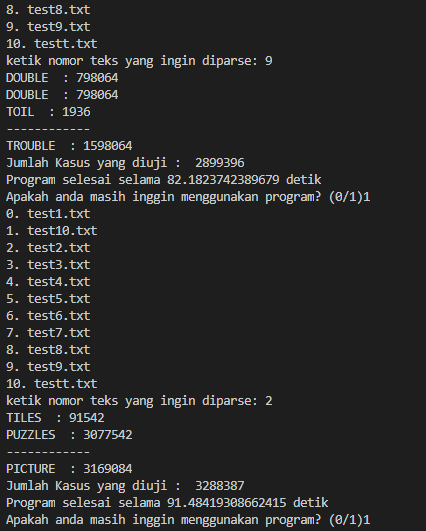
        print("Program selesai selama %s detik" % (time.time() - start\_time))

        ulang\_program()

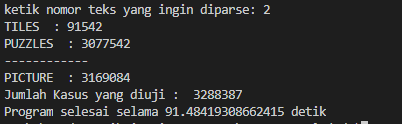
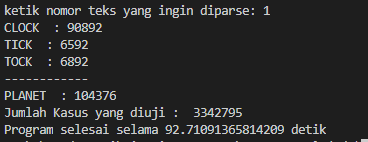
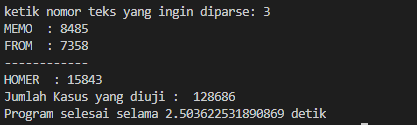
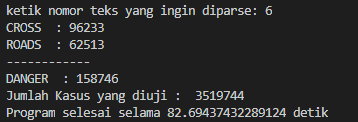
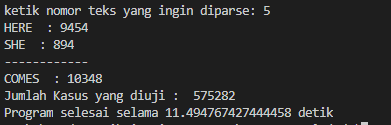
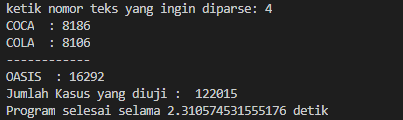
main()

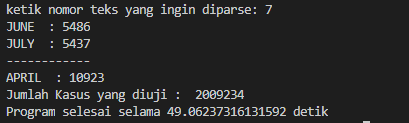
1. Screenshot Input & Output

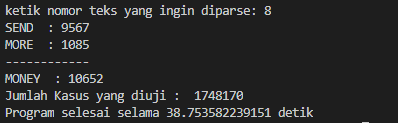
Tampilan penuh program:

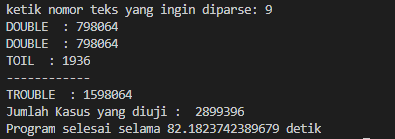


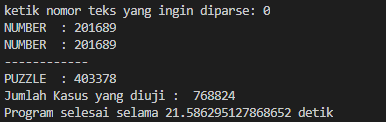
Output program:









1. Tabel Penilaian

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

1. Alamat Drive Kode Program

https://github.com/alvinwilta/IF2211\_Tucil1