LAPORAN TUGAS KECIL 1 IF2211 STRATEGI ALGORITMA
SEMESTER GENAP – 2020/2021
13519028 – HAFID ABI DANISWARA

1. Penjelasan algoritma brute force

pada tugas kecil 1 : cryptarithmetic, saya menggunakan bahasa pemrograman ptython dan penerapan algoritma brute force pada cryptarithmetic adalah sebagai berikut:

a. Tahap satu :

setelah membaca data dari file .txt, kemudian setelah dipisah menjadi array, array di masukkan ke sebuah fungsi yang bekerja untuk menghasilkan huruf-huruf unik apa saja yang terdapat pada string tersebut. Semisal pada array [“SEND”,”MORE”,”MONEY”] maka akan menghasilkan array [“S”, ”E”, ”N”, ”D”, ”M”, ”O”, ”R”, “Y”].

namun untuk mendukung pendekatan heuristic yang akan dilakukan pada tahap berikutnya, maka pengurutan array indeks ke-0 adalah huruf pertama dari baris pertama dan indeks ke-1 adalah huruf pertama dari baris terakhir (hasil), namun apabila baris pertama dan terkahir memiliki awalan huruf yang sama, maka indeks ke-1 akan diisi oleh huruf pertama dari kata baris ke 2/3/4/dst yang memiliki huruf yang berbeda dari indeks ke-0 dan kemudian di swap (swapping indeks ke 0 dan 1 dilakukan khusus untuk kasus apabila huruf pertama di baris pertama sama dengan huruf pertama di baris terakhir).

b. Tahap kedua:

Melakukan permutasi dari array 10 elemen berisi [0,1,2,3,4,5,6,7,8,9] menjadi permutasi array n elemen, dimana n adalah jumlah huruf yang dipakai pada file txt / length elemen dari array yang dihasilkan pada tahap satu. Hasil permutasi akan dikembalikan pada list of list ([[permutasi 1], [permutasi 2], … ,[permutasi ke-n]]).

Namun untuk mengoptimalkan bruteforce, dilakukan pendekatan heuristic dengan cara tidak mengembalikan permutasi yang elemen di indeks ke-0 adalah 0 dan indeks ke-1 adalah 0, karena sesuai spek huruf pertama tidak boleh merepresentasikan angka nol. Kemudian pendekatan heuristic lain yang dilakukan adalah mendeteksi apakah pada bagian baris terakhir (hasil penjumlahan), panjang stringnya lebih panjang dari baris-baris lainya (kecuali baris “------”) atau dalam istilah lain ada ‘tabungan’. Apabila terdapat maka permutasi dengan indeks ke-1 (indeks ke-1 adalah huruf pertama dari baris terakhir / hasil) dengan angka lebih dari jumlah baris yang elemenya adalah satu kurangnya dari elemen baris terakhir. Sebagai contoh: SEND+MORE = MONEY, maka apabila permutasi dengan M > 1 akan terkena seleksi alam dan tidak dikembalikan.

Dasar penggunaan heuristic tersebut adalah, apabila misal kita punya SEND+MORE=MONEY, maka M tidak mungkin lebih dari satu karena semisal kita taruh 9546+8735 < 20.000, dan misal kita punya AAAA+AAAA=BCCCD dan A = 9, maka 9999+9999<20.000. Untuk kasus lain seperti AAAA+AAAA+AAAA+AAAA=BCCCD, maka B<5, karena misal A=9, 9999+9999+9999+9999<50.000, sehingga dari hal ini bisa kita manfaatkan untuk mempersempit ruang percobaan untuk memberi waktu yang lebih optimal dalam percobaan brute force.

c. *Tahap ketiga:*
Dengan memanfaatkan permutasi yang diperoleh pada tahap kedua, kita dapat memasangkan list hasil permutasi tersebut dengan mengubah string-string di txt dengan angka-angka di masing-masing permutasi, kemudian di cocokan antara hasil dan penjumlahan, apabila sama maka permutasi tersebut benar maka tampilkan hasil ke layar. Kemudian proses ini diulangi terus hingga semua permutasi telah dicoba.

2. Source code program

a. Main program

```
path = "..\\test\\"
fileteks = str(input("masukkan nama file test (including dot txt) = "))
inputan = str((open(path+fileteks,'r')).read()).split()
h = getHuruf(inputan)
print("banyak komponen huruf = "+str(len(h))+" \n")
starttime= dt.datetime.today()
bruteforce(inputan,starttime)
endtime = dt.datetime.today()
print("\nFINISH ALL IN = "+str((endtime-starttime).total_seconds())+" SECOND")
input()
```

b. Fungsi bruteforce (untuk melakukan bruteforce)

```
def bruteforce(data,starttime):
    menabung = cekTabunganAngka(data)
    listhuruf = getHuruf(data)
    pasangan_angka = getPermutasiAnkga(len(listhuruf),menabung)
    test = 0
    reslt = 0

    if len(listhuruf)<9:
        for variasi in pasangan_angka:
            test = test + 1
            hasil_cripta = criptaarimatic(data, listhuruf,  variasi)
            # print(hasil_cripta)
            if hasil_cripta['status']:
                reslt = reslt + 1
                printSolution(data, hasil_cripta['data'], reslt)
                print("percobaan ke = " + str(test)+"/"+str(len(pasangan_angka)))
                waktu = (dt.datetime.today() - starttime).total_seconds()
                print("waktu dibutuhkan = " + str(waktu) + " detik")
                print()
    else:
        for v in range(len(pasangan_angka)-1,-1,-1):
            test = test+1
            hasil_cripta = criptaarimatic(data, listhuruf, pasangan_angka[v])
            if hasil_cripta['status']:
                reslt = reslt + 1
                printSolution(data, hasil_cripta['data'], reslt)
                print("percobaan ke = " + str(test)+ "/"+str(len(pasangan_angka))
                waktu = (dt.datetime.today() - starttime).total_seconds()
                print("waktu dibutuhkan = " + str(waktu) + " detik")
                print()
```

c. Fungsi printSolution (untuk output solusi ke layar dengan arr0 adalah soal, arr1 jawaban, dan index adalah solusi ke berapa)

```
def printSolution(arr0,arr1,index):
    print("Solusi ke-"+str(index))
    count =0
    longest_length = len(arr0[len(arr0)-1])+1
    while(arr0[count]!="------"):
        spacing = longest_length-len(arr0[count])
        for x in range(spacing): print(' ',end='')
        print(arr0[count],end='')
        print('   ',end='')
        spacing = spacing+(len(arr0[count])-len(str(arr1[count])))
        for x in range(spacing): print(' ', end='')
```

```
            print(arr1[count])
            count=count+1

        print("+",end='')
        for x in range(longest_length-1):print("-",end='')
        print('   +',end='')
        for x in range(longest_length - 1): print("-", end='')
        print()
        print(" "+arr0[len(arr0)-1]+"    "+str(arr1[len(arr1)-1]))
```

d. Fungsi cekTabunganAngka (mengecek apakah memiliki tabungan angka atau tidak)

```
def cekTabunganAngka(data):
    n = len(data[len(data)-1])
    la = len(data)
    c = 0
    for x in data:
        if x!="------" and x!=data[len(data)-1] and len(x)+1==n:
            c=c+1
        elif(x!="------" and x!=data[len(data)-1] and len(x)>=n):
            #print(x)
            return 9

    if c>1:
        return c-1
    else:
        return 2
```

e. Fungsi criptaatimatic (mengubah huruf menjadi angka dan mencocokan apakah benar atau tidak sesuai dengan kaidah cryptarithmetic)

```
def criptaarimatic(arr,listhuruf,listangka):
    angka = []
    flag = True
    count = 0

    while (count < len(arr) and flag):
        if arr[count] != "------":
            temp = ""
            for y in arr[count]:
                temp = temp + str(pencocokan(listhuruf, listangka, y))

            if temp[0] == "0": flag = False
            angka.append(int(temp))

        count = count + 1

    count = 0
    if flag:
        for x in range(len(angka) - 1):
            count = count + angka[x]

        if count == angka[len(angka) - 1]:
            return {
                'status': True,
                'data': angka
            }
        else:
            return {
                'status': False,
                'data': angka
            }
    else:
        return {
            'status':False,
            'data' : []
        }
```

f. Fungsi pencocokan (mencocokan list huruf dengan list angka, dan mengembalikan angka yang bersesuaian dengan huruf berkaitan.)

```
def pencocokan(arrHuruf,arrAngka,huruf):
    if len(arrAngka)==len(arrHuruf):
        flag = False
        c = 0
        while c<len(arrHuruf) and flag==False:
            if arrHuruf[c] == huruf:
                flag=True
            else:
                c=c+1

        if flag:
            return arrAngka[c]
        else:
            return -1
    else:
        return -1
```

g. Fungsi getPermutasiAngka (mengembalikan array berisikan angka yang telah dipermutasi. P(10,n))

```
def getPermutasiAnkga(n,tabungan = 9):
    arr = [x for x in range(10)]
    if n<10:
        a1 = pisahkan(kombinatorial(arr,n,0,[0 for x in range(n)]),n)
        a2 = []
        for z in a1:
            a2 = a2 + permutationEngine(z,len(z),tabungan)

        return pisahkan(a2,n)
    elif n==10:
        return pisahkan(permutationEngine(arr,10,tabungan),10)
    else:
        return []
```

h. Fungsi kombinatorial (untuk melakukan kombinasi, jadi untuk melakukan P(10,n) dimana n<10 maka dilakukan kombinasi baru dilakukan permutasi)

```
def kombinatorial(arr,n,pos0,res):
    if(n==0):
        return res
    else:
        reslt = []
        for x in range(pos0,len(arr)-n+1):
            res[len(res)-n] = arr[x]
            reslt = reslt + kombinatorial(arr,n-1,x+1,res)

        return reslt
```

i. Fungsi permutationEngnine (fungsi yang bertanggung jawab untuk menghasilkan permutasi dari suatu array. Fungsi ini menghasilkan permutasi list sebanyak P(n,n)).

```
def permutationEngine(arr,arrSize,tabungan=9):
    if(arrSize==1):
        if str(arr[0])!='0' and str(arr[1])!='0':
            if int(arr[1])>tabungan:
                return []
            else:
                return arr
        else:
            return []
    else:
        res = []
        for y in range(arrSize):
            res = res+permutationEngine(arr,arrSize-1,tabungan)
            if(arrSize%2==1):
                arr[0],arr[arrSize-1] = arr[arrSize-1],arr[0]
            else:
                arr[y],arr[arrSize-1] = arr[arrSize-1],arr[y]

        return res
```

j. Fungsi getHuruf

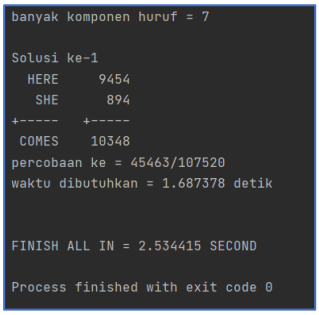
```
def getHuruf(arrofinput):
    r = list()
    r.append(arrofinput[0][0])
    temp = arrofinput[len(arrofinput)-1][0]
    if (not(temp in r)):
        r.append(temp)
    else:
        t=1
        while arrofinput[t][0] in r and arrofinput[t]!="------":
            t =t +1
        r.append(arrofinput[t][0])
        r[0],r[1] = r[1],r[0]

    for x in arrofinput:
        if (x!="------"):
            for y in x:
                if (not(y in r)):
                    r.append(y)

    return r
```

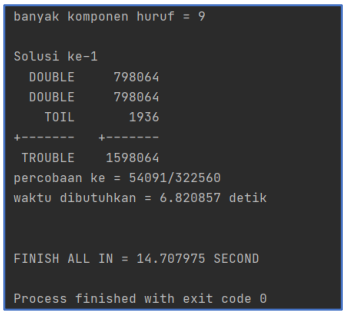
k. Fungsi pisahkan

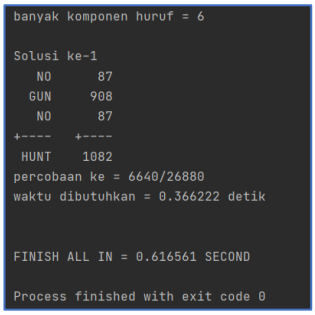
```
def pisahkan(arr,n):
    res = []
    temp = []
    count = 0
    for x in arr:
        if count<n:
            temp.append(x)
            count=count+1
        else:
            res.append(temp)
            temp=[]
            temp.append(x)
            count = 1

    res.append(temp)
    return res
```

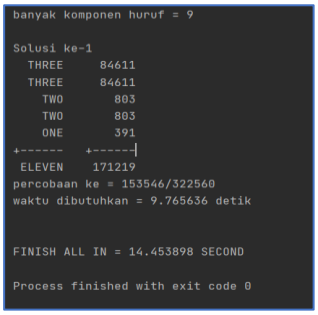
3. Testing program

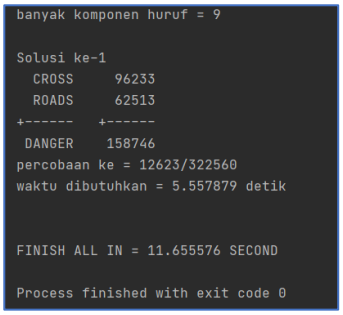
Berikut adalah dokumentasi percobaan program bruteforce untuk kasus cryptarithmetic:

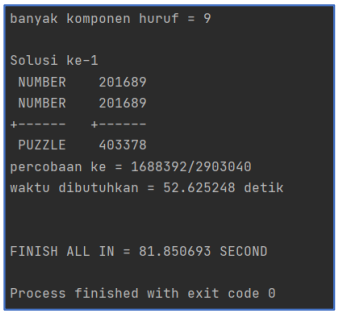


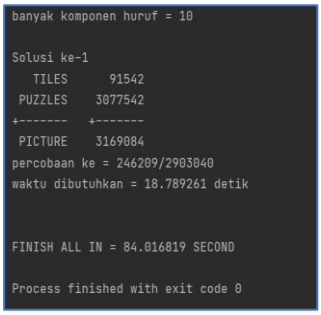


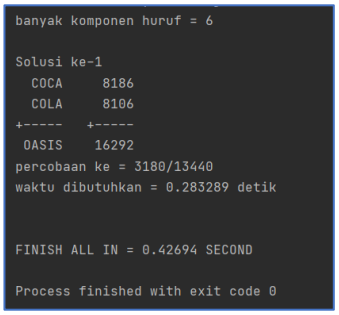












4. Kode program

Untuk source code dapat diakses di: https://github.com/hafidabid/tucil1stima

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.	✓	