

1. Penjelasan algoritma brute force

pada tugas kecil 1 : cryptarithmic, saya menggunakan bahasa pemrograman python dan penerapan algoritma brute force pada cryptarithmic 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 terakhir 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 lainnya (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 elemennya 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 memasang list hasil permutasi tersebut dengan mengubah string-string di txt dengan angka-angka di masing-masing permutasi, kemudian di cocokkan 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_crypta = cryptaarimatic(data, listhuruf, variasi)
            # print(hasil_crypta)
            if hasil_crypta['status']:
                reslt = reslt + 1
                printSolution(data, hasil_crypta['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_crypta = cryptaarimatic(data, listhuruf, pasangan_angka[v])
                    if hasil_crypta['status']:
                        reslt = reslt + 1
                        printSolution(data, hasil_crypta['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(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 criptaarimatic (mengubah huruf menjadi angka dan mencocokkan apakah benar atau tidak sesuai dengan kaidah cryptarithmic)

```

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 (mencocokkan 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 `permutationEngine` (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 cryptarithmic:

| | | |
|--|--|--|
| <p>banyak komponen huruf = 7</p> <p>Solusi ke-1</p> <pre> HERE 9454 SHE 894 +-----+----- COMES 10348 percobaan ke = 45463/107520 waktu dibutuhkan = 1.687378 detik FINISH ALL IN = 2.534415 SECOND Process finished with exit code 0 </pre> | <p>banyak komponen huruf = 9</p> <p>Solusi ke-1</p> <pre> DOUBLE 798064 DOUBLE 798064 TOIL 1936 +-----+----- TROUBLE 1598064 percobaan ke = 54091/322560 waktu dibutuhkan = 6.820857 detik FINISH ALL IN = 14.707975 SECOND Process finished with exit code 0 </pre> | <p>banyak komponen huruf = 6</p> <p>Solusi ke-1</p> <pre> NO 87 GUN 908 NO 87 +-----+----- HUNT 1082 percobaan ke = 6640/26880 waktu dibutuhkan = 0.366222 detik FINISH ALL IN = 0.616561 SECOND Process finished with exit code 0 </pre> |
| <p>banyak komponen huruf = 9</p> <p>Solusi ke-1</p> <pre> THREE 84611 THREE 84611 TWO 803 TWO 803 ONE 391 +-----+----- ELEVEN 171219 percobaan ke = 153546/322560 waktu dibutuhkan = 9.765636 detik FINISH ALL IN = 14.453898 SECOND Process finished with exit code 0 </pre> | <p>banyak komponen huruf = 9</p> <p>Solusi ke-1</p> <pre> CROSS 96233 ROADS 62513 +-----+----- DANGER 158746 percobaan ke = 12623/322560 waktu dibutuhkan = 5.557879 detik FINISH ALL IN = 11.655576 SECOND Process finished with exit code 0 </pre> | <p>banyak komponen huruf = 9</p> <p>Solusi ke-1</p> <pre> NUMBER 201689 NUMBER 201689 +-----+----- PUZZLE 403378 percobaan ke = 1688392/2903040 waktu dibutuhkan = 52.625248 detik FINISH ALL IN = 81.850693 SECOND Process finished with exit code 0 </pre> |
| <p>banyak komponen huruf = 10</p> <p>Solusi ke-1</p> <pre> TILES 91542 PUZZLES 3077542 +-----+----- PICTURE 3169084 percobaan ke = 246209/2903040 waktu dibutuhkan = 18.789261 detik FINISH ALL IN = 84.016819 SECOND Process finished with exit code 0 </pre> | <p>banyak komponen huruf = 10</p> <p>Solusi ke-1</p> <pre> CLOCK 90892 TICK 6592 TOCK 6892 +-----+----- PLANET 104376 percobaan ke = 104998/645120 waktu dibutuhkan = 8.630775 detik FINISH ALL IN = 21.004809 SECOND Process finished with exit code 0 </pre> | <p>banyak komponen huruf = 6</p> <p>Solusi ke-1</p> <pre> COCA 8186 COLA 8106 +-----+----- OASIS 16292 percobaan ke = 3180/13440 waktu dibutuhkan = 0.283289 detik FINISH ALL IN = 0.42694 SECOND Process finished with exit code 0 </pre> |

4. Kode program

Untuk source code dapat diakses di:

https://drive.google.com/drive/folders/1iXTzuxq6TYTByPwq_5pqiRXW5R3VJ418?usp=sharing

| Poin | Ya | Tidak |
|--|-------------------------------------|-------------------------------------|
| 1. Program berhasil dikompilasi tanpa kesalahan (no syntax error) | <input checked="" type="checkbox"/> | <input type="checkbox"/> |
| 2. Program berhasil <i>running</i> | <input checked="" type="checkbox"/> | <input type="checkbox"/> |
| 3. Program dapat membaca file masukan dan menuliskan luaran. | <input checked="" type="checkbox"/> | <input type="checkbox"/> |
| 4. Solusi <i>cryptarithmic</i> hanya benar untuk persoalan <i>cryptarihtmetic</i> dengan dua buah <i>operand</i> . | <input type="checkbox"/> | <input checked="" type="checkbox"/> |
| 5. Solusi <i>cryptarithmic</i> benar untuk persoalan <i>cryptarihtmetic</i> untuk lebih dari dua buah <i>operand</i> . | <input checked="" type="checkbox"/> | <input type="checkbox"/> |