**LAPORAN TUGAS KECIL 1**

**PENYELESAIAN PUZZLE CRYPTARITHMETICS DENGAN ALGORITMA BRUTE FORCE**

**IF2211 STRATEGI ALGORITMA**



Oleh:

**Hizkia Raditya Pratama Roosadi K02/13519087**

**PROGRAM STUDI TEKNIK INFORMATIKA**

**INSTITUT TEKNOLOGI BANDUNG**

**2021**

1. **Definisi Masalah**

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.

A picture containing table

Description automatically generated

Gambar 1. Contoh Penyelesaian Cryptarithmetic

Pada tugas ini, sebuah algoritma untuk menyelesaikan permasalahan cryptarithmetic harus dibuat oleh mahasiswa IF 2019. Algoritma yang dibuat harus menggunakan prinsip *brute force* yang dapat diartikan sebagai mencoba setiap kemungkinan hingga didapat hasil yang sesuai.

1. **Algoritma *Brute Force***

Menurut Munir (2021), algoritma *brute force* merupakan pendekatan penyelesaian masalah yang lempang. Berdasar definisinya (*brute force*) algoritma ini lebih mementingkan kekuatan untuk menyelesaikan masalah dibanding efisiensi atau cara yang lainnya. Algoritma *brute force* dapat digunakan dalam beberapa kasus, diantaranya: mencari elemen maksimum dan minimum dalam suatu *list*, mengalikan 2 buah matriks, mengurutkan suatu *list* bilangan, dsb. Pada tugas ini, prinsip tersebut akan digunakan untuk menyelesaikan puzzle cryptarithmetic.

Pada tugas ini, penulis memutuskan untuk menerapkan konsep bruteforce pada program dengan bahasa Python sesuai Langkah-langkah berikut:

1. Ambil seluruh kata dari file input dan masukan ke dalam array
2. Ambil seluruh huruf unik dan masukan kedalam array yang berbeda
3. Pisahkan operan dengan hasil
4. *Assign* suatu variabel, missal x, dengan angka 10n-1 dengan n adalah jumlah elemen pada array berisi alfabet unik
5. Tambahkan x dengan 1 sampai didapatkan suatu angka dengan kriteria berikut, angka tersebut unik (seluruh angka pada tiap digitnya berbeda) dan memiliki jumlah digit yang sama dengan jumlah elemen pada array berisi alfabet unik
6. Gunakan tipe bentukan bawaan dari Python yaitu dict, sebuah tipe bentukan yang dapat meng-*assign* suatu key dengan suatu value, misal key adalah huruf P pada array alfabet dan value untuk key tersebut adalah 1
7. Karena digit angka dan jumlah elemen memiliki jumlah yang sama, akan didapatkan pasangan key:value yang menandakan value untuk semua huruf
8. Ubah seluruh operan menjadi angka sesuai pasangan key:value
9. Ubah hasil menjadi angka sesuai pasangan key:value
10. Jumlahkan seluruh operan
11. Jika jumlah seluruh operan sama dengan nilai ubah menjadi angka dari hasil, maka salah satu solusi ditemukan
12. Jika tidak, iterasi akan dilanjutkan dengan menambah 1 ke x
13. Lakukan iterasi sampai seluruh kemungkinan dicapai

Dengan menambah satu ke suatu angka, akan selalu didapatkan kemungkinan pasangan key:value untuk tiap alfabet unik dalam array. Secara intuitif, misal untuk kata “PIE”, dengan asumsi array huruf unik memiliki 3 elemen yaitu [P,I,E], program akan memulai dari 100, 101, dan 102 (angka unik dan jumlah digit=jumlah elemen pertama), melakukan pengecekan syarat, lalu ditambah satu menjadi 103, dan seterusnya. Dengan metode seperti ini, semua permutasi subset dari [0,1,2..9] dengan jumlah elemen sama dengan jumlah elemen alfabet unik akan dicapai (walaupun tidak mangkus dan memerlukan waktu yang lama).

1. ***Source code***

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

Gambar 2. *Source Code* Program dalam Bahasa Python

1. **Hasil Input & Output Program**
   1. **Test 1**

**Graphical user interface, text, application

Description automatically generated**

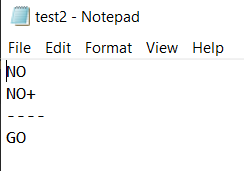
Gambar 3. Input Test 1

**Graphical user interface, text, application, email

Description automatically generated**

Gambar 4. Hasil dari Test 1

* 1. **Test 2**

****

Gambar 5. Input Test 2

**Graphical user interface, text, application

Description automatically generated**

Gambar 6. Hasil dari Test 2

* 1. **Test 3**

**Graphical user interface, text, application

Description automatically generated**

Gambar 7. Input Test 3

**Graphical user interface, text, application, email

Description automatically generated**

Gambar 8. Hasil dari Test 3

* 1. **Test 4**

**Graphical user interface, text, application

Description automatically generated**

Gambar 9. Input Test 4

Graphical user interface, text, application

Description automatically generated

Gambar 10. Hasil Test 4

* 1. **Test 5**

**Graphical user interface, text, application

Description automatically generated**

Gambar 11. Input Test 5

**Graphical user interface, text, application, email

Description automatically generated**

Gambar 12. Hasil dari Test 5

* 1. **Test 6**

**Graphical user interface, text, application

Description automatically generated**

Gambar 13. Input Test 6

**Graphical user interface, text, application

Description automatically generated**

Gambar 14. Hasil dari Test 6

* 1. **Test 7**

**Graphical user interface, text, application

Description automatically generated**

Gambar 15. Input Test 7

**Graphical user interface, text, application

Description automatically generated**

Gambar 16. Hasil dari Test 7

* 1. **Test 8**

**Graphical user interface, text, application

Description automatically generated**

Gambar 17. Input Test 8

Graphical user interface, text

Description automatically generated with medium confidence

Gambar 18. Hasil dari Test 8

1. **Alamat Drive**

Untuk program utuh dalam bentuk ZIP, ada di link berikut.

Kelas K2: <https://drive.google.com/drive/u/0/folders/1y7u0LIIha5tV8R6sr9vOnMFQqVzRVUf2>

1. **Tabel *Checklist***

|  |  |  |
| --- | --- | --- |
| **Poin** | **Ya** | **Tidak** |
| 1. Program berhasil dikompilasi tanpa  kesalahan (no syntax error) | **V** |  |
| 2. Program berhasil running | **V** |  |
| 3. Program dapat membaca file  masukan dan menuliskan luaran. | **V** |  |
| 4. Solusi cryptarithmetic hanya benar  untuk persoalan cryptarihtmetic  dengan dua buah operand. |  | **V** |
| 5. Solusi cryptarithmetic benar untuk  persoalan cryptarihtmetic untuk  lebih dari dua buah operand. | **V** |  |