LAPORAN TUGAS KECIL I IF2211 STRATEGI ALGORITMA

SEMESTER II TAHUN 2020/2021

PENYELESAIAN *CRYPTARITHMETIC* DENGAN ALGORITMA *BRUTE FORCE*

Logo

Description automatically generated

OLEH

RAYHAN ALGHIFARI FAUZTA (13519039)

PROGRAM STUDI TEKNIK INFORMATIKA

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

2021

1. **Algoritma Brute Force**

Tugas kecil ini meminta mahasiswa untuk memecahkan permasalahan *cryptarithmetic* dengan membuat program komputer yang memanfaatkan algoritma *brute force*. Pada tugas ini, program yang saya buat menggunakan bahasa pemrograman Python 3 dikarenakan lebih familiar dibanding pilihan bahasa lainnya (C/C++/Java).

Alur program adalah sebagai berikut:

1. Program dimulai dengan meminta masukan file test case yang akan diuji dari pengguna.
2. Mulai menghitung waktu dan mencetak problem yang dibaca dari test case.
3. Inisialisasi parameter-parameter yang dibutuhkan untuk fungsi solve().
4. Mulai mencari solusi dengan fungsi solve() yang juga memanfaatkan fungsi permutations. Kedua fungsi ini akan dijelaskan lebih lanjut.
5. Ketika solusi ditemukan, program mencetak solusi dengan format sesuai dengan test case. Langkah ini dilakukan di dalam fungsi solve().
6. Terakhir, program akan mencetak waktu eksekusi program dihitung sejak selesai meminta input pengguna hingga solusi berhasil ditemukan. Jumlah total tes yang dilakukan untuk mendapatkan solusi juga akan dicetak setelah dihitung di fungsi solve().
7. Program berakhir.

Fungsi menarik pertama yang dijelaskan adalah fungsi permutations(). Fungsi ini memiliki 2 parameter masukan *list­­\_num* (list angka dari 0 sampai 9) dan *r* (panjang elemen permutasi dari *list\_num*)digunakan untuk mencari seluruh permutasi dari input list angka untuk kemudian diuji dengan karakter-karakter yang ada. Sebagian besar algoritmanya diambil dari dokumentasi resmi Python 3 ([itertools — Functions creating iterators for efficient looping — Python 3.9.1 documentation](https://docs.python.org/3/library/itertools.html" \l "itertools.permutations)). Modifikasi yang dilakukan adalah membuat parameter *r* menjadi wajib ada (tidak boleh None). Cara kerja fungsi ini secara garis besar adalah menyimpan permutasi saat ini pada variabel *index* yang akan diupdate oleh variabel *cycle* (dari prinsip permutasi siklik). Misal elemen *cycle* di indeks ke-*i* saat itu adalah *j*, maka elemen *index* ke-*i* dari kiri akan ditukar dengan elemen ke­-*j* dari kanan. Hal ini terus dilakukan hingga seluruh elemen telah ditemukan, tentunya dengan memperhatikan beberapa kasus seperti jika *cycle* mencapai 0.

Fungsi permutations() akan digunakan untuk memecahkan permasalahan dalam fungsi solve(). Cara kerja fungsi solve() adalah meminta beberapa masukan yaitu operands (*op*), hasil (*res*), karakter unik (*char*), dan list angka 0-9 (*list\_num*). Cara kerja fungsi secara garis besar adalah untuk setiap elemen permutasi *perm*, akan dilakukan penjumlahan terhadap angka yang merepresentasikan operands dan result. Jumlah operand kemudian akan dicocokkan dengan jumlah result. Jika jumlah sesuai dan jumlah digit result sesuai dengan jumlah huruf pada stringnya, maka solusi akan diprint dan keluar dari fungsi. Setiap iterasinya akan dicatat pada variabel *count* untuk ditampilkan di akhir program.

1. **Source Code**

Graphical user interface, text, application, email

Description automatically generated

Text

Description automatically generated with medium confidence

Text

Description automatically generated with medium confidence

1. **Testing**

|  |  |
| --- | --- |
| No. Test Case | Hasil |
| 1 | Text  Description automatically generated |
| 2 | Text  Description automatically generated |
| 3 | Text  Description automatically generated |
| 4 | Text  Description automatically generated |
| 5 | Text  Description automatically generated |
| 6 | Text  Description automatically generated |
| 7 | Text  Description automatically generated |
| 8 |  |

1. **Alamat Source Code & Checklist**

Kode dapat diakses di tautan Google Drive berikut:

<https://drive.google.com/drive/u/0/folders/1wb5hmqUxUlWa5SahqD7oxCDjFZB-GyF3>

Atau dapat juga diakses di repositori GitHub:

<https://github.com/rayfazt/stima-cryptarithm>

Checklist

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **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 *cryptarithmetic* dengan dua buah operand |  | ✓ |
| 5 | Solusi *cryptarithmetic* benar untuk persoalan *cryptarithmetic* untuk lebih dari dua buah operand | ✓ |  |