Laporan Penyelesaian *Cryptarithmetic* dengan Algoritma Brute Force

TUGAS KECIL STRATEGI ALGORITMA

Oleh:

Karel Renaldi

13519180

PROGRAM STUDI TEKNIK INFORMATIKA

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

2020

A. Langkah – langkah penyelesaian Cryptarithmethic dengan brute force

Pemecahan masalah Cryptarithmethic dengan menggunakan teknik brute force secara general adalah sebagai berikut :

Pada source code tucil.cpp yang ada pada folder src, terdapat 7 fungsi pembantu dengan 1 fungsi utama, pada fungsi pembantu, terdapat fungsi :

1. getListWord
   Fungsi ini menerima 2 parameter yaitu vector<string> / array of string & nama file dari file yang akan dibaca. fungsi ini akan membaca file dengan nama yang ada di parameter dan memasukan setiap operan kedalam vector<string> / array of string
2. getListUniqueChars
   Fungsi ini menerima 2 parameter yaitu vector<string> / array of string & vector<char> / array of char. Fungsi ini akan memasukan setiap karakter unik kedalam vector<char> / array of char.
3. getListIdxNonZero
   Fungsi ini menerima 3 parameter yaitu vector<int> / array of integer, vector<string> / array of string, dan vector<char> / array of character. Fungsi ini akan memasukkan semua index pada vector<char> yang mana karakter pada index tersebut di vector<char> ini merupakan character pertama dari elemen" vector<string>
4. getIndex
   Fungsi ini menerima 2 parameter yaitu vector<char> & sebuah character.Fungsi ini akan mengembalikan index vector<char> yang mana element pada index tersebut sama dengan character pada parameter
5. nextPermutation
   Fungsi ini menerima 1 parameter yaitu vector<int>. Fungsi ini akan merubah vector<int> dengan bentuk permutasinya. Fungsi ini juga menggunakan *lexicographical algorithm*.Fungsi ini akan bernilai true jika permutasi dari vector<int> masih ada.
6. getMaxNumberAtI
   Fungsi ini menerima 2 parameter yaitu vector<int> & indeks bertipe integer. Fungsi ini berfungsi untuk menentukan nilai maksimum pada vector<int> / array of integer pada posisi ke i.
7. includeIdxNonZero
   Fungsi ini menerima 2 parameter vector<int> yang berisi index" yang tidak boleh bernilai nol dan vector<int> yang berisi angka hasil permutasi. Fungsi akan mengembalikan nilai true jika element pada vector<int> hasil permutasi ada yang bernilai nol pada posisi yang ada di vector<int> yang berisi index" yang tidak boleh bernilai nol.

Pada fungsi main, terdapat beberapa langkah yang dilakukan dalam mencapai output / hasil , yaitu sebagai berikut :

1. Baca file lalu masukkan kedalam array bertipe string
2. Buat array yang berisi karakter-karakter unik dari array string yang pada langkah 1 dihasilkan.
3. Buat array yang berisi angka dari 0 sampai N dimana N adalah panjang array karakter unik.
4. Lakukan permutasi dari array angka tersebut.Sambungkan angka-angka hasil permutasi tersebut sesuai dengan representasi string yang ada pada array di langkah 1 lalu masukkan ke array bertipe integer yang baru.
5. Lakukan iterasi penjumlahan sampai elemen kedua terakhir pada array yang dihasilkan di langkah 4.
6. Cek apakah hasil penjumlahan tersebut sama dengan elemen terakhir array pada langkah 4 atau tidak. Jika ya maka lanjut ke langkah 8, jika tidak maka lanjut ke langkah 7.
7. Ganti nilai nilai pada elemen di array langkah 3 dengan nilai-nilai 0 – 9 dengan menggunakan permutasi N dari 10 (angka 0 - 9) dimana kemungkinan angka angka permutasi tersebut tidak pernah muncul pada langkah 4.Jika sudah tidak ada permutasi lagi maka lanjut ke langkah 8 jika masih ada lanjut ke langkah 4.
8. Hasil dari cryparithm ditemukan jika permutasi pada langkah 7 belum selesai. Jika permutasi pada langkah 7 sudah selesai maka tidak ada solusi dari soal cryparithm yang ada pada file.

B. Source Code

```cpp
// Karel Renaldi / 13519180 / K-04 / Tucil Stima 1

#include <bits/stdc++.h>
using namespace std;
using namespace std::chrono;

void getListWord(vector<string> &operands, string filename)
{
    ifstream file(filename);
    string input;

    while (file >> input)
        operands.push_back(input);

    operands.erase(operands.end() - 2);
    for (int i = 0; i < operands.size(); i++)
        if (operands[i].find('+') != string::npos)
            operands[i].pop_back();
}

void getListUniqueChars(vector<string> &words, vector<char> &uniqueChars
)
{
    for (string word : words)
        for (char character : word)
            if (find(uniqueChars.begin(), uniqueChars.end(), character)
== uniqueChars.end())
                uniqueChars.push_back(character);
}

void getListIdxNonZero(vector<int> &idxNotZero, vector<string> &words, v
ector<char> &uniqueChars)
{
    for (int i = 0; i < uniqueChars.size(); i++)
    {
        int j = 0;
        bool found = false;
        while (j < words.size() && !found)
        {
            if (words[j][0] == uniqueChars[i])
            {
                found = true;
                idxNotZero.push_back(i);
            }
            j++;
```

```
        }
    }
}

int getIndex(vector<char> &uniqueChars, char chr)
{
    auto position = find(uniqueChars.begin(), uniqueChars.end(), chr);
    if (position != uniqueChars.end())
        return position - uniqueChars.begin();
    else
        return -1;
}

bool nextPermutation(vector<int> &v)
{
    int i, j;

    i = v.size() - 1;

    // Find pivot index
    while (i > 0 && v[i - 1] >= v[i])
        i--;

    if (i == 0)
        return false;

    // find suffix element > pivot
    j = v.size() - 1;
    while (v[j] <= v[i - 1])
        j--;

    // swap element suffix > pivot with pivot
    swap(v[i - 1], v[j]);

    j = v.size() - 1;

    // reverse suffix array
    while (i < j)
    {
        int temp = v[i];
        v[i] = v[j];
        v[j] = temp;
        i++;
        j--;
    }
    return true;
}
```

```cpp
int getMaxNumberAtI(vector<int> v, int i)
{
    return 9 - (v.size() - 1) + i;
}

bool includeIdxNonZero(vector<int> &idxNotZero, vector<int> &v)
{
    int i = 0;
    bool found = false;

    while (i < idxNotZero.size() && !found)
    {
        if (v[idxNotZero[i]] == 0)
            found = true;
        i++;
    }

    return found;
}

int main()
{
    string filename;
    string path = "../test/";

    cout << "Masukkan nama file: ";
    cin >> filename;
    cout << endl;

    string fullPath = path + filename;

    bool solve = false;
    vector<string> words;
    vector<char> uniqueChars;
    vector<int> idxNotZero;

    getListWord(words, fullPath);

    time_point<system_clock> start, end;
    start = system_clock::now();

    getListUniqueChars(words, uniqueChars);
    getListIdxNonZero(idxNotZero, words, uniqueChars);

    // Initialize charnNum
    vector<int> charNum(uniqueChars.size()), wordNum(words.size());
```

```
    iota(charNum.begin(), charNum.end(), 0);

    int i = charNum.size() - 1;
    bool finish = false, solved = true;
    int test = 0;
    while (!finish)
    {
        vector<int> currentCharNum = charNum;
        do
        {
            int res = 0;
            for (int i = 0; i < words.size(); i++)
            {
                int temp = 0;
                for (int j = 0; j < words[i].size(); j++)
                {
                    int idx = getIndex(uniqueChars, words[i][j]);
                    temp += (int)(currentCharNum[idx] * pow(10.0, (float
)(words[i].size() - j - 1)));
                }
                wordNum[i] = temp;
            }

            for (int i = 0; i < wordNum.size() - 1; i++)
                res += wordNum[i];

            if (res == wordNum.back())
            {
                if (!includeIdxNonZero(idxNotZero, currentCharNum))
                {
                    charNum = currentCharNum;
                    solved = true;
                    finish = true;
                }
            }
            test++;
        } while (nextPermutation(currentCharNum) && !finish);

        if (!finish)
        {
            if (charNum[i] + 1 <= getMaxNumberAtI(charNum, i))
                charNum[i]++;
            else
            {
                while (charNum[i] + 1 > getMaxNumberAtI(charNum, i) && i
 > 0)
                    i--;
```

```
                if (i != 0 || charNum[i] + 1 <= getMaxNumberAtI(charNum,
 i))
                {
                    charNum[i] += 1;
                    while (charNum[i] + 1 <= getMaxNumberAtI(charNum, i
+ 1) && i < charNum.size() - 1)
                    {
                        charNum[i + 1] = charNum[i] + 1;
                        i++;
                    }
                }
                else
                {
                    finish = true;
                }
            }
        }
    }

    // Output
    for (int i = 0; i < words.size(); i++)
    {
        cout << words[i] << endl;
        if (i == words.size() - 2)
            cout << "-------+" << endl;
    }
    cout << endl;

    if (solved)
    {
        for (int i = 0; i < words.size(); i++)
        {
            for (int j = 0; j < words[i].size(); j++)
            {
                int idx = getIndex(uniqueChars, words[i][j]);
                cout << charNum[idx];
            }
            cout << endl;
            if (i == words.size() - 2)
            {
                cout << "-------+" << endl;
            }
        }
    }
    else
    {
```

```
        cout << "Tidak ada solusi ditemukan" << endl;
    }

    cout << endl;

    end = system_clock::now();
    duration<double> elapsed_seconds = end - start;
    cout << "Waktu eksekusi: " << elapsed_seconds.count() << " "
         << "detik" << endl;
    cout << "Total test: " << test;
    return 0;
}
```

C. INPUT/OUTPUT

```
Masukkan nama file: test1.txt

SEND
MORE
-------+
MONEY

9567
1085
-------+
10652

Waktu eksekusi: 1.1589 detik
Total test: 844280
```

```
Masukkan nama file: test2.txt

NUMBER
NUMBER
-------+
PUZZLE

201689
201689
-------+
403378

Waktu eksekusi: 2.87722 detik
Total test: 1532493
```

```
Masukkan nama file: test3.txt

TILES
PUZZLES
-------+
PICTURE

91542
3077542
-------+
3169084

Waktu eksekusi: 6.84443 detik
Total test: 3328707
```

```
Masukkan nama file: test4.txt

COCA
COLA
-------+
OASIS

8186
8106
-------+
16292

Waktu eksekusi: 0.0528582 detik
Total test: 24277
```

```
Masukkan nama file: test5.txt

HERE
SHE
-------+
COMES

9454
894
-------+
10348

Waktu eksekusi: 0.270278 detik
Total test: 206373
```

```
Masukkan nama file: test6.txt

THREE
THREE
TWO
TWO
ONE
-------+
ELEVEN

84611
84611
803
803
391
-------+
171219

Waktu eksekusi: 4.67384 detik
Total test: 1756932
```

```
Masukkan nama file: test8.txt

CROSS
ROADS
-------+
DANGER

96233
62513
-------+
158746

Waktu eksekusi: 6.20705 detik
Total test: 3614574
```

```
Masukkan nama file: test7.txt

NO
GUN
NO
-------+
HUNT

87
908
87
-------+
1082

Waktu eksekusi: 0.0378873 detik
Total test: 25051
```

D. Link Alamat Drive :

https://github.com/karelrenaldi/Tucil1-Stima

E. Tabel Ceklist :

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**	