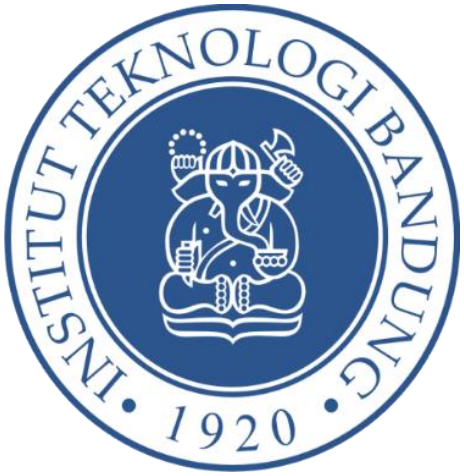
Laporan Tugas Kecil 1

IF2211 Strategi Algoritma

"*Penyelesaian Cryptarithmetic dengan Algoritma Brute Force*"



Oleh:

Nama : Nabilah Erfariani

NIM : 13519181

PROGRAM STUDI TEKNIK INFORMATIKA

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

2021

BAB 1

DESKRIPSI MASALAH

Penyelesaian *Cryptarithmetic* dengan Algoritma *Brute Force*

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.

Contoh:

```
  S E N D
+ M O R E
---------
M O N E Y
```

Solusinya adalah:

```
  9 5 6 7
+ 1 0 8 5
---------
1 0 6 5 2
```

Jadi, S = 9, E = 5, N = 6, D = 7, M = 1, O = 0, R = 8, Y = 2

Contoh-contoh *cryptrithmetic* dengan solusinya:

```
  JUNE      7924
+ JULY    + 7906
---------   ----
 APRIL      15830
```

```
    F O R T Y          2 9 7 8 6
      T E N              8 5 0
+     T E N          +   8 5 0
-------------      -------------
  S I X T Y          3 1 4 8 6
```

BAB 2
SOLUSI

Algoritma *Brute Force*

1. Program akan menerima file berupa *input.txt*. File tersebut terdiri dari operan dan hasil. Program lalu akan membuat *list of characters* yang unik dari operan dan hasil tersebut dan menyabaikan tanda “+”.
2. Program melakukan enumerasi permutasi dari angka 0-9, sehingga menghasilkan sebuah *list of integer* yang unik.
3. Program melakukan pemetaan dari *list of integer* pada Langkah 2 ke *list of character* pada langkah pertama.
4. Permutasi dilakukan dengan memasangkan karakter unik dengan integer. Pemasangan tersebut dilakukan sebanyak jumlah karakter unik.
5. Pemasangan dilakukan secara satu-per-satu dari setiap operan dan hasil. Program akan melakukan pengecekan apakah angka saat ini merupakan angka pertama dari operan/hasil, pengecekan apakah nilai yang akan di-assign sudah diberikan kepada huruf sebelumnya, serta melakukan pengecekan apakah terjadi inkonsistensi nilai(huruf yang sama memiliki nilai yang berbeda).
6. Hasil pemetaan pasangan tersebut akan dikonversi menjadi *array of string* yang terdiri dari huruf beserta nilai yang di-assign nya.
7. Program lalu akan menguji hasil pemetaan tersebut dengan melakukan operasi penjumlahan.
8. Jika operasi penjumlahan benar, maka program akan mengeksekusi hasil tersebut. Jika tidak, maka program akan mengulang langkah ke-4 hingga langkah ke-7 sampai mendapatkan hasil yang benar.

BAB 3

IMPLEMENTASI PROGRAM DALAM JAVA

```java
/* Nama : Nabilah Erfariani
   NIM  : 13519181
   Kelas: K04
   Tugas Kecil IF2211 Strategi Algoritma
   Cryptarithmetic
*/

import java.io.File;
import java.util.Scanner;
import java.util.List;
import java.util.ArrayList;
import java.io.IOException;
import java.nio.charset.StandardCharsets;
import java.nio.file.Files;
import java.util.Enumeration;
import java.util.Hashtable;
import java.nio.file.Path;
import java.nio.file.Paths;

public class Cryptarithmetic {
    /* Deklarasi variabel */
    public int pjgKata;
    public int i;
    public int idx;
    public int[] assignedNumber;
    public int value;
    public ArrayList<Integer> listAngkaBoleh = new ArrayList<Integer>();
    public ArrayList<ArrayList<Character>> listhuruf = new ArrayList<ArrayList<Character>>();
    public Hashtable<Character, Integer> jmlAssigned = new Hashtable<Character, Integer>();
    public Hashtable<Character, Integer> solusiAngka = new Hashtable<Character, Integer>();
```

```java
/* main program */
Run | Debug
public static void main(String[] args){
    /* read file txt to string */
    String nameFile = "../test/input1.txt";

    /* read AllLines in file */
    Path getFile = Paths.get(nameFile);
    try{
        /* menampilkan file ke layar */
        File outputFile = new File(nameFile);
        Scanner readerfile = new Scanner(outputFile);
        while (readerfile.hasNextLine()){
            String output = readerfile.nextLine();
            System.out.println(output);
        }
        readerfile.close();

        List<String> kata = Files.readAllLines(getFile, StandardCharsets.UTF_8);

        /* mendapatkan panjang kata hingga sebelum tanda + */
        int pjgKata = kata.size()-1;

        /* deklarasi waktu mulai dan waktu selesai perhitungan dengan tipe long in milisecond */
        long waktumulai = System.currentTimeMillis();
```

```
    /* Proses untuk mencari solusi angka dari setiap huruf */

    Cryptarithmetic cryptarithmetic = new Cryptarithmetic(pjgKata);
    for(int i=0; i<pjgKata-1; i++){
        for(int j = kata.get(i).length()-1; j>=0; j--){
            char z = kata.get(i).charAt(j);
            cryptarithmetic.getSolusiAngka(i,z);
        }
    }
    for(int j = kata.get(kata.size()-1).length()-1; j >=0; j--){
        char y = kata.get(pjgKata).charAt(j);
        cryptarithmetic.getSolusiAngka(pjgKata-1,y);
    }

    cryptarithmetic.setAngkaBoleh();
    cryptarithmetic.calculate();

    long waktuselesai = System.currentTimeMillis();
    long waktueksekusi = (waktuselesai - waktumulai);

    cryptarithmetic.execute();
    System.out.println(" ");
    System.out.println("Waktu eksekusi: " +waktueksekusi+" ms");
} catch (IOException e){
    e.printStackTrace();
}
```

```
/* Dibawah ini Fungsi-fungsi/prosedur-prosedur */

public boolean isSolved(){
    /* melakukan pengecekan apakah semua huruf sudah terisi angka */
    boolean solved = true;
    for(int i=0;i<listhuruf.size();i++)
    {
        if(listhuruf.get(i).size()!=assignedNumber[i])
            solved=false;
    }
    return solved;
}

public void setAngkaBoleh(){
    /* Set angka-angka yang diperbolehkan untuk di-assign ke huruf*/
    for (int i=0;i<=listhuruf.get(pjgKata-1).size();i++){
        listAngkaBoleh.add(0);
    }
}
```

```
public boolean sudahAda (char huruf){
    /* melakukan pengecekan apakah sudah ada angka yang di-assign pada huruf lainnya*/
    char key=' ';
    boolean udahAda = false;
    Enumeration<Character> enumerasiKey = solusiAngka.keys();
    while(!udahAda && enumerasiKey.hasMoreElements()){
            key = enumerasiKey.nextElement();
            if (solusiAngka.get(key)==value){
                udahAda = true;}
    }
    if(udahAda){
        if(key != huruf){
            return true;
        }else{
            return false;}
    }
    return udahAda;
}

public void getSolusiAngka (int i, char huruf){
    /* assign solusi ke list huruf yg unik */
    listhuruf.get(i).add(huruf);
    jmlAssigned.put(huruf,0);
}
```

```java
public boolean isFirstChar (char huruf){
    /* pengecekan apakah huruf saat ini sama dengan huruf paling depan/huruf pertama */
    boolean firstchar = false;
    for(int i=0; i<pjgKata; i++){
        if(!firstchar){
            int first = listhuruf.get(i).size()-1;
            if(listhuruf.get(i).get(first).equals(huruf)){
                firstchar = true;
            }
        }
    }
    return firstchar;
}

public boolean isInkonsistenValue (char huruf) {
    /* melakukan pengecekan apakah huruf yang sama memiliki nilai yang berbeda */
    if(solusiAngka.getOrDefault(huruf,10) == 10){
        return false;
    } else {
        return value≠solusiAngka.get(huruf);
    }
}
```

```java
public void permutasi(){
    /* melakukan permutasi satu-satu dengan mengecek setiap huruf ke depan */
    value = 0;
    int last = listhuruf.size()-1;
    idx++;
    idx = idx % pjgKata;
    if(idx==0){
        i++;
        jumlahpercobaan = jumlahpercobaan+1;
    }
    if(assignedNumber[idx]== listhuruf.get(idx).size() && (idx≠last && i<listhuruf.get(last).size())) {
        permutasi();
        jumlahpercobaan = jumlahpercobaan+1;
    }
}
```

```java
public void permute(){
    /* melakukan permutasi satu-satu dengan mengecek setiap huruf ke belakang */
    int A = pjgKata-1;
    idx = idx + A ;
    idx = idx % pjgKata;
    if(idx==A){
        i--;
    }
    if(i≥listhuruf.get(idx).size()){
        permute();
        jumlahpercobaan = jumlahpercobaan+1;
    }else{
        value = solusiAngka.get(listhuruf.get(idx).get(i))+1;
        assignedNumber[idx]--;
        char huruf = listhuruf.get(idx).get(i);
        if(jmlAssigned.get(huruf)≠1){
            int jmlAssignedTiapHuruf = jmlAssigned.get(huruf);
            jmlAssigned.put(huruf, jmlAssignedTiapHuruf - 1);
            jumlahpercobaan = jumlahpercobaan+1;
        }else {
            solusiAngka.remove(huruf);
            int jmlAssignedTiapHuruf = jmlAssigned.get(huruf);
            jmlAssigned.put(huruf,jmlAssignedTiapHuruf-1);
            jumlahpercobaan = jumlahpercobaan+1;
        }
    }
    if(idx==A){
        permute();
        jumlahpercobaan = jumlahpercobaan+1;
    }
}
```

```
public void calculate() {
    /* proses pencarian solusi dan assign nilai ke huruf */
    value = 0;
    i = 0;
    idx = 0;
    while(!isSolved()) {
        if (idx == pjgKata - 1) {
            int A = listAngkaBoleh.get(i);
            for (int b = 0; b < pjgKata - 1; b++) {
                if (i < listhuruf.get(b).size()) {
                    A += solusiAngka.get(listhuruf.get(b).get(i));
                    jumlahpercobaan = jumlahpercobaan+1;
                }
            }
            char huruf = listhuruf.get(idx).get(i);
            listAngkaBoleh.set(i + 1, A / 10);
            value = A % 10;
            if ((value==0 && isFirstChar(huruf)) || sudahAda(huruf) || isInkonsistenValue(huruf)) {
                listAngkaBoleh.set(i + 1, 0);
                permute();
                jumlahpercobaan = jumlahpercobaan+1;
            }else {
                solusiAngka.put(huruf,value);
                assignedNumber[idx]++;
                int jmlAssignedTiapHuruf = jmlAssigned.get(huruf);
                jmlAssigned.put(huruf,jmlAssignedTiapHuruf+1);
                permutasi();
                jumlahpercobaan = jumlahpercobaan+1;
            }
```

```
        }else {
            char huruf = listhuruf.get(idx).get(i);
            while ((value==0 && isFirstChar(huruf)) || sudahAda(huruf) || isInkonsistenValue(huruf) && value <= 9) {
                value++;
                jumlahpercobaan = jumlahpercobaan+1;
            }
            if (value > 9) {
                permute();
                jumlahpercobaan = jumlahpercobaan+1;
            }else{
                solusiAngka.put(huruf,value);
                assignedNumber[idx]++;
                int jmlAssignedTiapHuruf = jmlAssigned.get(huruf);
                jmlAssigned.put(huruf,jmlAssignedTiapHuruf+1);
                permutasi();
                jumlahpercobaan = jumlahpercobaan+1;
            }
        }
    }
}
```

```
public Cryptarithmetic(int pjgKata){
    /* membuat list of character */
    int i = 0;
    listhuruf = new ArrayList<>();
    for(i = 0; i<pjgKata; i++){
        listhuruf.add(new ArrayList<Character>());
    }
    listAngkaBoleh = new ArrayList<>();
    assignedNumber = new int[pjgKata];
    solusiAngka = new Hashtable<>();
    jmlAssigned = new Hashtable<>();
    for(i=0; i<pjgKata; i++){
        assignedNumber[i]=0;
    }
    this.pjgKata = pjgKata;
}
```

```java
public void execute(){
    /* mencetak hasil kalkulasi ke layar*/
    System.out.println(" ");
    int maxlist = listhuruf.get(0).size();
    for (int i = 1;i<listhuruf.size();i++){
        if(listhuruf.get(i).size()>maxlist){
            maxlist = listhuruf.get(i).size();
        }
    }
    for(int i=0;i<pjgKata-1;i++){
        for(int k = listhuruf.get(i).size(); k<maxlist;k++) {
            System.out.print(" ");
        }
        for (int j=listhuruf.get(i).size()-1;j>=0;j--){
            char huruf = listhuruf.get(i).get(j);
            System.out.print(solusiAngka.getOrDefault(huruf,-1));
        }
        System.out.println("");
    }
    System.out.println("———————— +");
    for (int j=listhuruf.get(pjgKata-1).size()-1;j>=0;j--){
        char huruf = listhuruf.get(pjgKata-1).get(j);
        System.out.print(solusiAngka.getOrDefault(huruf,-1));
    }
    System.out.println(" ");
    System.out.println(" ");
    System.out.println("Jumlah percobaan: "+jumlahpercobaan);
}
```

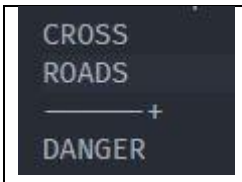
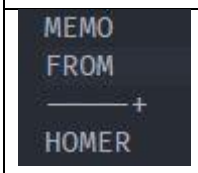
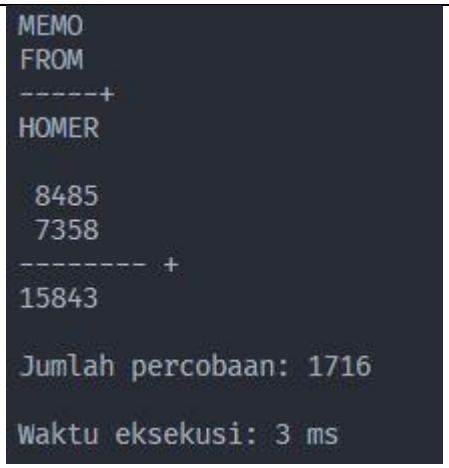
BAB IV

EKSEKUSI PROGRAM

INPUT	OUTPUT
NUMBER NUMBER ———+ PUZZLE	NUMBER NUMBER -----+ PUZZLE 798321 798321 -------- + 596642 Jumlah percobaan: 240 Waktu eksekusi: 1 ms

TILES PUZZLES ———————+ PICTURE	TILES PUZZLES -------+ PICTURE 91542 3077542 -------- + 3169084 Jumlah percobaan: 1621 Waktu eksekusi: 4 ms
CLOCK TICK TOCK ———————+ PLANET	CLOCK TICK TOCK ------+ PLANET 90892 6592 6892 -------- + 104376 Jumlah percobaan: 19902 Waktu eksekusi: 16 ms
COCA COLA ———————+ OASIS	COCA COLA -----+ OASIS 8186 8106 -------- + 16292 Jumlah percobaan: 17838 Waktu eksekusi: 14 ms
HERE SHE ———————+ COMES	HERE SHE -----+ COMES 9454 894 -------- + 10348 Jumlah percobaan: 2042 Waktu eksekusi: 3 ms

DOUBLE DOUBLE TOIL ———————+ TROUBLE	DOUBLE DOUBLE TOIL -------+ TROUBLE 798064 798064 1936 -------- + 1598064 Jumlah percobaan: 9155 Waktu eksekusi: 8 ms
NO GUN NO ————+ HUNT	NO GUN NO ----+ HUNT 87 908 87 -------- + 1082 Jumlah percobaan: 8305 Waktu eksekusi: 7 ms
THREE THREE TWO TWO ONE ———————+ ELEVEN	THREE THREE TWO TWO ONE ------+ ELEVEN 84611 84611 803 803 391 -------- + 171219 Jumlah percobaan: 3990 Waktu eksekusi: 4 ms

Link Source Code:

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

Poin	Ya	Tidak
1. Program berhasil dikompilasi	✓	
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.	✓	