LAPORAN TUGAS KECIL 1 IF2211 STRATEGI ALGORITMA

SEMESTER II TAHUN 2022/2023

PENYELESAIAN PERMAINAN KARTU 24 DENGAN ALGORITMA BRUTE FORCE

Disusun oleh :

Alex Sander **13521061**

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

BANDUNG

2022

DAFTAR ISI

BAB I

DESKRIPSI MASALAH

Permainan kartu 24 adalah permainan kartu aritmatika dengan tujuan mencari cara untuk mengubah 4 buah angka random sehingga mendapatkan hasil akhir sejumlah 24. Permainan ini menarik cukup banyak peminat dikarenakan dapat meningkatkan kemampuan berhitung serta mengasah otak agar dapat berpikir dengan cepat dan akurat. Permainan Kartu 24 biasa dimainkan dengan menggunakan kartu remi. Kartu remi terdiri dari 52 kartu yang terbagi menjadi empat suit (sekop, hati, keriting, dan wajik) yang masing-masing terdiri dari 13 kartu (As, 2, 3, 4, 5, 6, 7, 8, 9, 10, Jack, Queen, dan King). Yang perlu diperhatikan hanyalah nilai kartu yang didapat (As, 2, 3, 4, 5, 6, 7, 8, 9, 10, Jack, Queen, dan King). As bernilai 1, Jack bernilai 11, Queen bernilai 12, King bernilai 13, sedangkan kartu bilangan memiliki nilai dari bilangan itu sendiri. Pada awal permainan moderator atau salah satu pemain mengambil 4 kartu dari dek yang sudah dikocok secara random. Permainan berakhir ketika pemain berhasil menemukan solusi untuk membuat kumpulan nilainya menjadi 24. Pengubahan nilai tersebut dapat dilakukan menggunakan operasi dasar matematika penjumlahan (+), pengurangan (-), perkalian (×), divisi (/) dan tanda kurung (()). Tiap kartu harus digunakan tepat sekali dan urutan penggunaannya bebas.

Dikutip dari:

https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2015-2016/Makalah-2016/MakalahStima-2016-038.pdf

1. Tulislah program kecil (sederhana) dalam Bahasa C/C++/Java yang mengimplementasikan algoritma Brute Force untuk mencari solusi word search puzzle.

2. Input: 4 angka/huruf yang terdiri dari: (A, 2, 3, 4, 5, 6, 7, 8, 9, 10, J, Q, K).

Contoh input:

A 8 9 Q

Selain itu, input juga dapat dilakukan dengan program men-generate 4 angka/hurufnya sendiri secara random. Pengguna dapat memilih apakah program meminta input dari pengguna atau generate sendiri. Apabila masukan tidak sesuai, maka program menampilkan luaran “Masukan tidak sesuai” dan akan meminta ulang.

3. Output:

a. Banyak solusi yang ditemukan.

b. Solusi dari permainan kartu 24 ditampilkan di layar dan terdapat opsi untuk menyimpan solusi dalam file text. Untuk contoh kasus di atas A 8 9 Q, maka salah satu solusinya adalah:

((9 + A) - 8) * Q atau ((9 + 1) - 8) * 12

c. Waktu eksekusi program (tidak termasuk waktu pembacaan file input).

BAB II

ALGORITMA

Algoritma yang digunakan akan memeriksa semua permutasi yang mungkin dari ke-empat angka yang diterima. Aspek yang akan dipermutasikan adalah posisi ke-empat angka/nilai serta kombinasi operator-operator yang akan digunakan di antaranya.

Permutasi posisi ke-empat angka dapat dengan mudah dicari, dengan jumlah permutasinya adalah 4! atau 24. Kemudian akan dicari kombinasi operator-operator yang akan digunakan. Pola-pola yang muncul saat menggunakan operator +,-,*,/,() adalah sebagai berikut.

1. ((a opr b) opr c) opr d
2. (a opr (b opr c)) opr d
3. a opr ((b opr c) opr d)
4. a opr (b opr (c opr d))
5. (a opr b) opr (c opr d)

a,b,c,d adalah angka/nilai yang digunakan/terima dan opr adalah operator yang digunakan

Setiap kasus akan memunculkan 1536 kemungkinan, didapatkan dari 4! x 4 x 4 x 4. Algoritma yang digunakan akan memeriksa setiap kemungkinan tersebut dan mencari tahu apakah hasil dari persamaan yang muncul adalah 24.

Program yang disusun tidak memiliki penanganan solusi dengan bentuk yang serupa.

BAB III

SOURCE CODE

A. Solver.java

```
public class Solver{
    public double[][] sols;
    public String[] solstype;
```

Class *Solver*, memiliki sebuah matriks berisi elemen double dan array berisi string.

```
public void isiArr(double a, double b, double c, double d){
    for(int i=0; i<24; i++){
        this.solstype[i] = "";
    }

    this.sols[0][0] = a;
    this.sols[0][1] = b;
    this.sols[0][2] = c;
    this.sols[0][3] = d;

    this.sols[1][0] = a;
    this.sols[1][1] = b;
    this.sols[1][2] = d;
    this.sols[1][3] = c;

    this.sols[2][0] = a;
    this.sols[2][1] = c;
    this.sols[2][2] = b;
    this.sols[2][3] = d;

    this.sols[3][0] = a;
    this.sols[3][1] = c;
    this.sols[3][2] = d;
    this.sols[3][3] = b;

    this.sols[4][0] = a;
```

Fungsi isiArr akan mengisi matriks dalam suatu variabel dengan class *Solver* dengan semua permutasi yang mungkin dari nilai a,b,c,d.

```
public Solver solproc1(){
    Solver mtemp;
    mtemp = new Solver(n: 2);
    for (int i = 0; i<24; i++){
        mtemp.sols[4*i][0] = this.sols[i][0]+this.sols[i][1];
        mtemp.sols[4*i][1] = this.sols[i][2];
        mtemp.sols[4*i][2] = this.sols[i][3];
        mtemp.solstype[4*i] = "(" +  this.sols[i][0] + " + " + this.sols[i][1] +")";

        mtemp.sols[4*i+1][0] = this.sols[i][0]-this.sols[i][1];
        mtemp.sols[4*i+1][1] = this.sols[i][2];
        mtemp.sols[4*i+1][2] = this.sols[i][3];
        mtemp.solstype[4*i+1] = "(" +  this.sols[i][0] + " - " + this.sols[i][1] +")";

        mtemp.sols[4*i+2][0] = this.sols[i][0]*this.sols[i][1];
        mtemp.sols[4*i+2][1] = this.sols[i][2];
        mtemp.sols[4*i+2][2] = this.sols[i][3];
        mtemp.solstype[4*i+2] = "(" +  this.sols[i][0] + " * " + this.sols[i][1] +")";

        mtemp.sols[4*i+3][0] = this.sols[i][0]/this.sols[i][1];
        mtemp.sols[4*i+3][1] = this.sols[i][2];
        mtemp.sols[4*i+3][2] = this.sols[i][3];
        mtemp.solstype[4*i+3] = "(" +  this.sols[i][0] + " / " + this.sols[i][1] +")";
    }
    return mtemp;
}
```

```
public Solver solproc2(){
    Solver mtemp;
    mtemp = new Solver(n: 3);
    for (int i = 0; i<96; i++){
        mtemp.sols[4*i][0] = this.sols[i][0]+this.sols[i][1];
        mtemp.sols[4*i][1] = this.sols[i][2];
        mtemp.solstype[4*i] = "(" +  this.solstype[i] + " + " + this.sols[i][1] +")";

        mtemp.sols[4*i+1][0] = this.sols[i][0]-this.sols[i][1];
        mtemp.sols[4*i+1][1] = this.sols[i][2];
        mtemp.solstype[4*i+1] = "(" +  this.solstype[i] + " - " + this.sols[i][1] +")";

        mtemp.sols[4*i+2][0] = this.sols[i][0]*this.sols[i][1];
        mtemp.sols[4*i+2][1] = this.sols[i][2];
        mtemp.solstype[4*i+2] = "(" +  this.solstype[i] + " * " + this.sols[i][1] +")";

        mtemp.sols[4*i+3][0] = this.sols[i][0]/this.sols[i][1];
        mtemp.sols[4*i+3][1] = this.sols[i][2];
        mtemp.solstype[4*i+3] = "(" +  this.solstype[i] + " / " + this.sols[i][1] +")";
    }
    return mtemp;
}
```

```java
public Solver solproc3(){
    Solver mtemp;
    mtemp = new Solver(n: 4);
    for (int i = 0; i<384; i++){
        mtemp.sols[4*i][0] = this.sols[i][0]+this.sols[i][1];
        mtemp.solstype[4*i] = "(" +  this.solstype[i] + " + " + this.sols[i][1] +")";

        mtemp.sols[4*i+1][0] = this.sols[i][0]-this.sols[i][1];
        mtemp.solstype[4*i+1] = "(" +  this.solstype[i] + " - " + this.sols[i][1] +")";

        mtemp.sols[4*i+2][0] = this.sols[i][0]*this.sols[i][1];
        mtemp.solstype[4*i+2] = "(" +  this.solstype[i] + " * " + this.sols[i][1] +")";

        mtemp.sols[4*i+3][0] = this.sols[i][0]/this.sols[i][1];
        mtemp.solstype[4*i+3] = "(" +  this.solstype[i] + " / " + this.sols[i][1] +")";
    }
    return mtemp;
}
```

```java
public Solver solproc2n2(){
    Solver mtemp;
    mtemp = new Solver(n: 5);
    for (int i = 0; i<96; i++){
        mtemp.sols[4*i][0] = this.sols[i][0];
        mtemp.sols[4*i][1] = this.sols[i][1] + this.sols[i][2];
        mtemp.solstype[8*i] = this.solstype[i];
        mtemp.solstype[8*i+1] = "(" +  this.sols[i][1] + " + " + this.sols[i][2] +")";

        mtemp.sols[4*i+1][0] = this.sols[i][0];
        mtemp.sols[4*i+1][1] = this.sols[i][1] - this.sols[i][2];
        mtemp.solstype[8*i+2] = this.solstype[i];
        mtemp.solstype[8*i+3] = "(" +  this.sols[i][1] + " - " + this.sols[i][2] +")";

        mtemp.sols[4*i+2][0] = this.sols[i][0];
        mtemp.sols[4*i+2][1] = this.sols[i][1] * this.sols[i][2];
        mtemp.solstype[8*i+4] = this.solstype[i];
        mtemp.solstype[8*i+5] = "(" +  this.sols[i][1] + " * " + this.sols[i][2] +")";

        mtemp.sols[4*i+3][0] = this.sols[i][0];
        mtemp.sols[4*i+3][1] = this.sols[i][1] / this.sols[i][2];
        mtemp.solstype[8*i+6] = this.solstype[i];
        mtemp.solstype[8*i+7] = "(" +  this.sols[i][1] + " / " + this.sols[i][2] +")";
    }
    return mtemp;
}
```

```
public Solver solproc3n2(){
    Solver mtemp;
    mtemp = new Solver(n: 4);
    for (int i = 0; i<384; i++){
        mtemp.sols[4*i][0] = this.sols[i][0]+this.sols[i][1];
        mtemp.solstype[4*i] = "(" +  this.solstype[2*i] + " + " + this.solstype[2*i+1] +")";

        mtemp.sols[4*i+1][0] = this.sols[i][0]-this.sols[i][1];
        mtemp.solstype[4*i+1] = "(" +  this.solstype[2*i] + " - " + this.solstype[2*i+1] +")";

        mtemp.sols[4*i+2][0] = this.sols[i][0]*this.sols[i][1];
        mtemp.solstype[4*i+2] = "(" +  this.solstype[2*i] + " * " + this.solstype[2*i+1] +")";

        mtemp.sols[4*i+3][0] = this.sols[i][0]/this.sols[i][1];
        mtemp.solstype[4*i+3] = "(" +  this.solstype[2*i] + " / " + this.solstype[2*i+1] +")";
    }
    return mtemp;
}
```

```
public Solver solproc1n3(){
    Solver mtemp;
    mtemp = new Solver(n: 2);
    for (int i = 0; i<24; i++){
        mtemp.sols[4*i][0] = this.sols[i][0];
        mtemp.sols[4*i][1] = this.sols[i][2]+this.sols[i][1];
        mtemp.sols[4*i][2] = this.sols[i][3];
        mtemp.solstype[4*i] = "(" +  this.sols[i][1] + " + " + this.sols[i][2] +")";

        mtemp.sols[4*i+1][0] = this.sols[i][0];
        mtemp.sols[4*i+1][1] = this.sols[i][1]-this.sols[i][2];
        mtemp.sols[4*i+1][2] = this.sols[i][3];
        mtemp.solstype[4*i+1] = "(" +  this.sols[i][1] + " - " + this.sols[i][2] +")";

        mtemp.sols[4*i+2][0] = this.sols[i][0];
        mtemp.sols[4*i+2][1] = this.sols[i][2]*this.sols[i][1];
        mtemp.sols[4*i+2][2] = this.sols[i][3];
        mtemp.solstype[4*i+2] = "(" +  this.sols[i][1] + " * " + this.sols[i][2] +")";

        mtemp.sols[4*i+3][0] = this.sols[i][0];
        mtemp.sols[4*i+3][1] = this.sols[i][1]/this.sols[i][2];
        mtemp.sols[4*i+3][2] = this.sols[i][3];
        mtemp.solstype[4*i+3] = "(" +  this.sols[i][1] + " / " + this.sols[i][2] +")";
    }
    return mtemp;
}
```

```
public Solver solproc2n3(){
    Solver mtemp;
    mtemp = new Solver(n: 3);
    for (int i = 0; i<96; i++){
        mtemp.sols[4*i][0] = this.sols[i][0]+this.sols[i][1];
        mtemp.sols[4*i][1] = this.sols[i][2];
        mtemp.solstype[4*i] = "(" +  this.sols[i][0] + " + " + this.solstype[i] +")";

        mtemp.sols[4*i+1][0] = this.sols[i][0]-this.sols[i][1];
        mtemp.sols[4*i+1][1] = this.sols[i][2];
        mtemp.solstype[4*i+1] ="(" +  this.sols[i][0] + " - " + this.solstype[i] +")";

        mtemp.sols[4*i+2][0] = this.sols[i][0]*this.sols[i][1];
        mtemp.sols[4*i+2][1] = this.sols[i][2];
        mtemp.solstype[4*i+2] = "(" +  this.sols[i][0] + " * " + this.solstype[i] +")";

        mtemp.sols[4*i+3][0] = this.sols[i][0]/this.sols[i][1];
        mtemp.sols[4*i+3][1] = this.sols[i][2];
        mtemp.solstype[4*i+3] = "(" +  this.sols[i][0] + " / " + this.solstype[i] +")";
    }
    return mtemp;
}
```

```
public Solver solproc3v2(){
    Solver mtemp;
    mtemp = new Solver(n: 4);
    for (int i = 0; i<384; i++){
        mtemp.sols[4*i][0] = this.sols[i][1]+this.sols[i][0];
        mtemp.solstype[4*i] = "(" +  this.sols[i][1] + " + " + this.solstype[i] +")";

        mtemp.sols[4*i+1][0] = this.sols[i][1]-this.sols[i][0];
        mtemp.solstype[4*i+1] = "(" +  this.sols[i][1] + " - " + this.solstype[i] +")";

        mtemp.sols[4*i+2][0] = this.sols[i][1]*this.sols[i][0];
        mtemp.solstype[4*i+2] = "(" +  this.sols[i][1] + " * " + this.solstype[i] +")";

        mtemp.sols[4*i+3][0] = this.sols[i][1]/this.sols[i][0];
        mtemp.solstype[4*i+3] = "(" +  this.sols[i][1] + " / " + this.solstype[i] +")";
    }
    return mtemp;
}
```

Fungsi-fungsi *solprocXXX* akan mengembalikan *Solver* yang telah diproses dan berisi kemungkinan-kemungkinan operator yang mungkin. Variasi fungsi ini muncul dari 5 kemungkinan yang ada serta proses yang dipisahkan secara pertahap.

B. IO.java

```java
public static void createFile(String fname){
    try {
      File myObj = new File("test/" + fname);
      if (myObj.createNewFile()) {
        System.out.println("File created: " + myObj.getName());
      } else {
        System.out.println(x: "File already exists.");
      }
    } catch (Exception e) {
        System.out.println(x: "An error occurred.");
        e.printStackTrace();
  }
}
```

Fungsi createFile yang akan membuat suatu file dengan nama file tertentu.

```java
public static void outputFileSolusi(Double a, Double b, Double c, Double d, Solver p, Solver k, Solver q, Solver o, Solver l, String nama){
    int i,count=0;
    String s,aa,bb,cc,dd,sabcd;
    try {
        FileWriter write = new FileWriter("test/"+ nama);
        aa= Double.toString(a);
        bb= Double.toString(b);
        cc= Double.toString(c);
        dd= Double.toString(d);
        sabcd= aa + " " + bb + " " + cc + " " + dd;
        write.write(sabcd);
        write.write(str: "\n");

        for (i = 0; i<1536;i++){
          if (p.sols[i][0]==24) {
              write.write(p.solstype[i]);
              write.write(str: "\n");
              count++;
          }
          if (k.sols[i][0]==24) {
              write.write(k.solstype[i]);
              write.write(str: "\n");
              count++;
          }
          if (q.sols[i][0]==24) {
              write.write(q.solstype[i]);
              write.write(str: "\n");
              count++;
          }
```

```
            }
            if (o.sols[i][0]==24) {
                write.write(o.solstype[i]);
                write.write(str: "\n");
                count++;
            }
            if (l.sols[i][0]==24) {
                write.write(l.solstype[i]);
                write.write(str: "\n");
                count++;
            }
          }

          if (count==0) {
            write.write(str: "Tidak ada solusi yang ditemukan.");
          }
          else {
              s= Integer.toString(count);
              write.write("Jumlah solusi yang ditemukan adalah " + s);
          }
          write.close();
      } catch (IOException e) {
        e.printStackTrace();
      }
    }
```

Fungsi outputFileSolusi yang akan menuliskan semua solusi yang ditemukan dalam file tertentu.

C. Menu.java

```
public void pilihan(int x) {
    Scanner scan = new Scanner(System.in);
    int i,pilsave;
    int count=0;
    double a,b,c,d;
    long start,finish,timeE;
    boolean kartuValid;
    Solver m,n,q,r,p,o,k,l;
    switch (x) {
```

Fungsi pilihan yang akan digunakan untuk pilihan utama dalam kejalanan program. Dalam fungsi ini disusun sedemikian sehingga algoritma yang dibuat di file IO.java dan Solver.java dapat digunakan.

```
case 1:
    count=0;
    kartuValid = true;
    System.out.println(x: "Masukkan 4 kartu yang digunakan!");
    a = scan.next().charAt(index: 0);
        if ((a > 49) && (a <58)) {
            a = a-48;
        }
        else if (a==49) {
            a = 10;
        }
        else if (a==65) {
            a = 1;
        }
        else if (a==74) {
            a = 11;
        }
        else if (a==81) {
            a = 12;
        }
        else if (a==75) {
            a = 13;
        }
        else {
            kartuValid = false;
        }
```

```
while (kartuValid == false) {
    kartuValid = true;
    a = scan.next().charAt(index: 0);
    if ((a > 49) && (a <58)) {
        a = a-48;
    }
    else if (a==49) {
        a = 10;
    }
    else if (a==65) {
        a = 1;
    }
    else if (a==74) {
        a = 11;
    }
    else if (a==81) {
        a = 12;
    }
    else if (a==75) {
        a = 13;
    }
    else {
        kartuValid = false;
    }

    b = scan.next().charAt(index: 0);
    if ((b > 49) && (b <58)) {
        b = b-48;
```

Bagian penerimaan input serta penanganan input yang salah

```
System.out.println(x: "!!!RANDOM!!!");
count = 0;
int max = 13;
int min = 1;
a = Math.floor(Math.random() *(max - min + 1) + min);
b = Math.floor(Math.random() *(max - min + 1) + min);
c = Math.floor(Math.random() *(max - min + 1) + min);
d = Math.floor(Math.random() *(max - min + 1) + min);
```

Bagian penanganan kasus dimana dipilihnya pemilihan nilai secara acak/*Random*

```
m = new Solver(n: 1);
m.isiArr(a,b,c,d);

start = System.currentTimeMillis();

n = m.solproc1();
n = n.solproc2();
p = n.solproc3();
k = n.solproc3v2();

q = m.solproc1();
q = q.solproc2n2();
q = q.solproc3n2();

r = m.solproc1n3();
r = r.solproc2n3();
o = r.solproc3();
l = r.solproc3v2();


finish = System.currentTimeMillis();
timeE = finish-start;

System.out.println();
```

Bagian penyusunan fungsi-fungsi dalam Solver.java dan pengukuran waktu

```
for (i = 0; i<1536;i++) {
    if (p.sols[i][0]==24) {
        System.out.println(p.solstype[i]);
        count++;
    }
    if (k.sols[i][0]==24) {
        System.out.println(k.solstype[i]);
        count++;
    }
    if (q.sols[i][0]==24) {
        System.out.println(q.solstype[i]);
        count++;
    }
    if (o.sols[i][0]==24) {
        System.out.println(o.solstype[i]);
        count++;
    }
    if (l.sols[i][0]==24) {
        System.out.println(l.solstype[i]);
        count++;
    }
}

if (count !=0) {
    System.out.println("Banyaknya solusi yang ditemukan adalah " + count);
}
else {
    System.out.println(x: "Tidak ada solusi yang ditemukan.");
}

System.out.println("Time Elapsed : " + timeE + " ms");
```

Bagian display solusi yang didapatkan serta waktu yang telah diukur sebelumnya

```
System.out.println(x: "Apakah ingin menyimpan solusi dalam file?\n1.Ya\n2.Tidak");
System.out.print(s: "Pilihan: ");

pilsave = scan.nextInt();
while ((pilsave<1)||(pilsave>2)) {
    System.out.println(x: "Masukan tidak sesuai!");
    System.out.print(s: "Pilihan: ");
    pilsave = scan.nextInt();
}

if (pilsave == 1) {
    String filename;
    System.out.print(s: "Masukkan nama file yang ingin dibuat (dalam .txt): ");
    filename = scan.next();
    IO.createFile(filename);
    IO.outputFileSolusi(a,b,c,d,p,k,q,o,l, filename);
}

System.out.println();
```

Bagian pilihan penyimpanan hasil pencarian solusi serta penyusunan fungsi dari IO.java

D. main.java

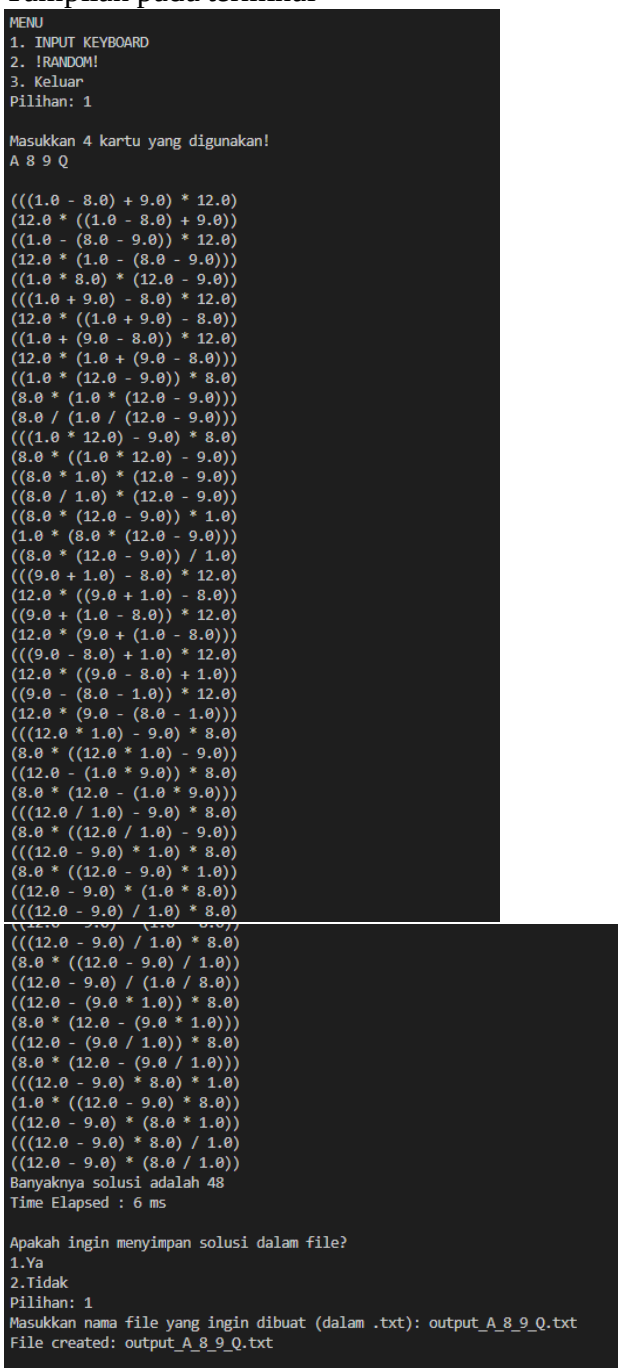
```java
public class main{
    Run | Debug
    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);
        Menu menu = new Menu();
        int pil;
        boolean aktif = true;
        System.out.println(x: "=================== Program Mulai ====================");
        System.out.println(x: "===== PERMAINAN KARTU 24 =====");
        while (aktif){
            System.out.println(x: "MENU");
            System.out.println(x: "1. INPUT KEYBOARD\n2. !RANDOM!\n3. Keluar");
            System.out.print(s: "Pilihan: ");
            pil = scan.nextInt();
            if ((pil < 3) && pil > 0) {
                System.out.println(x: "");
                menu.pilihan(pil);
                System.out.println(x: "=================== SELESAI ====================");
            } else if (pil == 3) {
                aktif = false;
            } else {
                System.out.println();
                System.out.println(x: "Masukan tidak sesuai!");
                System.out.println();
            }
        }
        System.out.println(x: "=================== Program Tutup ====================");
        scan.close();
    }
}
```

main.java berisi kode untuk interface awal tiap iterasi serta interface akhir program.

BAB IV

TEST CASE

1. Kasus Kontrol
   Program dijalankan dengan input sesuai pada file panduan Tucil1-Stima-2023.pdf
   a. A 8 9 Q
      Tampilan pada terminal



```
MENU
1. INPUT KEYBOARD
2. !RANDOM!
3. Keluar
Pilihan: 1

Masukkan 4 kartu yang digunakan!
A 8 9 Q

(((1.0 - 8.0) + 9.0) * 12.0)
(12.0 * ((1.0 - 8.0) + 9.0))
((1.0 - (8.0 - 9.0)) * 12.0)
(12.0 * (1.0 - (8.0 - 9.0)))
((1.0 * 8.0) * (12.0 - 9.0))
(((1.0 + 9.0) - 8.0) * 12.0)
(12.0 * ((1.0 + 9.0) - 8.0))
((1.0 + (9.0 - 8.0)) * 12.0)
(12.0 * (1.0 + (9.0 - 8.0)))
((1.0 * (12.0 - 9.0)) * 8.0)
(8.0 * (1.0 * (12.0 - 9.0)))
(8.0 / (1.0 / (12.0 - 9.0)))
(((1.0 * 12.0) - 9.0) * 8.0)
(8.0 * ((1.0 * 12.0) - 9.0))
((8.0 * 1.0) * (12.0 - 9.0))
((8.0 / 1.0) * (12.0 - 9.0))
((8.0 * (12.0 - 9.0)) * 1.0)
(1.0 * (8.0 * (12.0 - 9.0)))
((8.0 * (12.0 - 9.0)) / 1.0)
(((9.0 + 1.0) - 8.0) * 12.0)
(12.0 * ((9.0 + 1.0) - 8.0))
((9.0 + (1.0 - 8.0)) * 12.0)
(12.0 * (9.0 + (1.0 - 8.0)))
(((9.0 - 8.0) + 1.0) * 12.0)
(12.0 * ((9.0 - 8.0) + 1.0))
((9.0 - (8.0 - 1.0)) * 12.0)
(12.0 * (9.0 - (8.0 - 1.0)))
(((12.0 * 1.0) - 9.0) * 8.0)
(8.0 * ((12.0 * 1.0) - 9.0))
((12.0 - (1.0 * 9.0)) * 8.0)
(8.0 * (12.0 - (1.0 * 9.0)))
(((12.0 / 1.0) - 9.0) * 8.0)
(8.0 * ((12.0 / 1.0) - 9.0))
(((12.0 - 9.0) * 1.0) * 8.0)
(8.0 * ((12.0 - 9.0) * 1.0))
((12.0 - 9.0) * (1.0 * 8.0))
(((12.0 - 9.0) / 1.0) * 8.0)
(((12.0 - 9.0) / 1.0) * 8.0)
(8.0 * ((12.0 - 9.0) / 1.0))
((12.0 - 9.0) / (1.0 / 8.0))
((12.0 - (9.0 * 1.0)) * 8.0)
(8.0 * (12.0 - (9.0 * 1.0)))
((12.0 - (9.0 / 1.0)) * 8.0)
(8.0 * (12.0 - (9.0 / 1.0)))
(((12.0 - 9.0) * 8.0) * 1.0)
(1.0 * ((12.0 - 9.0) * 8.0))
((12.0 - 9.0) * (8.0 * 1.0))
(((12.0 - 9.0) * 8.0) / 1.0)
((12.0 - 9.0) * (8.0 / 1.0))
Banyaknya solusi adalah 48
Time Elapsed : 6 ms

Apakah ingin menyimpan solusi dalam file?
1.Ya
2.Tidak
Pilihan: 1
Masukkan nama file yang ingin dibuat (dalam .txt): output_A_8_9_Q.txt
File created: output_A_8_9_Q.txt
```

b. 7 5 8 3

```
MENU
1. INPUT KEYBOARD
2. !RANDOM!
3. Keluar
Pilihan: 1

Masukkan 4 kartu yang digunakan!
7 5 8 3

(3.0 - (7.0 * (5.0 - 8.0)))
((7.0 * 5.0) - (8.0 + 3.0))
(((7.0 * 5.0) - 8.0) - 3.0)
((7.0 * 5.0) - (3.0 + 8.0))
(((7.0 * 5.0) - 3.0) - 8.0)
((7.0 * (8.0 - 5.0)) + 3.0)
(3.0 + (7.0 * (8.0 - 5.0)))
(((7.0 * 3.0) - 5.0) + 8.0)
(8.0 + ((7.0 * 3.0) - 5.0))
((7.0 * 3.0) - (5.0 - 8.0))
(((7.0 * 3.0) + 8.0) - 5.0)
((7.0 * 3.0) + (8.0 - 5.0))
((5.0 * 7.0) - (8.0 + 3.0))
(((5.0 * 7.0) - 8.0) - 3.0)
((5.0 * 7.0) - (3.0 + 8.0))
(((5.0 * 7.0) - 3.0) - 8.0)
(8.0 - (5.0 - (7.0 * 3.0)))
(3.0 - ((5.0 - 8.0) * 7.0))
(8.0 - (5.0 - (3.0 * 7.0)))
((8.0 + (7.0 * 3.0)) - 5.0)
(((8.0 - 5.0) * 7.0) + 3.0)
(3.0 + ((8.0 - 5.0) * 7.0))
((8.0 - 5.0) + (7.0 * 3.0))
((8.0 - 5.0) + (3.0 * 7.0))
((8.0 + (3.0 * 7.0)) - 5.0)
(((3.0 * 7.0) - 5.0) + 8.0)
(8.0 + ((3.0 * 7.0) - 5.0))
((3.0 * 7.0) - (5.0 - 8.0))
(((3.0 * 7.0) + 8.0) - 5.0)
((3.0 * 7.0) + (8.0 - 5.0))
Banyaknya solusi adalah 30
Time Elapsed : 5 ms

Apakah ingin menyimpan solusi dalam file?
1.Ya
2.Tidak
Pilihan: 1
Masukkan nama file yang ingin dibuat (dalam .txt): output_7_5_8_3.txt
File created: output_7_5_8_3.txt
```

2. Kasus Random
   Program dijalankan dengan input nilai secara acak/*Random* menggunakan fitur yang ada
   a. K 6 3 J

   Tampilan pada terminal

```
==================== Program Mulai ====================
===== PERMAINAN KARTU 24 =====
MENU
1. INPUT KEYBOARD
2. !RANDOM!
3. Keluar
Pilihan: 2

!!!RANDOM!!!

13.0 6.0 3.0 11.0

(((13.0 + 6.0) - 11.0) * 3.0)
(3.0 * ((13.0 + 6.0) - 11.0))
((13.0 + (6.0 - 11.0)) * 3.0)
(3.0 * (13.0 + (6.0 - 11.0)))
(((13.0 - 11.0) + 6.0) * 3.0)
(3.0 * ((13.0 - 11.0) + 6.0))
((13.0 - (11.0 - 6.0)) * 3.0)
(3.0 * (13.0 - (11.0 - 6.0)))
(((6.0 + 13.0) - 11.0) * 3.0)
(3.0 * ((6.0 + 13.0) - 11.0))
((6.0 + (13.0 - 11.0)) * 3.0)
(3.0 * (6.0 + (13.0 - 11.0)))
(((6.0 - 11.0) + 13.0) * 3.0)
(3.0 * ((6.0 - 11.0) + 13.0))
((6.0 - (11.0 - 13.0)) * 3.0)
(3.0 * (6.0 - (11.0 - 13.0)))
Banyaknya solusi yang ditemukan adalah 16
Time Elapsed : 18 ms

Apakah ingin menyimpan solusi dalam file?
1.Ya
2.Tidak
Pilihan: 1
Masukkan nama file yang ingin dibuat (dalam .txt): output_K_6_3_J.txt
File created: output_K_6_3_J.txt
```

output_K_6_3_J.txt

```
13.0 6.0 3.0 11.0
(((13.0 + 6.0) - 11.0) * 3.0)
(3.0 * ((13.0 + 6.0) - 11.0))
((13.0 + (6.0 - 11.0)) * 3.0)
(3.0 * (13.0 + (6.0 - 11.0)))
(((13.0 - 11.0) + 6.0) * 3.0)
(3.0 * ((13.0 - 11.0) + 6.0))
((13.0 - (11.0 - 6.0)) * 3.0)
(3.0 * (13.0 - (11.0 - 6.0)))
(((6.0 + 13.0) - 11.0) * 3.0)
(3.0 * ((6.0 + 13.0) - 11.0))
((6.0 + (13.0 - 11.0)) * 3.0)
(3.0 * (6.0 + (13.0 - 11.0)))
(((6.0 - 11.0) + 13.0) * 3.0)
(3.0 * ((6.0 - 11.0) + 13.0))
((6.0 - (11.0 - 13.0)) * 3.0)
(3.0 * (6.0 - (11.0 - 13.0)))
Jumlah solusi yang ditemukan adalah 16
```

b. K 6 2 K

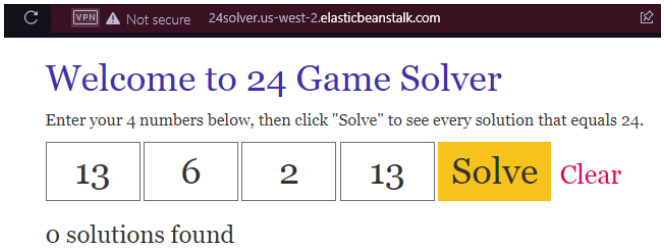
Tampilan pada terminal

```
MENU
1. INPUT KEYBOARD
2. !RANDOM!
3. Keluar
Pilihan: 2

!!!RANDOM!!!

13.0 6.0 2.0 13.0

Tidak ada solusi yang ditemukan.
Time Elapsed : 7 ms

Apakah ingin menyimpan solusi dalam file?
1.Ya
2.Tidak
Pilihan: 1
Masukkan nama file yang ingin dibuat (dalam .txt): output_K_6_2_K.txt
File created: output_K_6_2_K.txt
```

output_K_6_2_K.txt

```
13.0 6.0 2.0 13.0
Tidak ada solusi yang ditemukan.
```

Hasil pada 24solver.us



c. 9 10 7 3

Tampilan pada terminal

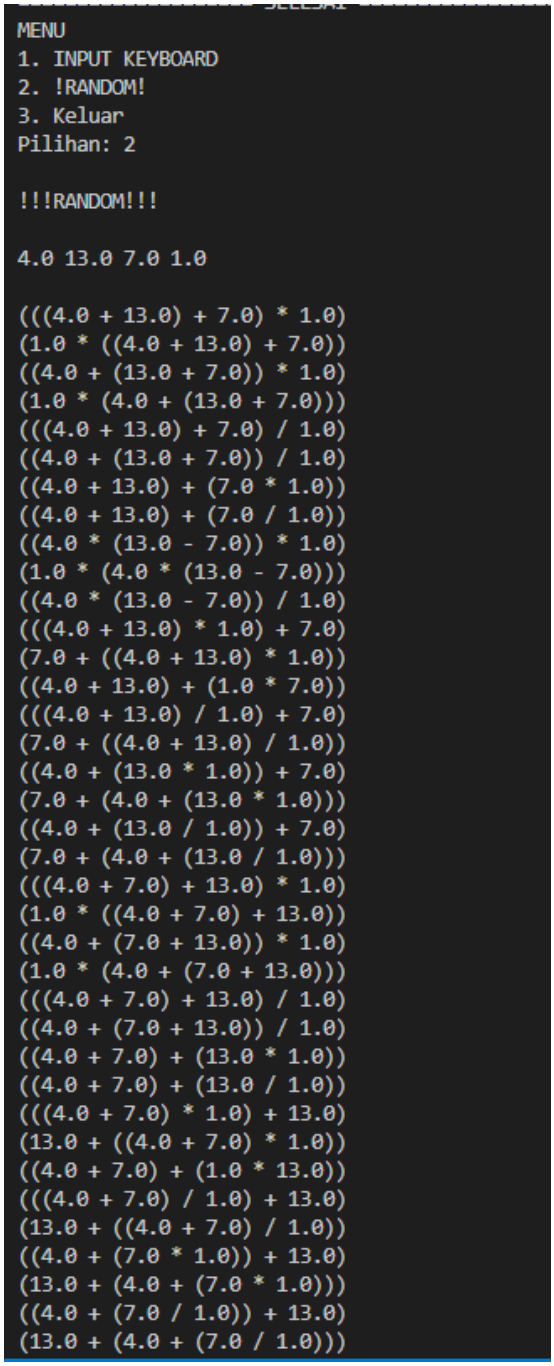
```
MENU
1. INPUT KEYBOARD
2. !RANDOM!
3. Keluar
Pilihan: 2

!!!RANDOM!!!

9.0 10.0 7.0 3.0

((9.0 * (10.0 - 7.0)) - 3.0)
(((9.0 * 3.0) - 10.0) + 7.0)
(7.0 + ((9.0 * 3.0) - 10.0))
((9.0 * 3.0) - (10.0 - 7.0))
(((9.0 * 3.0) + 7.0) - 10.0)
((9.0 * 3.0) + (7.0 - 10.0))
(((10.0 - 9.0) + 7.0) * 3.0)
(3.0 * ((10.0 - 9.0) + 7.0))
((10.0 - (9.0 - 7.0)) * 3.0)
(3.0 * (10.0 - (9.0 - 7.0)))
(7.0 - (10.0 - (9.0 * 3.0)))
(((10.0 + 7.0) - 9.0) * 3.0)
(3.0 * ((10.0 + 7.0) - 9.0))
((10.0 + (7.0 - 9.0)) * 3.0)
(3.0 * (10.0 + (7.0 - 9.0)))
(((10.0 - 7.0) * 9.0) - 3.0)
(7.0 - (10.0 - (3.0 * 9.0)))
(((7.0 - 9.0) + 10.0) * 3.0)
(3.0 * ((7.0 - 9.0) + 10.0))
((7.0 - (9.0 - 10.0)) * 3.0)
(3.0 * (7.0 - (9.0 - 10.0)))
((7.0 + (9.0 * 3.0)) - 10.0)
(((7.0 + 10.0) - 9.0) * 3.0)
(3.0 * ((7.0 + 10.0) - 9.0))
((7.0 + (10.0 - 9.0)) * 3.0)
(3.0 * (7.0 + (10.0 - 9.0)))
((7.0 - 10.0) + (9.0 * 3.0))
((7.0 - 10.0) + (3.0 * 9.0))
((7.0 + (3.0 * 9.0)) - 10.0)
(((3.0 * 9.0) - 10.0) + 7.0)
(7.0 + ((3.0 * 9.0) - 10.0))
((3.0 * 9.0) - (10.0 - 7.0))
(((3.0 * 9.0) + 7.0) - 10.0)
((3.0 * 9.0) + (7.0 - 10.0))
Banyaknya solusi yang ditemukan adalah 34
Time Elapsed : 6 ms

Apakah ingin menyimpan solusi dalam file?
```

output_9_10_7_3.txt

```
9.0 10.0 7.0 3.0
((9.0 * (10.0 - 7.0)) - 3.0)
(((9.0 * 3.0) - 10.0) + 7.0)
(7.0 + ((9.0 * 3.0) - 10.0))
((9.0 * 3.0) - (10.0 - 7.0))
(((9.0 * 3.0) + 7.0) - 10.0)
((9.0 * 3.0) + (7.0 - 10.0))
(((10.0 - 9.0) + 7.0) * 3.0)
(3.0 * ((10.0 - 9.0) + 7.0))
((10.0 - (9.0 - 7.0)) * 3.0)
(3.0 * (10.0 - (9.0 - 7.0)))
(7.0 - (10.0 - (9.0 * 3.0)))
(((10.0 + 7.0) - 9.0) * 3.0)
(3.0 * ((10.0 + 7.0) - 9.0))
((10.0 + (7.0 - 9.0)) * 3.0)
(3.0 * (10.0 + (7.0 - 9.0)))
(((10.0 - 7.0) * 9.0) - 3.0)
(7.0 - (10.0 - (3.0 * 9.0)))
(((7.0 - 9.0) + 10.0) * 3.0)
(3.0 * ((7.0 - 9.0) + 10.0))
((7.0 - (9.0 - 10.0)) * 3.0)
(3.0 * (7.0 - (9.0 - 10.0)))
((7.0 + (9.0 * 3.0)) - 10.0)
(((7.0 + 10.0) - 9.0) * 3.0)
(3.0 * ((7.0 + 10.0) - 9.0))
((7.0 + (10.0 - 9.0)) * 3.0)
(3.0 * (7.0 + (10.0 - 9.0)))
((7.0 - 10.0) + (9.0 * 3.0))
((7.0 - 10.0) + (3.0 * 9.0))
((7.0 + (3.0 * 9.0)) - 10.0)
(((3.0 * 9.0) - 10.0) + 7.0)
(7.0 + ((3.0 * 9.0) - 10.0))
((3.0 * 9.0) - (10.0 - 7.0))
(((3.0 * 9.0) + 7.0) - 10.0)
((3.0 * 9.0) + (7.0 - 10.0))
Jumlah solusi yang ditemukan adalah 34
```

d. 4 K 7 A

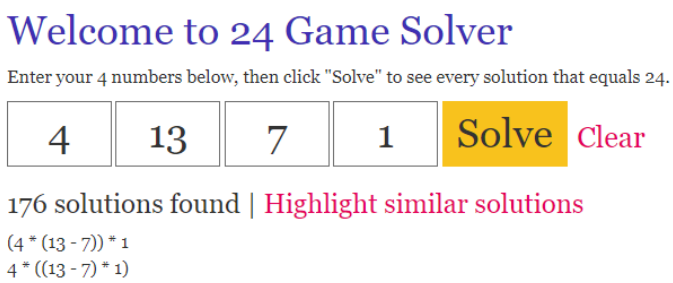
Tampilan pada terminal



.
.
.

```
(((7.0 / 1.0) + 13.0) + 4.0)
(4.0 + ((7.0 / 1.0) + 13.0))
((7.0 / 1.0) + (13.0 + 4.0))
((1.0 * (4.0 + 13.0)) + 7.0)
(7.0 + (1.0 * (4.0 + 13.0)))
(((1.0 * 4.0) + 13.0) + 7.0)
(7.0 + ((1.0 * 4.0) + 13.0))
((1.0 * 4.0) + (13.0 + 7.0))
((1.0 * 4.0) * (13.0 - 7.0))
((1.0 * (4.0 + 7.0)) + 13.0)
(13.0 + (1.0 * (4.0 + 7.0)))
(((1.0 * 4.0) + 7.0) + 13.0)
(13.0 + ((1.0 * 4.0) + 7.0))
((1.0 * 4.0) + (7.0 + 13.0))
((1.0 * (13.0 + 4.0)) + 7.0)
(7.0 + (1.0 * (13.0 + 4.0)))
(((1.0 * 13.0) + 4.0) + 7.0)
(7.0 + ((1.0 * 13.0) + 4.0))
((1.0 * 13.0) + (4.0 + 7.0))
((1.0 * (13.0 + 7.0)) + 4.0)
(4.0 + (1.0 * (13.0 + 7.0)))
((1.0 * (13.0 - 7.0)) * 4.0)
(4.0 * (1.0 * (13.0 - 7.0)))
(4.0 / (1.0 / (13.0 - 7.0)))
(((1.0 * 13.0) + 7.0) + 4.0)
(4.0 + ((1.0 * 13.0) + 7.0))
((1.0 * 13.0) + (7.0 + 4.0))
(((1.0 * 13.0) - 7.0) * 4.0)
(4.0 * ((1.0 * 13.0) - 7.0))
((1.0 * (7.0 + 4.0)) + 13.0)
(13.0 + (1.0 * (7.0 + 4.0)))
(((1.0 * 7.0) + 4.0) + 13.0)
(13.0 + ((1.0 * 7.0) + 4.0))
((1.0 * 7.0) + (4.0 + 13.0))
((1.0 * (7.0 + 13.0)) + 4.0)
(4.0 + (1.0 * (7.0 + 13.0)))
(((1.0 * 7.0) + 13.0) + 4.0)
(4.0 + ((1.0 * 7.0) + 13.0))
((1.0 * 7.0) + (13.0 + 4.0))
Banyaknya solusi yang ditemukan adalah 212
Time Elapsed : 5 ms

Apakah ingin menyimpan solusi dalam file?
1.Ya
2.Tidak
Pilihan: 1
Masukkan nama file yang ingin dibuat (dalam .txt): output_4_K_7_A.txt
File created: output_4_K_7_A.txt
```

Hasil pada 24solver.us



e. 3 5 J 4

Tampilan pada terminal

```
MENU
1. INPUT KEYBOARD
2. !RANDOM!
3. Keluar
Pilihan: 2

!!!RANDOM!!!

3.0 5.0 11.0 4.0

((3.0 * 11.0) - (5.0 + 4.0))
(((3.0 * 11.0) - 5.0) - 4.0)
((3.0 * 11.0) - (4.0 + 5.0))
(((3.0 * 11.0) - 4.0) - 5.0)
(((3.0 + 4.0) * 5.0) - 11.0)
((5.0 * (3.0 + 4.0)) - 11.0)
((5.0 * (4.0 + 3.0)) - 11.0)
((11.0 * 3.0) - (5.0 + 4.0))
(((11.0 * 3.0) - 5.0) - 4.0)
((11.0 * 3.0) - (4.0 + 5.0))
(((11.0 * 3.0) - 4.0) - 5.0)
(((4.0 + 3.0) * 5.0) - 11.0)
Banyaknya solusi yang ditemukan adalah 12
Time Elapsed : 6 ms

Apakah ingin menyimpan solusi dalam file?
1.Ya
2.Tidak
Pilihan: 1
Masukkan nama file yang ingin dibuat (dalam .txt): output_3_5_J_4.txt
File created: output_3_5_J_4.txt

================== SELESAI ==================
```

output_3_5_J_4.txt

```
3.0 5.0 11.0 4.0
((3.0 * 11.0) - (5.0 + 4.0))
(((3.0 * 11.0) - 5.0) - 4.0)
((3.0 * 11.0) - (4.0 + 5.0))
(((3.0 * 11.0) - 4.0) - 5.0)
(((3.0 + 4.0) * 5.0) - 11.0)
((5.0 * (3.0 + 4.0)) - 11.0)
((5.0 * (4.0 + 3.0)) - 11.0)
((11.0 * 3.0) - (5.0 + 4.0))
(((11.0 * 3.0) - 5.0) - 4.0)
((11.0 * 3.0) - (4.0 + 5.0))
(((11.0 * 3.0) - 4.0) - 5.0)
(((4.0 + 3.0) * 5.0) - 11.0)
Jumlah solusi yang ditemukan adalah 12
```

f. Q 7 5 4

Tampilan pada terminal

```
MENU
1. INPUT KEYBOARD
2. !RANDOM!
3. Keluar
Pilihan: 2

!!!RANDOM!!!

12.0 7.0 5.0 4.0

((12.0 / (7.0 - 5.0)) * 4.0)
(4.0 * (12.0 / (7.0 - 5.0)))
((12.0 * 4.0) / (7.0 - 5.0))
(4.0 / ((7.0 - 5.0) / 12.0))
(12.0 / ((7.0 - 5.0) / 4.0))
(((5.0 - 7.0) + 4.0) * 12.0)
(12.0 * ((5.0 - 7.0) + 4.0))
((5.0 - (7.0 - 4.0)) * 12.0)
(12.0 * (5.0 - (7.0 - 4.0)))
(((5.0 + 4.0) - 7.0) * 12.0)
(12.0 * ((5.0 + 4.0) - 7.0))
((5.0 + (4.0 - 7.0)) * 12.0)
(12.0 * (5.0 + (4.0 - 7.0)))
((4.0 * 12.0) / (7.0 - 5.0))
(((4.0 - 7.0) + 5.0) * 12.0)
(12.0 * ((4.0 - 7.0) + 5.0))
((4.0 - (7.0 - 5.0)) * 12.0)
(12.0 * (4.0 - (7.0 - 5.0)))
((4.0 / (7.0 - 5.0)) * 12.0)
(12.0 * (4.0 / (7.0 - 5.0)))
(((4.0 + 5.0) - 7.0) * 12.0)
(12.0 * ((4.0 + 5.0) - 7.0))
((4.0 + (5.0 - 7.0)) * 12.0)
(12.0 * (4.0 + (5.0 - 7.0)))
Banyaknya solusi yang ditemukan adalah 24
Time Elapsed : 4 ms

Apakah ingin menyimpan solusi dalam file?
1.Ya
2.Tidak
Pilihan: 1
Masukkan nama file yang ingin dibuat (dalam .txt): output_Q_7_5_4.txt
File created: output_Q_7_5_4.txt
```

output_Q_7_5_4.txt

```
12.0 7.0 5.0 4.0
((12.0 / (7.0 - 5.0)) * 4.0)
(4.0 * (12.0 / (7.0 - 5.0)))
((12.0 * 4.0) / (7.0 - 5.0))
(4.0 / ((7.0 - 5.0) / 12.0))
(12.0 / ((7.0 - 5.0) / 4.0))
(((5.0 - 7.0) + 4.0) * 12.0)
(12.0 * ((5.0 - 7.0) + 4.0))
((5.0 - (7.0 - 4.0)) * 12.0)
(12.0 * (5.0 - (7.0 - 4.0)))
(((5.0 + 4.0) - 7.0) * 12.0)
(12.0 * ((5.0 + 4.0) - 7.0))
((5.0 + (4.0 - 7.0)) * 12.0)
(12.0 * (5.0 + (4.0 - 7.0)))
((4.0 * 12.0) / (7.0 - 5.0))
(((4.0 - 7.0) + 5.0) * 12.0)
(12.0 * ((4.0 - 7.0) + 5.0))
((4.0 - (7.0 - 5.0)) * 12.0)
(12.0 * (4.0 - (7.0 - 5.0)))
((4.0 / (7.0 - 5.0)) * 12.0)
(12.0 * (4.0 / (7.0 - 5.0)))
(((4.0 + 5.0) - 7.0) * 12.0)
(12.0 * ((4.0 + 5.0) - 7.0))
((4.0 + (5.0 - 7.0)) * 12.0)
(12.0 * (4.0 + (5.0 - 7.0)))
Jumlah solusi yang ditemukan adalah 24
```

BAB V

TABEL

Poin	YA	TIDAK
1. Program berhasil dikompilasi tanpa kesalahan	✓	
2. Program berhasil running	✓	
3. Program dapat membaca input/generate sendiri dan memberikan luaran	✓	
4. Solusi yang diberikan program memenuhi tujuan (berhasil mencapai 24)	✓	
5. Program dapat menyimpan solusi dalam file teks	✓	

LINK REPOSITORY

https://github.com/maximatey/Tucil1_13521061