LAPORAN TUGAS KECIL 1
STRATEGI ALGORITMA

SALOMO REINHART GREGORY MANALU
13521063

PROGRAM STUDI TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
2023

BAB I

ALGORITMA BRUTE FORCE

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.

Brute force adalah sebuah pendekatan yang lempang (straight forward) untuk memecahkan suatu masalah, biasanya didasarkan pada pernyataan masalah (problem statement) dan definisi konsep yang dilibatkan. Algoritma brute force memecahkan masalah dengan sangat sederhana, langsung dan dengan cara yang jelas.

1. Algoritma brute force umumnya tidak cerdas, karena ia membutuhkan jumlah langkah yang besar dalam penyelesaiannya. Kadang-kadang algoritma brute force disebut juga algoritma naif (naïve algorithm).
2. Algoritma brute force seringkali merupakan pilihan yang kurang disukai karena ketidakmangkusannya itu, tetapi dengan mencari pola-pola yang mendasar, keteraturan, atau trik-trik khusus, biasanya akan membantu kita menemukan algoritma yang lebih cerdas dan lebih mangkus.
3. Untuk masalah yang ukurannya kecil, kesederhanaan brute force biasanya lebih diperhitungkan daripada ketidakmangkusannya. Algoritma brute force sering digunakan sebagai basis bila membandingkan beberapa alternatif algoritma yang mangkus.
4. Algoritma brute force seringkali lebih mudah diimplementasikan daripada algoritma yang lebih canggih, dan karena kesederhanaannya, kadang-kadang algoritma brute force dapat lebih mangkus (ditinjau dari segi implementasi).

BAB II

SOURCE CODE

solver.cpp

src > solver.cpp > ...

```cpp
#include <iostream>
#include <string>
#include <fstream>
#include <time.h>
using namespace std;

int inputConvertCharToInt(char input) {
    if (input == 'A') {
        return 1;
    } else if (input == 'J') {
        return 11;
    } else if (input == 'Q') {
        return 12;
    } else if (input == 'K') {
        return 13;
    } else if (input == '10') {
        return 10;
    } else {
        return (int)input-48 ;
    }


}
```

```cpp
void printArray(int array[4]) {
    int i;

    for (i = 0; i < 4; i++) {
        cout << array[i];
        cout << " ";
    }
}

string randomGen(char a, char b, char c, char d) {
    string input;
    srand(time(0));

    char array[13] = {'A','2','3','4','5','6','7','8','9','J','Q','K'};
    int inda = rand() % 12;
    int indb = rand() % 12;
    int indc = rand() % 12;
    int indd = rand() % 12;

    a = array[inda];
    b = array[indb];
    c = array[indc];
    d = array[indd];

    cout << a; cout << " ";
    cout << b; cout << " ";
    cout << c; cout << " ";
    cout << d; cout << endl;

```

```cpp
    input = input + a + " ";
    input = input + b + " ";
    input = input + c + " ";
    input = input + d;

    cout << input;
    return input;
}
```

```cpp
bool valid(string input) {
    int i;

    while (input[i] != ' ') {
        i++;
        if (input[i] != 'A' && input[i] != '2' && input[i] != '3' && input[i] != '4' && input[i] != '5' && input[i] != '6' &&
        input[i] != '7' && input[i] != '8' && input[i] != '9' && input[i] != 'J' && input[i] != 'Q' && input[i] != 'K') {
            if (input[i] == '1' && input[i+1] == 0) {
                return true;
            } else {
                return false;
            }
        }
    }
}

void inputIntToArray (int input, int i, int array[4]) {
    array[i] = input;
}
```

```
int charToOps(char op, int num1, int num2) {
    if (op == '+') {
        return num1 + num2;
    } else if (op == '-') {
        return num1 - num2;
    } else if (op == '*') {
        return num1 * num2;
    } else {
        return num1 / num2;
    }
}
```

```
void permutation(int input[4]) {
    int i,j,k,l;
    int n;
    int temp[4];
    int x,y,z;
    char ops[4] = {'+', '-', '*', '/'};
    int count = 0;
    int num1, num2, num3, num4, num5, p;

    for (n = 0; n < 4; n++) {
        temp[n] = input[n];
    }

    for (i = 0; i < 4; i++) {
        for (j = 0; j < 4; j++) {
            for (k = 0; k < 4; k++) {
                for (l = 0; l < 4; l++) {
                    if (l != k && k != j && j != i && i != k && l != j && k != i && l != i) {
                        temp[0] = input[i];
                        temp[1] = input[j];
                        temp[2] = input[k];
                        temp[3] = input[l];

                        for (x = 0; x < 4; x++) {
                            for (y = 0; y < 4; y++) {
                                for (z = 0; z < 4; z++) {
                                    num1 = charToOps(ops[y], temp[1], temp[2]);
                                    num1 = charToOps(ops[z], num1, temp[3]);
                                    num1 = charToOps(ops[x], num1, temp[0]);
```

```

                                    num2 = charToOps(ops[z], temp[2], temp[3]);
                                    num2 = charToOps(ops[y], num2, temp[1]);
                                    num2 = charToOps(ops[x], num2, temp[0]);

                                    num3 = charToOps(ops[x], temp[0], temp[1]);
                                    num3 = charToOps(ops[y], num3, temp[2]);
                                    num3 = charToOps(ops[z], num3, temp[3]);

                                    num4 = charToOps(ops[y], temp[1], temp[2]);
                                    num4 = charToOps(ops[x], num4, temp[0]);
                                    num4 = charToOps(ops[z], num4, temp[3]);

                                    if (num1 == 24) {
                                        count = count + 1;
                                        hasil += temp[0] + " " + ops[x];
                                        hasil += " (";
                                        hasil += temp[1] + " " + ops[y];
                                        hasil += " (" + temp[2];
                                        hasil += " ";
                                        hasil += ops[z] + " " + temp[3];
                                        hasil += "))";
                                        hasil = hasil + '\n';
                                    }

                                    if (num2 == 24) {
                                        count = count + 1;
                                        hasil += temp[0] + " " + ops[x];
                                        hasil += " ((";
```

```
                                    hasil += temp[1] + " " + ops[y];
                                    hasil += " )" + temp[2];
                                    hasil += " ";
                                    hasil += ops[z] + " " + temp[3];
                                    hasil += ")";
                                    hasil = hasil + '\n';
                                }

                                if (num3 == 24) {
                                    count = count + 1;
                                    hasil += " ((";
                                    hasil += temp[0] + " " + ops[x] + temp[1];
                                    hasil += ") ";
                                    hasil += " " + ops[y];
                                    hasil += " )" + temp[2];
                                    hasil += " ";
                                    hasil += ops[z] + " " + temp[3];
                                    hasil = hasil + '\n';
                                }

                                if (num4 == 24) {
                                    count = count + 1;
                                    count = count + 1;
                                    hasil += " (";
                                    hasil += temp[0] + " " + ops[x];
                                    hasil += "( ";
                                    hasil += temp[1];
                                    hasil += " " + ops[y];
                                    hasil += " ))" + temp[2];
                                    hasil += " ";
                                    hasil += ops[z] + " " + temp[3];
                                    hasil = hasil + '\n';
                                }
                            }
                        }
                    }
                }
            }
        }
    }
```

```
                    }
                }
            }
        }
    }
}

for (i = 0; i < 4; i++) {
    for (j = 0; j < 4; j++) {
        for (k = 0; k < 4; k++) {
            for (l = 0; l < 4; l++) {
                if (l != k && k != j && j != i && i != k && l != j && k != i && l != i) {
                    temp[0] = input[i];
                    temp[1] = input[j];
                    temp[2] = input[k];
                    temp[3] = input[l];

                    for (x = 0; x < 4; x++) {
                        for (y = 0; y < 4; y++) {
                            for (z = 0; z < 4; z++) {
                                num1 = charToOps(ops[y], temp[1], temp[2]);
                                num1 = charToOps(ops[z], num1, temp[3]);
                                num1 = charToOps(ops[x], num1, temp[0]);

                                num2 = charToOps(ops[z], temp[2], temp[3]);
                                num2 = charToOps(ops[y], num2, temp[1]);
                                num2 = charToOps(ops[x], num2, temp[0]);
```

```
            num3 = charToOps(ops[x], temp[0], temp[1]);
            num3 = charToOps(ops[y], num3, temp[2]);
            num3 = charToOps(ops[z], num3, temp[3]);

            num4 = charToOps(ops[y], temp[1], temp[2]);
            num4 = charToOps(ops[x], num4, temp[0]);
            num4 = charToOps(ops[z], num4, temp[3]);

            if (num1 == 24) {
                count = count + 1;
                cout << temp[0] << " " << ops[x] << " (" << temp[1] << " " << ops[y] << " (" << temp[2] << " " << ops[z] << " " << temp[3] << "))";
                cout << endl;
            }

            if (num2 == 24) {
                count = count + 1;
                cout << temp[0] << " " << ops[x] << " ((" << temp[1] << " " << ops[y] << " " << temp[2] << ") " << ops[z] << " " << temp[3] << ")";
                cout << endl;
            }

            if (num3 == 24) {
                count = count + 1;
                cout << "(("<< temp[0] << " " << ops[x] << " " << temp[1] << ") " << ops[y] << " " << temp[2] << ")) " << ops[z] << " " << temp[3];
                cout << endl;
            }

            if (num4 == 24) {
                count = count + 1;
                cout << "("<< temp[0] << " " << ops[x] << " (" << temp[1] << " " << ops[y] << " " << temp[2] << ")) " << ops[z] << " " << temp[3];
```

```
                                cout << "(("<< temp[0] << " " << ops[x] << " " << temp[1] << ") " << ops[y] << " " << temp[2] << ")) " << op
                                cout << endl;
                            }

                            if (num4 == 24) {
                                count = count + 1;
                                cout << "("<< temp[0] << " " << ops[x] << " (" << temp[1] << " " << ops[y] << " " << temp[2] << ")) " << ops
                                cout << endl;
                            }
                        }
                    }
                }
            }
        }
    }

    cout << endl;
    cout << count;
    cout << " Solutions Found";
}
```

main.cpp

solver.cpp U | main.cpp U | README.md

src > main.cpp > main()

```cpp
#include <iostream>
#include <fstream>
#include "solver.cpp"
using namespace std;

int main() {
    char a[2], b[2], c[2], d[2];
    char e,f,g,h;
    string str;
    char ans;
    int num1, num2, num3, num4;
    int input[4];
    int i;
    time_t start, end;
    time(&start);

    cout << "-------------------- Game 24 Solver --------------------";
    cout << endl;
    cout << "Silahkan Masukkan angka. ";
    cout << "Apakah ingin menggunakan Random Generator?(y/n) ";
    scanf("%c", &ans);
    cout << endl;

    ofstream fileSave;
    fileSave.open("testCase.txt");

    if (ans == 'y') {
        randomGen(e,f,g,h);
        num1 = inputConvertCharToInt(e);
        num2 = inputConvertCharToInt(f);
        num3 = inputConvertCharToInt(g);
        num4 = inputConvertCharToInt(h);

        int input[4] = {num1, num2, num3, num4};

    } else {
        cout << "Masukkan 4 angka X X X X : ";
```

```cpp
        scanf("%c %c %c %c", &a, &b, &c, &d);
        num1 = inputConvertCharToInt(*a);
        num2 = inputConvertCharToInt(*b);
        num3 = inputConvertCharToInt(*c);
        num4 = inputConvertCharToInt(*d);


        int input[4] = {num1, num2, num3, num4};
    }

    permutation(input);
    time(&end);
    cout << endl;
    cout << "Execution Time = " << end - start;
    cout << endl;
    cout << "Apakah ingin menyimpan hasil?(y/n) ";
    char answer;
    scanf("%c", &answer);

    if (answer == 'y') {
        string file;
        string nameFile = ".txt";

        cout << "Please, enter the name for your output file: ";
        getline(cin, file);
        file += ".txt";
        string path = "C:/Users/Asus/OneDrive - Institut Teknologi Bandung/Documents/GitHub/Tucil1_13521063/test/";
        path += file;

        fileSave.open(path);
        cout << input;
        fileSave.close();
    } else {
        cout << "TERIMA KASIH";
        exit;
    }
```

BAB III

INPUT DAN OUTPUT PROGRAM

Test Case (1)

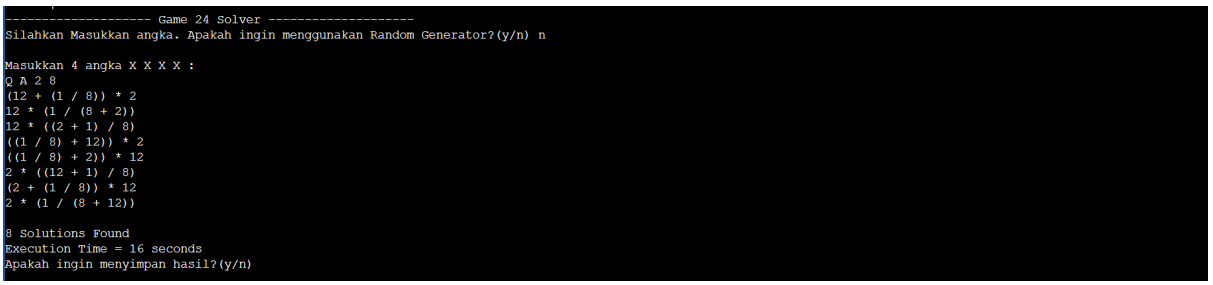
```
Silahkan Masukkan angka. Apakah ingin menggunakan Random Generator?(y/n) n

Masukkan 4 angka X X X X :
2 4 7 6
((2 - 5) + 7)) * 6
((2 + 7) - 5)) * 6
(2 + (7 - 5)) * 6
2 * (7 - (5 * 6))
(2 * (7 - 5)) * 6
((2 * 6) + 5)) + 7
((2 * 6) + 7)) + 5
2 * ((6 * 7) - 5)
(5 - (2 + 7)) * 6
5 + (2 * (6 + 7))
(5 + (2 * 6)) + 7
5 + ((7 + 2) * 6)
(5 - (7 + 2)) * 6
5 + ((7 + 6) * 2)
5 + (6 * (2 + 7))
(5 + (6 * 2)) + 7
((7 + 2) - 5)) * 6
(7 + (2 - 5)) * 6
7 + (2 * (6 + 5))
(7 + (2 * 6)) + 5
7 + ((5 + 2) * 6)
((7 - 5) + 2)) * 6
((7 - 5) * 2)) * 6
7 + ((5 + 6) * 2)
((7 - 5) * 6)) * 2
7 + (6 * (2 + 5))
(7 + (6 * 2)) + 5
((6 * 2) + 5)) + 7
6 * (2 - (5 + 7))
((6 * 2) + 7)) + 5
6 * (2 + (7 - 5))
6 * ((2 + 7) - 5)
6 * ((2 * 7) - 5)
```

```
5 + ((7 + 6) * 2)
5 + (6 * (2 + 7))
(5 + (6 * 2)) + 7
((7 + 2) - 5)) * 6
(7 + (2 - 5)) * 6
7 + (2 * (6 + 5))
(7 + (2 * 6)) + 5
7 + ((5 + 2) * 6)
((7 - 5) + 2)) * 6
((7 - 5) * 2)) * 6
7 + ((5 + 6) * 2)
((7 - 5) * 6)) * 2
7 + (6 * (2 + 5))
(7 + (6 * 2)) + 5
((6 * 2) + 5)) + 7
6 * (2 - (5 + 7))
((6 * 2) + 7)) + 5
6 * (2 + (7 - 5))
6 * ((2 + 7) - 5)
6 * ((2 * 7) - 5)
6 * ((5 - 2) + 7)
6 * ((5 - 7) + 2)
6 * (7 + (2 - 5))
6 * ((7 + 2) - 5)
6 * (7 - (5 + 2))
6 * (7 - (5 * 2))
(6 * (7 - 5)) * 2

40 Solutions Found
Execution Time = 11 seconds
Apakah ingin menyimpan hasil?(y/n)
```

Test Case (2)


---------- Game 24 Solver ----------
Silahkan Masukkan angka. Apakah ingin menggunakan Random Generator?(y/n) n
Masukkan 4 angka X X X X :
Q A 2 8
(12 + (1 / 8)) * 2
12 * (1 / (8 + 2))
12 * ((2 + 1) / 8)
((1 / 8) + 12)) * 2
((1 / 8) + 2)) * 12
2 * ((12 + 1) / 8)
(2 + (1 / 8)) * 12
2 * (1 / (8 + 12))
8 Solutions Found
Execution Time = 16 seconds
Apakah ingin menyimpan hasil?(y/n)

Test Case (3)

```
-------------------- Game 24 Solver --------------------
Silahkan Masukkan angka. Apakah ingin menggunakan Random Generator?(y/n) y

8 3 5 6
8 * ((3 + 5) / 6)
8 * ((3 / 5) + 6)
((8 * 3) / 5)) * 6
8 * ((3 * 6) - 5)
8 * (3 * (6 / 5))
8 * ((3 * 6) / 5)
8 * ((3 / 6) + 5)
((8 + 5) / 3)) * 6
8 * ((5 / 3) * 6)
((8 / 5) + 3)) * 6
(8 + (5 / 6)) * 3
8 * (5 + (6 / 3))
8 * (5 / (6 + 3))
8 * ((5 / 6) * 3)
8 * (6 * (3 / 5))
8 * (6 + (5 / 3))
8 * (6 - (5 * 3))
(8 * (6 - 5)) * 3
8 * (6 / (5 * 3))
(8 * (6 / 5)) * 3
(3 + (8 / 5)) * 6
3 * ((8 + 5) / 6)
((3 * 8) / 5)) * 6
(3 / (8 + 5)) * 6
3 * ((8 * 6) - 5)
3 * ((8 * 6) / 5)
(3 / (5 + 8)) * 6
(3 + (5 / 6)) * 8
3 * (5 / (6 + 8))
(3 / (5 + 6)) * 8
3 * (6 - (5 * 8))
(3 * (6 - 5)) * 8
3 * (6 / (5 * 8))
```

```
(3 * (6 / 5)) * 8
(3 / (6 + 5)) * 8
((5 + 8) / 3)) * 6
(5 / (8 * 3)) * 6
(5 / (3 * 8)) * 6
(5 / (3 * 6)) * 8
((5 / 6) + 8)) * 3
((5 + 6) / 3)) * 8
((5 / 6) + 3)) * 8
(5 / (6 * 3)) * 8
6 * (8 * (3 / 5))
6 * (8 + (5 / 3))
6 * (8 / (5 + 3))
6 * ((3 + 8) / 5)
6 * (3 * (8 / 5))
6 * ((3 / 8) + 5)
6 * ((3 / 5) + 8)
((6 * 3) / 5)) * 8
((6 - 5) * 8)) * 3
6 * (5 + (8 / 3))
6 * ((5 / 8) * 3)
((6 / 5) * 8)) * 3
((6 + 5) / 3)) * 8
((6 - 5) * 3)) * 8
6 * ((5 / 3) * 8)
((6 / 5) * 3)) * 8

60 Solutions Found
Execution Time = 2 seconds
Apakah ingin menyimpan hasil?(y/n)
```

Test Case (4)

```
---------- Game 24 Solver ----------
Silahkan Masukkan angka. Apakah ingin menggunakan Random Generator?(y/n) y

A J Q 3
1 + ((10 - 11) * 3)
1 * (10 + (11 + 3))
1 * ((10 + 11) + 3)
((1 * 10) + 11)) + 3
(1 * (10 + 11)) + 3
1 / (10 + (11 + 3))
1 / ((10 + 11) + 3)
(1 / (10 + 11)) + 3
1 + ((10 - 3) * 11)
1 * (10 + (3 + 11))
1 * ((10 + 3) + 11)
((1 * 10) + 3)) + 11
(1 * (10 + 3)) + 11
1 / (10 + (3 + 11))
1 / ((10 + 3) + 11)
(1 / (10 + 3)) + 11
1 * (11 + (10 + 3))
1 * ((11 + 10) + 3)
((1 * 11) + 10)) + 3
(1 * (11 + 10)) + 3
1 / (11 + (10 + 3))
1 / ((11 + 10) + 3)
(1 / (11 + 10)) + 3
1 + (11 * (3 - 10))
(1 + (11 * 3)) - 10
1 * (11 + (3 + 10))
1 * ((11 + 3) + 10)
((1 * 11) + 3)) + 10
(1 * (11 + 3)) + 10
1 / (11 + (3 + 10))
1 / ((11 + 3) + 10)
(1 / (11 + 3)) + 10
1 * (3 + (10 + 11))
```

```
1 * (11 + (3 + 10))
1 * ((11 + 3) + 10)
((1 * 11) + 3)) + 10
(1 * (11 + 3)) + 10
1 / (11 + (3 + 10))
1 / ((11 + 3) + 10)
(1 / (11 + 3)) + 10
1 * (3 + (10 + 11))
1 * ((3 + 10) + 11)
((1 * 3) + 10)) + 11
(1 * (3 + 10)) + 11
1 / (3 + (10 + 11))
1 / ((3 + 10) + 11)
(1 / (3 + 10)) + 11
1 + (3 * (11 - 10))
(1 + (3 * 11)) - 10
1 * (3 + (11 + 10))
1 * ((3 + 11) + 10)
((1 * 3) + 11)) + 10
(1 * (3 + 11)) + 10
1 / (3 + (11 + 10))
1 / ((3 + 11) + 10)
(1 / (3 + 11)) + 10
10 + (1 * (11 + 3))
10 + ((1 * 11) + 3)
(10 + (1 * 11)) + 3
10 + ((1 / 11) + 3)
10 - ((1 + 11) * 3)
((10 * 1) + 11)) + 3
((10 / 1) + 11)) + 3
10 + (1 * (3 + 11))
10 + ((1 * 3) + 11)
(10 + (1 * 3)) + 11
10 + ((1 / 3) + 11)
10 - ((1 + 3) * 11)
((10 * 1) + 3)) + 11
((10 / 1) + 3)) + 11
```

```
10 + ((11 + 1) * 3)
10 + (11 * (1 + 3))
((10 + 11) * 1)) + 3
(10 + (11 * 1)) + 3
10 + (11 / (1 + 3))
((10 + 11) / 1)) + 3
(10 + (11 / 1)) + 3
10 + (11 + (3 * 1))
10 + ((11 + 3) * 1)
((10 + 11) + 3)) * 1
(10 + (11 + 3)) * 1
10 + (11 + (3 / 1))
10 + ((11 + 3) / 1)
((10 + 11) + 3)) / 1
(10 + (11 + 3)) / 1
10 - (11 * (3 + 1))
(10 - (11 * 3)) + 1
10 + ((3 + 1) * 11)
10 + (3 * (1 + 11))
((10 + 3) * 1)) + 11
(10 + (3 * 1)) + 11
10 + (3 / (1 + 11))
((10 + 3) / 1)) + 11
(10 + (3 / 1)) + 11
10 + (3 + (11 * 1))
10 + ((3 + 11) * 1)
((10 + 3) + 11)) * 1
(10 + (3 + 11)) * 1
10 + (3 + (11 / 1))
10 + ((3 + 11) / 1)
((10 + 3) + 11)) / 1
(10 + (3 + 11)) / 1
10 - (3 * (11 + 1))
(10 - (3 * 11)) + 1
11 + (1 * (10 + 3))
11 + ((1 * 10) + 3)
(11 + (1 * 10)) + 3
```

```
11 + ((1 / 3) + 10)
((11 * 1) + 3)) + 10
((11 / 1) + 3)) + 10
11 + ((10 + 1) * 3)
11 + (10 * (1 + 3))
((11 + 10) * 1)) + 3
(11 + (10 * 1)) + 3
11 + (10 / (1 + 3))
((11 + 10) / 1)) + 3
(11 + (10 / 1)) + 3
11 + (10 + (3 * 1))
11 + ((10 + 3) * 1)
((11 + 10) + 3)) * 1
(11 + (10 + 3)) * 1
11 + (10 + (3 / 1))
11 + ((10 + 3) / 1)
((11 + 10) + 3)) / 1
(11 + (10 + 3)) / 1
11 + ((3 + 1) * 10)
11 + (3 * (1 + 10))
((11 + 3) * 1)) + 10
(11 + (3 * 1)) + 10
11 + (3 / (1 + 10))
((11 + 3) / 1)) + 10
(11 + (3 / 1)) + 10
((11 * 3) + 1)) - 10
11 + (3 + (10 * 1))
11 + ((3 + 10) * 1)
((11 + 3) + 10)) * 1
(11 + (3 + 10)) * 1
11 + (3 + (10 / 1))
11 + ((3 + 10) / 1)
((11 + 3) + 10)) / 1
(11 + (3 + 10)) / 1
((11 * 3) - 10)) + 1
3 + (1 * (10 + 11))
3 + ((1 * 10) + 11)
```

```
((3 / 1) + 11)) + 10
3 + ((10 + 1) * 11)
3 + (10 * (1 + 11))
((3 + 10) * 1)) + 11
(3 + (10 * 1)) + 11
3 + (10 / (1 + 11))
((3 + 10) / 1)) + 11
(3 + (10 / 1)) + 11
3 + (10 + (11 * 1))
3 + ((10 + 11) * 1)
((3 + 10) + 11)) * 1
(3 + (10 + 11)) * 1
3 + (10 + (11 / 1))
3 + ((10 + 11) / 1)
((3 + 10) + 11)) / 1
(3 + (10 + 11)) / 1
3 + ((11 + 1) * 10)
3 + (11 * (1 + 10))
((3 + 11) * 1)) + 10
(3 + (11 * 1)) + 10
3 + (11 / (1 + 10))
((3 + 11) / 1)) + 10
(3 + (11 / 1)) + 10
((3 * 11) + 1)) - 10
3 + (11 + (10 * 1))
3 + ((11 + 10) * 1)
((3 + 11) + 10)) * 1
(3 + (11 + 10)) * 1
3 + (11 + (10 / 1))
3 + ((11 + 10) / 1)
((3 + 11) + 10)) / 1
(3 + (11 + 10)) / 1
((3 * 11) - 10)) + 1

184 Solutions Found
Execution Time = 4 seconds
Apakah ingin menyimpan hasil?(y/n)
```

Link To Repository : https://github.com/Salomo309/Tucil1_13521063

Poin	Ya	Tidak
Program berhasil dikompilasi tanpa kesalahan	Ya	
Program berhasil running	Ya	
Program dapat membaca input / generate sendiri dan memberikan luaran	Ya	
Solusi yang diberikan program memenuhi (berhasil mencapai 24)	Ya	
Program dapat menyimpan solusi dalam file teks	Ya	