Laporan Tugas Kecil 1 IF2211 Strategi Algoritma
Semester II tahun 2022/2023

Program Pencari Solusi Permainan Kartu 24 dengan Algoritma Brute Force

Johann Christian Kandani – 13521138

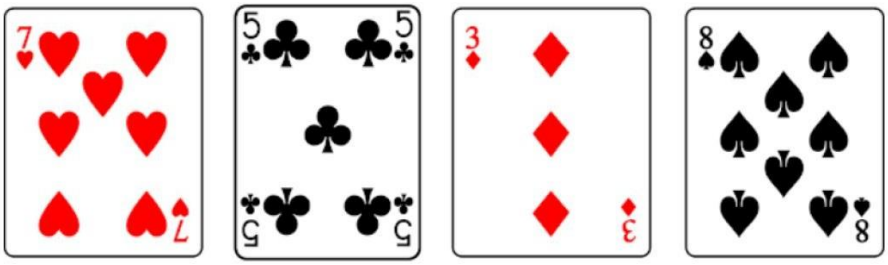
PROGRAM STUDI TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
2023

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. (Paragraf di atas dikutip dari:https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2015-2016/Makalah-2016/MakalahStima-2016-038.pdf).

Laporan ini membahas program yang dapat digunakan untuk mencari solusi dari permainan kartu 24 dengan algoritma *Brute Force*.



Gambar 1. Permainan Kartu 24

BAB 2 : ALGORITMA BRUTE FORCE UNTUK PENCARIAN SOLUSI PERMAINAN KARTU 24

Pencarian seluruh solusi permainan kartu 24 dapat diselesaikan dengan algoritma brute force dengan cara mengiterasi seluruh kemungkinan ekspresi matematis terhadap seluruh permutasi kartu-kartu yang mungkin. Ekspresi matematis suatu permutasi yang menghasilkan nilai 24 (sesuai dengan deskripsi permainan) merupakan salah satu solusi permainan yang valid. Algoritma *brute force* ini diimplementasikan menjadi tiga langkah:

- **Langkah 1 (Permutasi bilangan)**

  Pencarian permutasi kartu-kartu (bilangan-bilangan) yang mungkin. Permutasi didapatkan melalui algoritma brute force yang menggunakan teknik rekursi.

  - **Basis** : Permutasi dari sebuah senarai (list) dengan tepat satu elemen adalah senarai itu sendiri. Langsung kembalikan senarai tersebut.
  - **Rekurens** : Permutasi dari sebuah senarai adalah sebuah elemen dari senarai tersebut yang digabungkan dengan permutasi sisa elemen senarai tersebut. Permutasi yang dihasilkan disimpan ke dalam sebuah *set* terlebih dahulu untuk menghindari adanya permutasi ganda.

- **Langkah 2 (Operasi dasar)**
  Iterasi ekspresi-ekspresi matematis dari sebuah permutasi bilangan-bilangan menggunakan algoritma *brute force*, dengan mencoba seluruh kemungkinan partisi atau segmentasi dari permutasi, kemudian menghitung nilai hasil operasi dasar permainan (+, -, *, /) terhadap kedua segmen tersebut yang telah dilakukan operasi yang sama melalui rekursi.

  - **Basis** : Ekspresi matematis yang mungkin dari permutasi dari sebuah (tepat satu) bilangan adalah bilangan itu sendiri.
  - **Rekurens** : Iterasi partisi-partisi permutasi bilangan, kemudian hasilkan ekspresi-ekspresi matematis berdasarkan partisi melalui rekursi. Ekspresi-ekspresi yang dihasilkan kemudian digabungkan kembali dengan operator dasar (+, -, *, /). Partisi secara implisit merupakan operator kurung (‘()’).

- **Langkah 3 (Evaluasi)**
  Untuk setiap ekspresi yang telah dihasilkan pada langkah 2, periksa apakah nilai yang dihasilkan adalah 24. Untuk setiap ekspresi yang menghasilkan nilai 24, tambahkan ekspresi sebagai salah satu solusi yang mungkin.

Langkah 1 hingga 3 diulangi sebanyak jumlah permutasi yang dihasilkan dari masukan bilangan-bilangan. Jika tidak terdapat solusi, akan dihasilkan pesan tidak ada solusi yang memenuhi.

BAB 3 : IMPLEMENTASI PROGRAM DENGAN BAHASA C++

Algoritma pada Bab 2 diimplementasikan dengan sebuah program C++. Program diimplementasikan secara modular dengan 3 file; main.cpp yang memuat program utama, io.cpp sebagai modul yang memuat fungsi dan prosedur interaksi pengguna, card24.cpp yang memuat fungsi untuk menyelesaikan persoalan kartu 24, termasuk kakas yang digunakan untuk menyelesaikannya.

- ***main.cpp***

main.cpp memuat menu utama program dan pilihan-pilihan pengguna beserta fungsi-fungsi lain seperti pembangkit nilai acak dan perhitungan waktu eksekusi fungsi.

```
cout << "           Selamat datang pada program \"Permainan Kartu 24\" Solver!\n";
cout << "   ------------------------------------------------------------------------\n";
cout << "Program akan menghasilkan solusi-solusi dari masukan 4 buah nilai kartu bridge\n";
cout << "========================================================================\n";

for(;;){
    cout << "\n\nMode program:\n1. Masukan dari keyboard\n2. Masukan dari file\n3. Pembangkit acak program\n4. Keluar\n";
    cout << "Silakan pilih mode program (input berupa angka 1-4): ";
    getline(cin, line);
    option = *(line.begin());
    if(option == '1'){
        cout << "Masukkan 4 angka atau simbol kartu yang masing-masing dipisahkan oleh spasi\n";
        input = userIn();
        nums = parse24(input);
    }
    else if(option == '2'){
        cout << "Pastikan file berada pada directory \"test\"\n";
        cout << "Format file input berupa 4 angka atau simbol kartu yang masing-masing dipisahkan oleh spasi\n";
        input = fileIn();
        nums = parse24(input);
    }
    else if(option == '3'){
        srand(time(0)); // inisialisasi seed random
        randnum = rand()%12 + 1;
        cout << randnum << " ";
```

Gambar 2. main.cpp

- ***io.cpp***

io.cpp memuat fungsi dan prosedur untuk interaksi *input*/*output* pengguna serta parsing dan manipulasi file.

```
#ifndef IO_H
#define IO_H

#include <iostream>
#include <fstream>
#include <string>
#include <vector>
#include <cctype>

using namespace std;

vector<int> parse24(string input);
/* Mengembalikan array dengan 4 elemen yang berupa angka dengan nilai 1-13 jika input valid, */
/* dan mengembalikan empty vector jika input tidak sesuai */
/* input-input dipisahkan oleh spasi, parser mengabaikan input sisa setelah 4 input pertama telah valid */

string userIn();
/* Mengembalikan string berdasarkan input user dari keyboard */

string fileIn();
/* Mengembalikan string berdasarkan input dari file, mengembalikan "FAIL" jika gagal */

void printRes(vector<string> output);
/* I.S. output terdefinisi */
/* F.s. Mencetak elemen-elemen output ke layar yang dipisahkan dengan karakter newline (\n) */

void writeRes(vector<string> output, string fname = "output.out");
/* I.S. output terdefinisi */
/* F.s. Menuliskan output ke sebuah file dengan nama fname yang dipisahkan dengan karakter newline (\n) */

#endif
```

Gambar 3. io.cpp

Fungsi parse() menerima input sebuah string yang kemudian di-parse untuk menghasilkan 4 bilangan bulat 1-13 sebagai nilai dari kartu-kartu. parser akan membaca 4 input pertama yang dipisahkan oleh spasi, dan mengabaikan sisanya jika input tersebut valid. parser akan menghasilkan *vector* kosong jika parse gagal.

Fungsi userIn() menjalankan dan menerima input dari user (keyboard melalui terminal) dan fileIn() menjalankan dan menerima input dari file yang ditentukan user dengan lokasi relatif terhadap lokasi folder “test” pada folder program, file yang tidak valid akan diberi pilihan untuk mengulang input nama file.

Fungsi printRes(vector<string> output) mencetak hasil program pada layar dan writeRes(vector<string> output, string fname) Menuliskan hasil program ke dalam sebuah file yang ditentukan user dengan lokasi relatif terhadap lokasi folder “test” pada folder program input kosong secara *default* digantikan dengan nama file “output.out”.

- ***card24.cpp***

  card24.cpp memuat fungsi untuk menyelesaikan persoalan permainan kartu 24.

```cpp
#include <vector>
#include <string>
#include <set>

using namespace std;

struct expression
{
    float val;
    string expr;
};

vector<string> solve24(vector<int> nums);
/* Mengembalikan ekspresi-ekspresi yang mungkin sebagai solusi dari sebuah kumpulan angka */

vector<vector<int>> permute(vector<int> nums);
/* Mengembalikan permutasi-permutasi yang mungkin dari sebuah kumpulan angka */

vector<expression> count24(vector<int> nums);
/* I.S. nums (permutasi angka-angka) terdefinisi, start dan end terdefinisi */
/* F.s. solusi-solusi baru yang valid ditambahkan ke dalam expressions */
```

Gambar 4. card24.cpp

card24.cpp mendefinisikan tipe bentukan expression yang menyimpan sebuah ekspresi matematis yang dibentuk dan nilai dari ekspresi matematis tersebut dalam bentuk nilai floating point.

Fungsi solve24(vector<int> nums) menerima masukan sebuah kumpulan bilangan sebagai nilai kartu-kartu pada permainan kartu 24 yang kemudian menghasilkan solusi-solusi permainan berupa ekspresi-ekspresi matematis dalam bentuk kumpulan string.

```cpp
vector<string> solve24(vector<int> nums){
    /* KAMUS LOKAL */
    vector<string> solutions;
    vector<expression> expressions;
    vector<vector<int>> permutations;

    vector<vector<int>>::iterator it;
    vector<expression>::iterator jt;

    /* ALGORITMA */
    permutations = permute(nums);
    for(it = permutations.begin(); it != permutations.end(); ++it){
        expressions = count24(*it);
        for(jt = expressions.begin(); jt != expressions.end(); ++jt){
            if(jt->val == 24){
                solutions.push_back(jt->expr);
            }
        }
    }
    return solutions;
}
```

Gambar 5. solve24

Fungsi permute(vector<int> nums) menerima masukan sebuah kumpulan bilangan sebagai nilai kartu-kartu pada permainan kartu 24 yang kemudian menghasilkan permutasi-permutasi bilangan-bilangan tersebut tanpa elemen ganda.

```
vector<vector<int>> permute(vector<int> nums){
    /* KAMUS LOKAL */
    vector<int> subnums;
    vector<int>::iterator n_it;

    set<vector<int>> solutions;

    vector<vector<int>> nextpermute;
    vector<vector<int>>::iterator p_it;

    vector<vector<int>> solutionvec;

    /* ALGORITMA */
    /* BASIS */
    if(nums.size() == 1){
        solutionvec.push_back(vector<int>(nums));
        return solutionvec;
    }
    /* REKURENS */
    else{
        for(n_it = nums.begin(); n_it != nums.end(); ++n_it){
            /* create subset of number to permute */
            subnums.clear();
            subnums.insert(subnums.begin(), nums.begin(), n_it);
            subnums.insert(subnums.end(), n_it+1, nums.end());

            /* find next permutations based on remaining elements */
            nextpermute = permute(subnums);

            /* insert (append) current element not in subset of permuted remaining elements */
            for(p_it = nextpermute.begin(); p_it != nextpermute.end(); ++p_it){
                (*p_it).push_back(*n_it);
                solutions.insert(*p_it);
            }
        }
        solutionvec.assign(solutions.begin(), solutions.end());
        return solutionvec;
    }
}
```

Gambar 6. permute

Fungsi count24(vector<int> nums) menerima masukan sebuah kumpulan bilangan sebagai nilai kartu-kartu pada permainan kartu 24 yang kemudian menghasilkan ekspresi-ekspresi matematis dan nilai dari ekspresi tersebut berdasarkan kumpulan bilangan masukan sesuai dengan urutannya.

```
    /* ALGORITMA */
    /* BASIS */
    if(nums.size() == 1){
        temp.expr = to_string(*(nums.begin()));
        temp.val = *(nums.begin());
        solutions.push_back(temp);
        return solutions;
    }
    /* REKURENS */
    else{
        for(it = nums.begin(); it != nums.end()-1; ++it){
            /* segment */
            subnums.clear();
            subnums.insert(subnums.begin(), nums.begin(), it+1);
            left = count24(subnums);
            subnums.clear();
            subnums.insert(subnums.begin(), it+1, nums.end());
            right = count24(subnums);

            for(l_it = left.begin(); l_it != left.end(); ++l_it){
                for(r_it = right.begin(); r_it != right.end(); ++r_it){
                    /* + */
                    temp.expr = "(" + l_it->expr + " + " + r_it->expr + ")";
                    temp.val = l_it->val + r_it->val;
                    solutions.push_back(temp);
                    /* - */
                    temp.expr = "(" + l_it->expr + " - " + r_it->expr + ")";
                    temp.val = l_it->val - r_it->val;
                    solutions.push_back(temp);
                    /* * */
                    temp.expr = "(" + l_it->expr + " * " + r_it->expr + ")";
                    temp.val = l_it->val * r_it->val;
                    solutions.push_back(temp);
                    /* / */
                    if(r_it->val != 0){
                        temp.expr = "(" + l_it->expr + " / " + r_it->expr + ")";
                        temp.val = l_it->val / r_it->val;
                        solutions.push_back(temp);
                    }
                }
            }
        }
        return solutions;
    }
}
```

Gambar 7. count24

BAB 4 : EKSPERIMEN

Run program

```
           Selamat datang pada program "Permainan Kartu 24" Solver!
   ------------------------------------------------------------------------
Program akan menghasilkan solusi-solusi dari masukan 4 buah nilai kartu bridge
========================================================================


Mode program:
1. Masukan dari keyboard
2. Masukan dari file
3. Pembangkit acak program
4. Keluar
Silakan pilih mode program (input berupa angka 1-4):
```

Gambar 8. Main

Opsi 1:

```
Mode program:
1. Masukan dari keyboard
2. Masukan dari file
3. Pembangkit acak program
4. Keluar
Silakan pilih mode program (input berupa angka 1-4): 1
Masukkan 4 angka atau simbol kartu yang masing-masing dipisahkan oleh spasi
A 1 j 13
Jumlah solusi yang mungkin: 240
(1 - (1 - (11 + 13)))
(1 * (1 * (11 + 13)))
(1 / (1 / (11 + 13)))
(1 - ((1 - 11) - 13))
(1 * ((1 * 11) + 13))
((1 - 1) + (11 + 13))
```

Gambar 9. Opsi 1.1 - solution found (1)

```
((13 + (11 * 1)) / 1)
((13 + (11 / 1)) * 1)
((13 + (11 / 1)) / 1)
(((13 + 11) + 1) - 1)
(((13 + 11) - 1) + 1)
(((13 + 11) * 1) * 1)
(((13 + 11) * 1) / 1)
(((13 + 11) / 1) * 1)
(((13 + 11) / 1) / 1)
Proses telah selesai dijalankan.
(time taken: 2401 microseconds)
Simpan hasil dalam file (Y/N)? n
Program akan kembali ke menu utama
```

Gambar 10. Opsi 1.1 - solution found (2)

```
Mode program:
1. Masukan dari keyboard
2. Masukan dari file
3. Pembangkit acak program
4. Keluar
Silakan pilih mode program (input berupa angka 1-4): 1
Masukkan 4 angka atau simbol kartu yang masing-masing dipisahkan oleh spasi
A 7 J K
Tidak ada solusi yang mungkin
(time taken: 2901 microseconds)
```

Gambar 11. Opsi 1.2 - no solution found

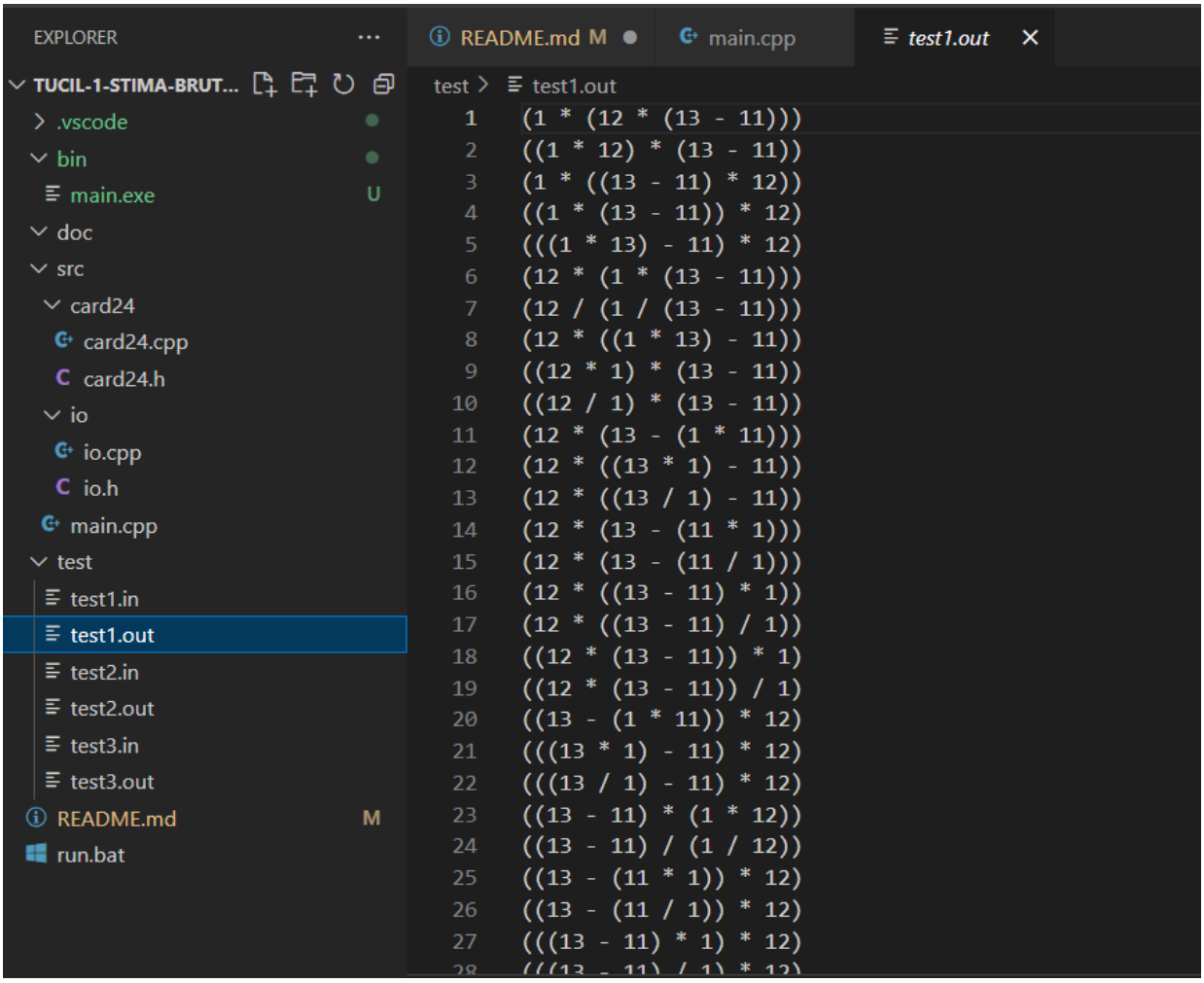
Opsi 2:

```
Mode program:
1. Masukan dari keyboard
2. Masukan dari file
3. Pembangkit acak program
4. Keluar
Silakan pilih mode program (input berupa angka 1-4): 2
Pastikan file berada pada directory "test"
Format file input berupa 4 angka atau simbol kartu yang masing-masing dipisahkan oleh spasi
Masukkan nama file: test1.txt
File tidak dapat dibuka, apakah anda ingin mengulangi input melalui file? (Y/N) y
Masukkan nama file: test1.in
Jumlah solusi yang mungkin: 32
(1 * (12 * (13 - 11)))
((1 * 12) * (13 - 11))
(1 * ((13 - 11) * 12))
((1 * (13 - 11)) * 12)
(((1 * 13) - 11) * 12)
(12 * (1 * (13 - 11)))
(12 / (1 / (13 - 11)))
```

Gambar 12. Opsi 2.1 - input dari file dan menyimpan ke sebuah file (1)

```
((13 - 11) * (1 * 12))
((13 - 11) / (1 / 12))
((13 - (11 * 1)) * 12)
((13 - (11 / 1)) * 12)
(((13 - 11) * 1) * 12)
(((13 - 11) / 1) * 12)
((13 - 11) * (12 * 1))
((13 - 11) * (12 / 1))
(((13 - 11) * 12) * 1)
(((13 - 11) * 12) / 1)
Proses telah selesai dijalankan.
(time taken: 2691 microseconds)
Simpan hasil dalam file (Y/N)? y
Masukkan nama file untuk menyimpan hasil: test1.out
File berhasil disimpan pada folder "test" dengan nama test1.out
Program akan kembali ke menu utama
```

Gambar 13. Opsi 2.1 - input dari file dan menyimpan ke sebuah file (2)



Gambar 14. Opsi 2.1 - input dari file dan menyimpan ke sebuah file (3)

Opsi 3:

```
Mode program:
1. Masukan dari keyboard
2. Masukan dari file
3. Pembangkit acak program
4. Keluar
Silakan pilih mode program (input berupa angka 1-4): 3
5 9 3 1
Jumlah solusi yang mungkin: 68
(1 * ((3 * 5) + 9))
((1 * (3 * 5)) + 9)
(((1 * 3) * 5) + 9)
(1 * ((5 * 3) + 9))
((1 - 5) * (3 - 9))
((1 * (5 * 3)) + 9)
```

Gambar 15. Opsi 3.1 - opsi pembangkit acak (1)

```
(9 + ((5 * 1) * 3))
(9 + ((5 / 1) * 3))
(9 + (5 * (3 * 1)))
(9 + (5 * (3 / 1)))
(9 + ((5 * 3) * 1))
(9 + ((5 * 3) / 1))
((9 + (5 * 3)) * 1)
((9 + (5 * 3)) / 1)
Proses telah selesai dijalankan.
(time taken: 2749 microseconds)
Simpan hasil dalam file (Y/N)?
```

Gambar 16. Opsi 3.1 - opsi pembangkit acak (2)

```
Mode program:
1. Masukan dari keyboard
2. Masukan dari file
3. Pembangkit acak program
4. Keluar
Silakan pilih mode program (input berupa angka 1-4): 3
8 2 1 9
Jumlah solusi yang mungkin: 38
((2 * 8) - (1 - 9))
(((2 * 8) - 1) + 9)
((2 * 8) + (9 - 1))
(((2 * 8) + 9) - 1)
((2 * (9 - 1)) + 8)
```

Gambar 17. Opsi 3.2 - opsi pembangkit acak (1)

```
(9 / ((2 + 1) / 8))
((9 / (2 + 1)) * 8)
(9 + ((2 * 8) - 1))
((9 + (2 * 8)) - 1)
(9 * (8 / (1 + 2)))
((9 * 8) / (1 + 2))
(9 * (8 / (2 + 1)))
(9 + ((8 * 2) - 1))
((9 * 8) / (2 + 1))
((9 + (8 * 2)) - 1)
Proses telah selesai dijalankan.
(time taken: 2960 microseconds)
Simpan hasil dalam file (Y/N)?
```

Gambar 18. Opsi 3.2 - opsi pembangkit acak (2)

Input invalid:

```
Mode program:
1. Masukan dari keyboard
2. Masukan dari file
3. Pembangkit acak program
4. Keluar
Silakan pilih mode program (input berupa angka 1-4): n
Input gagal, mohon hanya input angka dengan nilai dari 1 hingga 4!
```

Gambar 19. Opsi 2.1 - input dari file dan menyimpan ke sebuah file (2)

```
Mode program:
1. Masukan dari keyboard
2. Masukan dari file
3. Pembangkit acak program
4. Keluar
Silakan pilih mode program (input berupa angka 1-4): 1
Masukkan 4 angka atau simbol kartu yang masing-masing dipisahkan oleh spasi
A 1 2 $
Input gagal, mohon pastikan input anda sudah tepat
```

Gambar 20. Opsi 1: 4 input dengan 1 masukan tidak valid

```
Mode program:
1. Masukan dari keyboard
2. Masukan dari file
3. Pembangkit acak program
4. Keluar
Silakan pilih mode program (input berupa angka 1-4): 1
Masukkan 4 angka atau simbol kartu yang masing-masing dipisahkan oleh spasi
A 1 2 12 9 9 9 9
Jumlah solusi yang mungkin: 229
(1 - (1 - (2 * 12)))
(1 * (1 * (2 * 12)))
(1 / (1 / (2 * 12)))
```

Gambar 21. Opsi 1: 4 input dengan tambahan masukan tidak valid

Opsi 4 (exit):

```
Mode program:
1. Masukan dari keyboard
2. Masukan dari file
3. Pembangkit acak program
4. Keluar
Silakan pilih mode program (input berupa angka 1-4): 4
Program telah selesai dijalankan, semoga membantu!
```

Gambar 22. Opsi 4 - exit

LAMPIRAN

Repository Github

https://github.com/Genvictus/Tucil1_13521138

Tabel Spesifikasi

No	Poin	Ya	Tidak
1	Program berhasil dikompilasi tanpa kesalahan	✓	
2	Program berhasil running	✓	
3	Program dapat membaca input / generate sendiri dan memberi luaran	✓	
4	Solusi yang diberikan program memenuhi (berhasil mencapai 24)	✓	
5	Program dapat menyimpan solusi dalam file teks	✓	