

**Tugas Kecil 1 IF2211 Strategi Algoritma**  
**Penyelesaian Permainan Kartu 24 dengan Algoritma**  
**Brute Force**



**Disusun oleh :**  
**13521024 - Ahmad Nadil**

**INSTITUT TEKNOLOGI BANDUNG**  
**2023**

# Daftar Isi

<b>Daftar Isi</b>	<b>1</b>
<b>BAB I</b>	<b>2</b>
1.1 Algoritma Brute Force	2
1.2 Permainan Kartu 24	2
1.3 Algoritma Penyelesaian Permainan Kartu 24 dengan Pendekatan Brute Force	3
<b>BAB II</b>	<b>5</b>
2.1 File main.cpp	5
2.2 File operations.cpp	6
2.3 File cards.cpp	6
2.4 File result.cpp	7
2.5 Library	7
<b>BAB III</b>	<b>8</b>
3.1 Repository Program	8
3.2 Source Code Program	8
3.2.1 main.cpp	8
3.2.2 operations.cpp	13
3.2.3 cards.cpp	14
3.2.4 result.cpp	18
<b>BAB IV</b>	<b>21</b>
4.1 Input dari User	21
4.1.1 A89Q	21
4.1.2 J55Q	22
4.1.3 Q46J	23
4.2 Generate Random	24
4.2.1 8384	24
4.2.2 5QQ4	25
4.2.3 J8J9	26
<b>BAB V</b>	<b>27</b>
<b>REFERENSI</b>	<b>28</b>

# BAB I

## DESKRIPSI MASALAH DAN ALGORITMA

### 1.1 Algoritma Brute Force

Algoritma *brute force* merupakan sebuah pendekatan dari penyelesaian sebuah masalah algoritmik dengan menggunakan pendekatan yang lempeng (*straightforward*) untuk menyelesaikan sebuah persoalan yang telah didefinisikan. Algoritma *brute force* didasarkan pada pernyataan pada persoalan (*problem statement*) dan juga konsep yang dilibatkan oleh persoalan yang sedang dibahas.

Algoritma dengan pendekatan *brute force* biasanya memiliki ciri khas mempunyai konsep penyelesaian yang sederhana, langsung, jelas, dan intuitif. Pendekatan *brute force* seringkali melibatkan enumerasi semua kemungkinan solusi, sebelum akhirnya menghilangkan jawaban yang tidak memenuhi syarat dan mengambil solusi terbaik. Algoritma dengan pendekatan *brute force* dijamin akan menemukan sebuah solusi apabila solusi tersebut ada. Namun, algoritma dengan pendekatan *brute force* seringkali tidak mangkus atau tidak efektif, dengan  $O(n)$  yang lebih buruk dari polinomial.

### 1.2 Permainan Kartu 24

Permainan kartu 24 merupakan 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 simbol (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). Perlu diperhatikan hanyalah nilai kartu yang didapat. 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 (\*), pembagian (/), dan tanda kurung ( ). Setiap kartu harus digunakan tepat sekali dan urutan penggunaannya bebas.

### 1.3 Algoritma Penyelesaian Permainan Kartu 24 dengan Pendekatan Brute Force

Dalam menyelesaikan permasalahan kartu 24 secara algoritmik, penulis menggunakan pendekatan *brute force*. Adapun langkah-langkah penyelesaian permasalahan dalam algoritma dapat dijelaskan secara deskriptif sebagai berikut:

1. Program meminta pengguna untuk melakukan input 4 kartu atau meminta program untuk memilih kartu secara acak (*random*).
2. Program akan mencari semua kemungkinan urutan kartu dengan metode permutasi.

Misalnya, pada urutan kartu 2-3-4-5, akan dicari juga kemungkinan lainnya, seperti 2-4-5-3, 4-5-3-2, 3-2-4-5, dan seterusnya.

3. Setiap urutan kartu yang memungkinkan akan dihitung kemungkinannya untuk mencapai total 24 dengan berbagai kemungkinan operasi sebagai berikut :

$(a \text{ op } b) \text{ op } (c \text{ op } d)$

$a \text{ op } (b \text{ op } (c \text{ op } d))$

$a \text{ op } ((b \text{ op } c) \text{ op } d)$

$(a \text{ op } (b \text{ op } c)) \text{ op } d$

$((a \text{ op } b) \text{ op } c) \text{ op } d$

**op** adalah operasi aritmatika yaitu penjumlahan (+), pengurangan (-), perkalian (\*), dan pembagian (/).

4. Untuk mencari semua kemungkinan operasi yang mungkin, akan dilakukan 3 *nested for loops* (karena terdapat 3 operator aritmatika pada semua kemungkinan operasi yang ada) untuk mencari semua kemungkinan kombinasi operator aritmatika..
5. Lalu untuk bentuk-bentuk operasi yang memungkinkan, akan dilakukan pengecekan apakah dengan kombinasi tertentu dapat menghasilkan jumlah 24. Jika menghasilkan jumlah 24, akan disimpan operasi tersebut dalam sebuah *array of string*. Tetapi sebelum itu, akan dilakukan pengecekan

apakah elemen yang akan dimasukkan sudah berada pada *array* tersebut atau tidak untuk menghindari operasi yang duplikat. Jumlah dari elemen *array of string* akan menjadi jumlah solusi yang mungkin.

6. Keluarkan jumlah solusi serta bentuk solusi yang memungkinkan ke terminal. Program juga dapat menerima opsi dari user jika solusi tersebut ingin disimpan ke dalam bentuk .txt .

## BAB II

### IMPLEMENTASI ALGORITMA DALAM BAHASA C++

Dalam pembuatan program ini, penulis menggunakan bahasa pemrograman C++. Struktur dari program ini terbagi menjadi 4 file, yaitu ***main.cpp***, ***operations.cpp***, ***cards.cpp***, dan ***result.cpp***. Lalu juga terdapat 3 file header untuk masing-masing file fungsi, yaitu ***operations.hpp***, ***cards.hpp***, dan ***result.hpp***.

#### 2.1 File main.cpp

File ini merupakan *driver* utama dari program ini, sehingga tidak terdapat fungsi di dalamnya, hanya berisi menu utama dari program ini serta deklarasi variabel yang akan digunakan.

Variable Name	Description
<code>bool run</code>	Variabel untuk mengatur <i>main loop</i> dari program
<code>bool valid</code>	Variabel sebagai checker apakah input menu program benar
<code>int pilihan</code>	Variabel untuk menyimpan pilihan menu dari <i>user</i>
<code>vector&lt;string&gt; kartu_input</code>	Array of string untuk menyimpan input kartu dari pengguna
<code>vector&lt;string&gt; hasil</code>	Array of string untuk menyimpan semua kemungkinan operasi yang dapat menghasilkan 24
<code>string arr_kartu[]</code>	Array of string untuk menyimpan semua bentuk kartu remi
<code>double nilai_kartu[4]</code>	Variabel untuk menyimpan nilai kartu yang telah dikonversi dari bentuk stringnya
<code>double runtime</code>	Variabel untuk menyimpan waktu eksekusi dari program

## 2.2 File operations.cpp

File ini berisi operasi-operasi dasar yang digunakan untuk membantu perhitungan pencarian kemungkinan solusi serta untuk mengubah tipe data.

Methods	Description
<code>string inttostr (int num)</code>	Menerima sebuah integer dan mengembalikan nilai integer tersebut dalam bentuk string.
<code>string inttoop (int op)</code>	Menerima sebuah integer dan mengembalikan nilai operator tersebut dalam bentuk string.
<code>double operasi (double a, double b, double c)</code>	Menerima dua buah bilangan dan sebuah operator dalam bentuk integer, dan mengembalikan hasil operasi antara dua buah bilangan tersebut.

## 2.3 File cards.cpp

File ini berisi fungsi-fungsi yang berkaitan dengan manipulasi data kartu dan perhitungan banyak kemungkinan operasi yang dapat mencapai jumlah 24.

Methods	Description
<code>double strtocard (string c)</code>	Menerima sebuah string dan mengembalikan nilai kartu tersebut dalam bentuk integer
<code>void inputkartu (vector&lt;string&gt; *kartu_input)</code>	Menerima sebuah vector string dan mengisi vector string tersebut dengan 4 kartu yang diinput
<code>bool isduplicate (vector&lt;string&gt; *hasil, string str)</code>	Menerima sebuah vector string dan sebuah string lalu mengembalikan nilai true jika string tersebut sudah ada di dalam vector string dan false jika tidak
<code>void validator24 (double nums[4], vector&lt;string&gt; *hasil)</code>	Menerima sebuah array of double dan sebuah vector string dan mengisi vector string dengan semua kemungkinan operasi yang menghasilkan 24

<pre>void perm(double *nums, vector&lt;string &gt; *hasil)</pre>	Menerima sebuah array of double, dan vector string, lalu mengisi vector string dengan semua kemungkinan operasi yang menghasilkan 24 (terjadi pemanggilan fungsi validator24) dengan mencari semua kemungkinan urutan angka-angka yang terdapat pada array of double
--	--

## 2.4 File result.cpp

File ini berisi fungsi untuk menampilkan hasil dari solusi yang memungkinkan dan terdapat juga fungsi untuk melakukan penyimpanan ke file .txt .

Methods	Description
<pre>void writefile(vector&lt;st ring&gt; kartu, vector&lt;string&gt; *hasil, double runtime)</pre>	Menerima sebuah vector string kartu, vector string hasil, dan double waktu eksekusi program dan menuliskan data tersebut ke bentuk file .txt
<pre>Void printruntime (double runtime)</pre>	Menuliskan waktu eksekusi program dalam milisecond (ms)
<pre>Void splashscreen()</pre>	Memberikan output splashscreen untuk <i>user</i>

## 2.5 Library

Terdapat juga beberapa library yang digunakan untuk program ini, antara lain :

- `#include <iostream>`
- `#include <cstdlib>`
- `#include <string>`
- `#include <vector>`
- `#include <fstream>`
- `#include <chrono>`

Lalu juga menggunakan *using namespace std;* untuk mempermudah penulisan *commands*.



## BAB III

### SOURCE CODE PROGRAM

#### 3.1 Repository Program

Repository program dapat diakses melalui tautan *GitHub* berikut :

[https://github.com/IceTeaXXD/Tucill\\_13521024](https://github.com/IceTeaXXD/Tucill_13521024)

#### 3.2 Source Code Program

##### 3.2.1 main.cpp

```
#include <iostream>
#include <cstdlib>
#include <string>
#include <vector>
#include <fstream>
#include <chrono>

#include "operations.hpp"
#include "cards.hpp"
#include "result.hpp"

using namespace std;

int main(){
    /* INISIASI VARIABEL */
    splashscreen();
    bool run = true;
    bool valid;
    int pilihan;
    vector<string> kartu_input;
    vector<string> hasil;
    string arr_kartu[] =
{"A", "2", "3", "4", "5", "6", "7", "8", "9", "10", "J", "Q", "K"};
    double nilai_kartu[4];
    double runtime;
```

```

/* MAIN PROGRAM LOOP */
while (run){
    cout << "==== MENU UTAMA =====>> endl;
    cout << "1. Input Kartu" << endl;
    cout << "2. Random Kartu" << endl;
    cout << "3. Exit" << endl;

    /* PILIHAN MENU */
    cout << "Pilihan (1/2/3) : ";
    cin >> pilihan;

    /* MENU INPUT KARTU (1) */
    if (pilihan == 1){
        /* INPUT KARTU DAN VALIDASI */
        valid= false;
        while (!valid){
            inputkartu(&kartu_input);
            /* VALIDASI */
            for (int i = 0 ; i < 4 ; i++){
                /* MERUBAH INPUT KARTU MENJADI INTEGER */
                nilai_kartu[i] = strtocard(kartu_input[i]);
                if (nilai_kartu[i] == 999){
                    cout << "Kartu tidak valid" << endl;
                    kartu_input.clear();
                    break;
                }
            }
            /* Jika semua kartu berhasil divalidasi */
            if (i == 3){
                valid = true;
            }
        }
    }

    /* MAIN PROGRAM UNTUK MENCARI SOLUSI */
    auto start = chrono::steady_clock::now(); // start timer
    perm(nilai_kartu, &hasil); // Memanggil fungsi permutasi
    untuk mendapatkan semua kemungkinan kombinasi kartu
    auto end = chrono::steady_clock::now(); // hitung runtime

```

```

dalam milisecond
    double runtime =
chrono::duration_cast<chrono::microseconds>(end - start).count() *
0.001; // hitung runtime dalam milisecond

    /* OUTPUT AKHIR */
    if (hasil.size() == 0){
        cout << "===== " << endl;
        cout << "|Tidak ada solusi      |" << endl;
        cout << "===== " << endl;
        printruntime(runtime);
    }
    else{
        cout << "===== " << endl;
        cout << "| Jumlah solusi : " << hasil.size() << "
|" << endl;

        cout << "===== " << endl;
        for (int i = 0; i < hasil.size(); i++){
            cout << hasil[i] << endl;
        }
        printruntime(runtime);
    }

    /* MENULISKAN SOLUSI KE FILE */
    writefile(kartu_input, &hasil, runtime);

    /* RESET VARIABEL*/
    kartu_input.clear();
    hasil.clear();

    cout << "Press enter to continue...";
    cin.ignore();
    cin.get();
}

/* MENU RANDOM KARTU */
else if (pilihan == 2){
    srand(time(0)); // seed random menggunakan waktu

```

```

        /* LOOP RANDOM KARTU */
        for (int i = 0; i < 4; i++) {
            int rand_num = rand() % 13; // random number dari
0-12 (sesuai jumlah kartu)
            kartu_input.push_back(arr_kartu[rand_num]); //
memasukkan kartu ke vector, dari array kartu dengan index random
            nilai_kartu[i] = strtocard(arr_kartu[rand_num]); //
Merubah kartu menjadi integer
            cout << arr_kartu[rand_num] << " "; // menampilkan
kartu
        }
        cout << endl;

        /* MAIN PROGRAM UNTUK MENCARI SOLUSI */
        auto start = chrono::steady_clock::now(); // start timer
perm(nilai_kartu, &hasil); // Memanggil fungsi permutasi
untuk mendapatkan semua kemungkinan kombinasi kartu
        auto end = chrono::steady_clock::now(); // hitung runtime
dalam milisecond
        double runtime =
chrono::duration_cast<chrono::microseconds>(end - start).count() *
0.001; // hitung runtime dalam milisecond

        /* OUTPUT AKHIR */
        if (hasil.size() == 0){
            cout << "===== " << endl;
            cout << "Tidak ada solusi" << endl;
            cout << "===== " << endl;
            printruntime(runtime);
        }
        else{
            cout << "===== " << endl;
            cout << "| Jumlah solusi : " << hasil.size() << "
|" << endl;

            cout << "===== " << endl;
            for (int i = 0; i < hasil.size(); i++){
                cout << hasil[i] << endl;
            }
            printruntime(runtime);

```

```

    }

    /* MENULISKAN SOLUSI KE FILE */
    writefile(kartu_input, &hasil, runtime);

    /* RESET VARIABEL*/
    kartu_input.clear();
    hasil.clear();

    cout << "Press enter to continue...";
    cin.ignore();
    cin.get();
}

/* MENU EXIT */
else if (pilihan == 3){
    run = false;
    cout << "Thank you for using this program!" << endl;
    cout << "Have a nice day!" << endl;
    cout << "Press enter to exit...";
    cin.ignore();
    cin.get();
}

/* JIKA PILIHAN TIDAK VALID */
else{
    cout << "Pilihan salah, ulangi input!" << endl;
    cout << "Press enter to continue...";
    cin.ignore();
    cin.get();
}
splashscreen();
}
return 0;
}

```

### 3.2.2 operations.cpp

```
#include "operations.hpp"

string inttostr(int num){
    /*
    I.S. Menerima sebuah integer
    F.S. Mengembalikan nilai integer tersebut dalam bentuk string
    */

    string str = "";
    while (num > 0){
        str = (char)(num % 10 + 48) + str;
        num /= 10;
    }
    return str;
}

string inttoop(int op){
    /*
    I.S. Menerima sebuah integer
    F.S. Mengembalikan nilai operator tersebut dalam bentuk string
    */

    if (op == 0){
        return "+";
    }
    else if (op == 1){
        return "-";
    }
    else if (op == 2){
        return "*";
    }
    else if (op == 3){
        return "/";
    }
    else{
        return "error";
    }
}

double operasi (double a, double b, int op){
    /*
```

I.S. Menerima dua buah bilangan dan sebuah operator  
F.S. Mengembalikan hasil operasi antara dua buah bilangan tersebut  
\*/

```
if (op == 0){
    return a+b;
}
else if (op == 1){
    return a-b;
}
else if (op == 2){
    return a*b;
}
else if (op == 3){
    return a/b;
}
else{
    return 999.0;
}
}
```

### 3.2.3 cards.cpp

```
void perm(double *nums, vector<string> *hasil){
    /*
    I.S. Menerima sebuah array of double, dan sebuah vector string
    F.S. Mengisi vector string dengan semua kemungkinan operasi yang
    menghasilkan 24 dari angka-angka yang terdapat pada array of double
    */
    double nums_perm[4];
    for (int i = 0 ; i < 4 ; i++){
        for (int j = 0 ; j < 4 ; j++){
            for (int k = 0 ; k < 4 ; k++){
                for (int l = 0 ; l < 4 ; l++){
                    if (i != j && i != k && i != l && j != k && j != l && k
!= l){
                        nums_perm[0] = nums[i];
                        nums_perm[1] = nums[j];
                        nums_perm[2] = nums[k];
                        nums_perm[3] = nums[l];
                        validator24(nums_perm, hasil);
                    }
                }
            }
        }
    }
}
```

```

}
    }
    }
    }
    }

```

### 3.2.4 result.cpp

```
#include "result.hpp"

void writefile(vector<string> kartu, vector<string> *hasil, double runtime){
    /*
        I.S. Menerima sebuah vector string kartu, sebuah vector string hasil,
        dan sebuah double
        F.S. Menulis hasil ke file
    */

    /* INISIASI VARIABEL */
    bool save = false;
    string yesno;
    string filename;
    string path;

    /* SAVE LOOP */
    while (!save){
        cout << "Simpan hasil ke file? (Y/N) : ";
        cin >> yesno;

        /* SAVE FILE */
        if (yesno == "Y" || yesno == "y"){
            /* INPUT NAMA FILE */
            cout << "Masukkan nama file : ";
            cin >> filename;
            path = "test\\" + filename + ".txt";
            const char * filename_char = path.c_str(); // Mengubah string
            menjadi bentuk constan char

            /* MEMBUAT FILE */
            ofstream file;
```





```

    cout << "    ) | | | | _      | (----`| | | | | | |      \\\ \\\
/ | | _ | |_) |      " << endl;
    cout << "    / / _ _      \\\ \\\ | | | | | | |      \\\
/ | _ | |      /      " << endl;
    cout << "    / / _      | |      .----) | | `--' | | `----. \\\
/ | | _ _ | | \\\ \\\----." << endl;
    cout << " | _ | _ | _ _ /      \\\ _ _ / | _ _ |
\\ _ /      | _ _ || _ | `._ _|" << endl;
    cout << "
                                BY 13521024 - AHMAD NADIL
" << endl;
    cout <<
"=====
===== " << endl;
    cout << " " << endl;
}

```

## BAB IV

### MASUKAN DAN LUARAN PROGRAM

#### 4.1 Input dari User

##### 4.1.1 A89Q

```
=====
|  A SOLVER  |
|  BY 13521024 - AHMAD NADIL  |
=====

===== MENU UTAMA =====
1. Input Kartu
2. Random Kartu
3. Exit
Pilihan (1/2/3) : 1
Masukkan 4 kartu (A,2,3,4,5,6,7,8,9,10,J,Q,K) : A 8 9 Q
=====
```

```
=====
| Jumlah solusi : 48 |
=====
((1 - 8) + 9) * 12
(1 - (8 - 9)) * 12
(1 * 8) * (12 - 9)
1 * (8 * (12 - 9))
(1 + (9 - 8)) * 12
((1 + 9) - 8) * 12
1 * ((12 - 9) * 8)
(1 * (12 - 9)) * 8
((1 * 12) - 9) * 8
(8 * 1) * (12 - 9)
8 * (1 * (12 - 9))
8 * ((1 * 12) - 9)
(8 / 1) * (12 - 9)
8 / (1 / (12 - 9))
8 * (12 - (9 * 1))
8 * ((12 - 9) * 1)
(8 * (12 - 9)) * 1
8 * (12 - (9 / 1))
8 * ((12 - 9) / 1)
(8 * (12 - 9)) / 1
8 * (12 - (1 * 9))
8 * ((12 * 1) - 9)
8 * ((12 / 1) - 9)
((9 - 8) + 1) * 12
(9 - (8 - 1)) * 12
(9 + (1 - 8)) * 12
((9 + 1) - 8) * 12
```

```
(12 - 9) * (8 * 1)
((12 - 9) * 8) * 1
(12 - 9) * (8 / 1)
((12 - 9) * 8) / 1
12 * ((9 - 8) + 1)
12 * (9 - (8 - 1))
(12 - 9) * (1 * 8)
(12 - (9 * 1)) * 8
((12 - 9) * 1) * 8
(12 - (9 / 1)) * 8
((12 - 9) / 1) * 8
(12 - 9) / (1 / 8)
12 * (9 + (1 - 8))
12 * ((9 + 1) - 8)
(12 - (1 * 9)) * 8
12 * (1 + (9 - 8))
12 * ((1 + 9) - 8)
((12 * 1) - 9) * 8
((12 / 1) - 9) * 8
12 * ((1 - 8) + 9)
12 * (1 - (8 - 9))

=====
| Waktu Eksekusi : 0.172 ms |
=====
```

```
Simpan hasil ke file? (Y/N) : y
Masukkan nama file : A89Q
Solusi telah disimpan di file A89Q
Press enter to continue...|
```

#### 4.1.2 J55Q

```
=====
| 24 SOLVERS |
| BY 13521024 - AHMAD NADIL |
=====

===== MENU UTAMA =====
1. Input Kartu
2. Random Kartu
3. Exit
Pilihan (1/2/3) : 1
Masukkan 4 kartu (A,2,3,4,5,6,7,8,9,10,J,Q,K) : J 5 5 Q
```

```
=====
| Jumlah solusi : 22 |
=====
11 + ((5 * 5) - 12)
(11 + (5 * 5)) - 12
11 + ((5 / 5) + 12)
(11 + (5 / 5)) + 12
(11 + 12) + (5 / 5)
11 + (12 + (5 / 5))
(11 - 12) + (5 * 5)
11 - (12 - (5 * 5))
(5 * 5) + (11 - 12)
((5 * 5) + 11) - 12
(5 / 5) + (11 + 12)
((5 / 5) + 11) + 12
((5 * 5) - 12) + 11
(5 * 5) - (12 - 11)
(5 / 5) + (12 + 11)
((5 / 5) + 12) + 11
(5 * (12 - 5)) - 11
12 + ((5 / 5) + 11)
(12 + (5 / 5)) + 11
((12 - 5) * 5) - 11
(12 + 11) + (5 / 5)
12 + (11 + (5 / 5))
=====
| Waktu Eksekusi : 0.141 ms |
=====
```

```
Simpan hasil ke file? (Y/N) : y
Masukkan nama file : J55Q
Solusi telah disimpan di file J55Q
Press enter to continue...|
```

```
=====
```

```
  _-_- \ | || | _-_-  
 / / | _-_- |  
 / / | _-_- |  
 | _-_- | | _-_- |  
  
 _-_- _-_- | / _-_- \ | _-_- |  
 | ( _-_- \ | _-_- |  
 . _-_- ) | _-_- |  
 | _-_- \ | _-_- |  
 | _-_- |  
  
 BY 13521024 - AHMAD NADIL
```

```
=====
```

```
==== MENU UTAMA ====
```

```
1. Input Kartu  
2. Random Kartu  
3. Exit  
Pilihan (1/2/3) : 1  
Masukkan 4 kartu (A,2,3,4,5,6,7,8,9,10,J,Q,K) : Q 4 6 J
```

```
=====
| Waktu Eksekusi : 0.127 ms |
=====
Simpan hasil ke file? (Y/N) : y
Masukkan nama file : Q46J
Solusi telah disimpan di file Q46J
Press enter to continue...|
```

## 4.2 Generate Random

### 4.2.1 8384

```
=====
24 SOLVER
BY 13521024 - AHMAD NADIL
=====

===== MENU UTAMA =====
1. Input Kartu
2. Random Kartu
3. Exit
Pilihan (1/2/3) : 2
8 3 8 4
=====
Tidak ada solusi
=====
| Waktu Eksekusi : 0.081 ms |
=====
Simpan hasil ke file? (Y/N) : y
Masukkan nama file : 8384
Solusi telah disimpan di file 8384
Press enter to continue...
```

#### 4.2.2 5QQ4

```
=====
 5  SOLVER
=====
BY 13521024 - AHMAD NADIL
=====

===== MENU UTAMA =====
1. Input Kartu
2. Random Kartu
3. Exit
Pilihan (1/2/3) : 2
5 Q Q 4
```

```
=====
| Jumlah solusi : 18 |
=====
(5 + (12 / 12)) * 4
(5 - (12 / 4)) * 12
(5 - 4) * (12 + 12)
((5 - 4) * 12) + 12
12 * (5 - (12 / 4))
12 + ((5 - 4) * 12)
(12 * (5 - 4)) + 12
(12 / (5 - 4)) + 12
(12 + 12) * (5 - 4)
12 + (12 * (5 - 4))
(12 + 12) / (5 - 4)
12 + (12 / (5 - 4))
((12 / 12) + 5) * 4
12 - (12 * (4 - 5))
12 - (12 / (4 - 5))
12 - ((4 - 5) * 12)
4 * ((12 / 12) + 5)
4 * (5 + (12 / 12))
=====
| Waktu Eksekusi : 0.127 ms |
=====
```

```
Simpan hasil ke file? (Y/N) : y
Masukkan nama file : 5QQ4
Solusi telah disimpan di file 5QQ4
Press enter to continue...|
```

### 4.2.3 J8J9

```
=====
          SOLVER
          BY 13521024 - AHMAD NADIL
=====

==== MENU UTAMA ====
1. Input Kartu
2. Random Kartu
3. Exit
Pilihan (1/2/3) : 2
J 8 J 9
=====
| Jumlah solusi : 2      |
=====
((11 - 8) * 11) - 9
(11 * (11 - 8)) - 9
=====
| Waktu Eksekusi : 0.087 ms |
=====
Simpan hasil ke file? (Y/N) : y
Masukkan nama file : J8J9
Solusi telah disimpan di file J8J9
Press enter to continue...|
```



## **BAB V**

### **LAMPIRAN**

<b>Poin</b>	<b>Ya</b>	<b>Tidak</b>
1. Program berhasil dikompilasi tanpa kesalahan	<input checked="" type="checkbox"/>	
2. Program berhasil running	<input checked="" type="checkbox"/>	
3. Program dapat membaca input / generate sendiri dan memberikan luaran	<input checked="" type="checkbox"/>	
4. Solusi yang diberikan program memenuhi (berhasil mencapai 24)	<input checked="" type="checkbox"/>	
5. Program dapat menyimpan solusi dalam file teks	<input checked="" type="checkbox"/>	

## REFERENSI

[https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2021-2022/Algoritma-Brute-Force-\(2022\)-Bag1.pdf](https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2021-2022/Algoritma-Brute-Force-(2022)-Bag1.pdf)

[https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2021-2022/Algoritma-Brute-Force-\(2022\)-Bag2.pdf](https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2021-2022/Algoritma-Brute-Force-(2022)-Bag2.pdf)

<http://24solver.us-west-2.elasticbeanstalk.com/>