IF 2211 - Strategi Algoritma
Penyelesaian Permainan Kartu 24 dengan Algoritma Brute Force

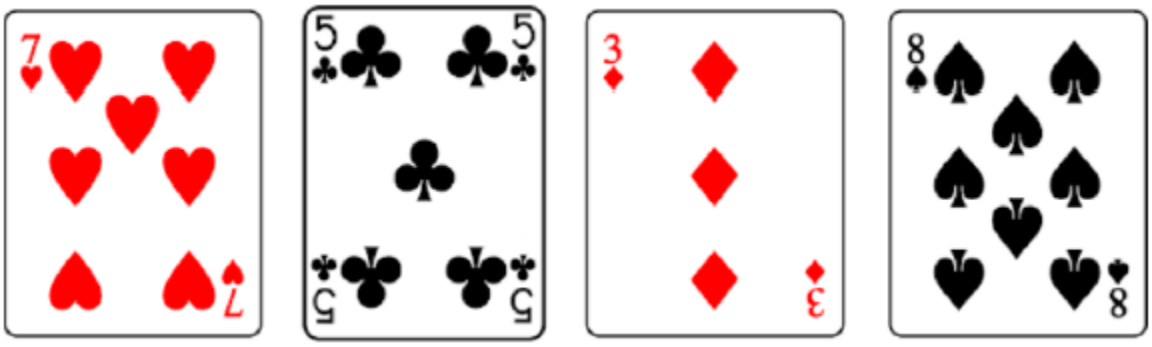
Disusun oleh :
13521139 - Nathania Calista Djunaedi

Institut Teknologi Bandung
Sekolah Teknik Elektro dan Informatika
Tahun Ajaran 2022/2023

Daftar Isi

Bab 1
Deskripsi Masalah

1.1 Permainan Kartu 24



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 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, King). As memiliki nilai 1, Jack bernilai 11, Queen bernilai 12, King bernilai 13, dan kartu bilangan memiliki nilai sesuai dengan 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 dapat dilakukan degan menggunakan operasi dasar matematika, yaitu penjumlahan (+), pengurangan (-), perkalian (x) , divisi (/) dan variasi tanda kurung. Tiap kartu harus digunakan tepat sekali dan urutan penggunaannya dibebaskan.

Bab 2

Landasan Teori dan Algoritma

2.1 Algoritma Brute Force

Algoritma *brute force* adalah pendekatan dari penyelesaian sebuah masalah algoritmik dengan menggunakan pendekatan yang lempeng atau *straight-forward*. Algoritma ini didasarkan pada pernyataan pada persoalan (*problem statement*) dan definisi konsep yang dilibatkan. Beberapa contoh penggunaan algoritma *brute force* yang sering digunakan adalah *Bubble Sort* dan *Selection Sort*. Algoritma ini memecahkan persoalan dengan sangat sederhana, langsung, dna jelas.

Karakteristik dari algoritma *brute force* adalah tidak cerdas, karena membutuhkan jumlah komputasi yang besar dan waktu yang lama. Selain itu, algoritma *brute force* lebih cocok untuk persoalan yang berukuran kecil karena algoritma ini sederhana dan implementasinya mudah. Meskipun bukan algoritma tercepat, algoritma ini dapat menyelesaikan hampir semua persoalan.

2.2 Kombinasi dan Permutasi

Kombinasi adalah cara untuk menggabungkan beberapa buah objek, tanpa memperhatikan urutan. Permutasi adalah susunan yang dapat dibentuk dair suatu kumpulan objek yang diambil sebagian atau seluruhnya dengan memperhatikan urutannya.

$$^{n}C_{k} = \frac{n!}{k!(n-k)!} \quad ^{n}P_{k} = \frac{n!}{(n-k)!}$$

Permutasi dapat dibagi lagi menjadi dua, yaitu permutasi dengan pengulangan dan permutasi tanpa pengulangan. Permutasi dengan pengulangan artinya permutasi yang memperbolehkan objek yang sama untuk muncul lebih dari sekali.

2.3 Langkah - Langkah Algoritma *Brute Force* dalam menyelesaikan permainan kartu 24

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

1. Meminta input 4 kartu dari user dan melakukan validasi atas input user. Jika input belum sesuai dengan ketentuan, akan diminta memasukkan ulang. Selain meminta input, ada juga *option* untuk mendapatkan angka secara *random.*Kartu yang dihasilkan, disimpan dalam sebuah string. Misalnya, user memasukkan input A 2 3 4, akan digabungkan menjadi string “A234”.
2. Dari keempat kartu yang didapatkan dari user atau *random generator*, dicari permutasi dan disimpan dalam sebuah array. Karena bentuk kartu merupakan string, array yang disediakan akan berukuran 24 dan menyimpan elemen dalam tipe data string.
3. Mencatat semua permutasi yang memungkinkan dari 4 jenis operator, yaitu penjumlahan (+), pengurangan (-), perkalian (*) dan pembagian (/). Dari 4 operator yang ada, hanya dipilih 3 operator, karena permainan kartu 24 hanya memiliki 4 kartu, sehingga operator yang digunakan hanya 3. Semua variasi operator disimpan di sebuah array of strings yang berukuran 64 (4 x 4 x 4). Misalnya, salah satu contoh permutasi dari semua operator tersebut adalah “+++”, “---”, “***”, “///”, dan masih banyak lagi.
4. Kemudian, untuk setiap kombinasi kartu, disisipkan semua variasi operator dan digabungkan menjadi sebuah operasi utuh yang terdiri dari 4 kartu dan 3 operator. Operasi yang didapatkan akan berjumlah 1536, yang didapatkan dari perkalian 24 (variasi kartu) dengan 64 (variasi operator). Misalnya ada kartu “A234” dan operator “---”, maka akan terbentuk sebuah operasi “A - 2 - 3 - 4”.
5. Untuk setiap operasi yang ada, ada 5 variasi posisi tanda kurung, yaitu :
    - ((a op b) op c) op d
    - (a op (b op c)) op d
    - (a op b) op (c op d)
    - a op ((b op c) op d)
    - a op (b op (c op d))

Sehingga, total variasi yang harus dihitung ada sebanyak 7680 yang merupakan hasil perkalian 1536 (total operasi) dengan 5 variasi kurung.

6. Dari semua hasil perhitungan, jika ada yang menghasilkan angka 24, operasinya akan disimpan dalam sebuah array, yang berukuran 1000. Pada tugas kali ini, penulis menggunakan array berukuran 1000, karena tidak mungkin solusi yang menghasilkan angka 24 mencapai angka 1000.
7. Setelah hasil ditampilkan di terminal, *user* akan diberikan pilihan untuk menyimpan jawaban atau tidak

Bab 3

Input dan Output Program

3.1 Testcase 1

Kasus dimana kartu berasal dari input user dan jawaban tidak disimpan

```
================== Welcome to ==================
================= 24 Game Solver=================
=================================================
Silakan masukkan nomor menu yang anda ingingkan :
1. Memasukkan kombinasi kartu secara manual
2. Mengambil angka random
3. Keluar
1
Masukkan kombinasi kartu yang anda inginkan :
J A K 2
```

```
Ada sebanyak 24 solusi yang ditemukan
J - ( ( A - 2 ) * K )
J - ( K * ( A - 2 ) )
J - ( K / ( A - 2 ) )
( J + K ) * ( 2 - A )
J + ( K * ( 2 - A ) )
( J + K ) / ( 2 - A )
J + ( K / ( 2 - A ) )
( J * ( 2 - A ) ) + K
J + ( ( 2 - A ) * K )
( J / ( 2 - A ) ) + K
K - ( ( A - 2 ) * J )
K - ( J * ( A - 2 ) )
K - ( J / ( A - 2 ) )
( K + J ) * ( 2 - A )
K + ( J * ( 2 - A ) )
( K + J ) / ( 2 - A )
K + ( J / ( 2 - A ) )
( K * ( 2 - A ) ) + J
K + ( ( 2 - A ) * J )
( K / ( 2 - A ) ) + J
( ( 2 - A ) * K ) + J
( 2 - A ) * ( K + J )
( ( 2 - A ) * J ) + K
( 2 - A ) * ( J + K )
```

```
Time taken by program is : 0.038960 seconds
Apakah anda mau menyimpan jawaban anda?
Ketik 1 jika mau dan 2 jika tidak
2
```

3.2. Testcase 2

Kasus dimana kartu diambil dari input user dan jawaban disimpan dalam file .txt

```
================== Welcome to ==================
================= 24 Game Solver=================
================================================
Silakan masukkan nomor menu yang anda ingingkan :
1. Memasukkan kombinasi kartu secara manual
2. Mengambil angka random
3. Keluar
1
Masukkan kombinasi kartu yang anda inginkan :
10 9 8 7
```

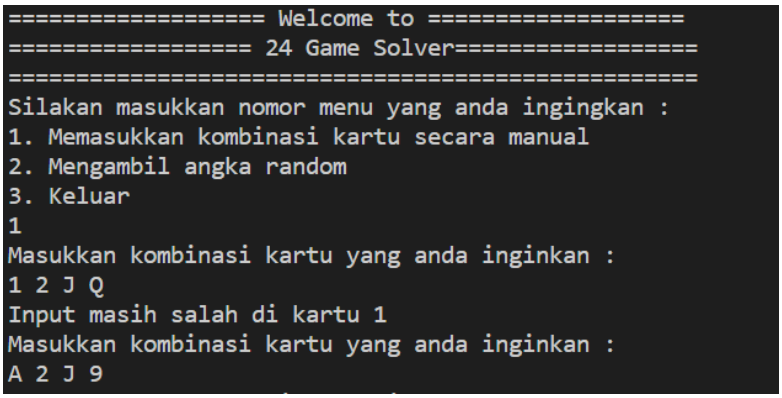
```
Ada sebanyak 8 solusi yang ditemukan
( 9 / ( 10 - 7 ) ) * 8
9 / ( ( 10 - 7 ) / 8 )
( 9 * 8 ) / ( 10 - 7 )
9 * ( 8 / ( 10 - 7 ) )
( 8 * 9 ) / ( 10 - 7 )
8 * ( 9 / ( 10 - 7 ) )
( 8 / ( 10 - 7 ) ) * 9
8 / ( ( 10 - 7 ) / 9 )
Time taken by program is : 0.009530 seconds
Apakah anda mau menyimpan jawaban anda?
Ketik 1 jika mau dan 2 jika tidak
1
Masukkan nama file yang anda inginkan (tanpa .txt)
testcase2
```

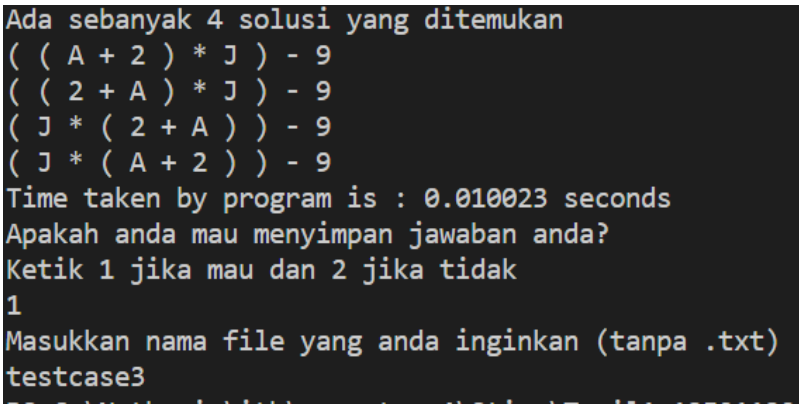
test > testcase2.txt

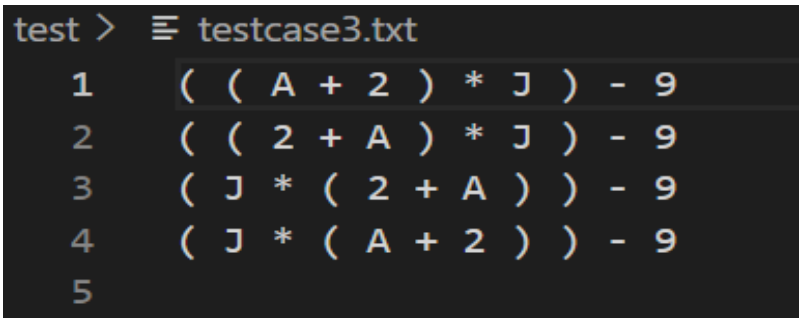
```
1   ( 9 / ( 10 - 7 ) ) * 8
2   9 / ( ( 10 - 7 ) / 8 )
3   ( 9 * 8 ) / ( 10 - 7 )
4   9 * ( 8 / ( 10 - 7 ) )
5   ( 8 * 9 ) / ( 10 - 7 )
6   8 * ( 9 / ( 10 - 7 ) )
7   ( 8 / ( 10 - 7 ) ) * 9
8   8 / ( ( 10 - 7 ) / 9 )
```

3.3 Testcase 3

Kasus dimana input yg user masukan tidak sesuai dengan format (*error handling*)







3.4 Testcase 4

Kasus dimana user meminta kartu *random* dan tidak terdapat jawaban

```
================== Welcome to ==================
================= 24 Game Solver=================
================================================
Silakan masukkan nomor menu yang anda ingingkan :
1. Memasukkan kombinasi kartu secara manual
2. Mengambil angka random
3. Keluar
2
Kartu yang anda dapat :
6 J A A
```

```
Ada sebanyak 0 solusi yang ditemukan
Time taken by program is : 0.008903 seconds
============ Terima kasih sudah bermain ============
```

3.5 Testcase 5

Kasus dimana user meminta input random, tapi tidak disimpan dalam .txt

```
================== Welcome to ==================
================= 24 Game Solver=================
================================================
Silakan masukkan nomor menu yang anda ingingkan :
1. Memasukkan kombinasi kartu secara manual
2. Mengambil angka random
3. Keluar
2
Kartu yang anda dapat :
9 9 5 Q
```

```
Ada sebanyak 16 solusi yang ditemukan
( 9 * ( 9 - 5 ) ) - Q
9 - ( ( 9 - Q ) * 5 )
( 9 * 5 ) - ( 9 + Q )
( ( 9 * 5 ) - 9 ) - Q
( ( 9 - 5 ) * 9 ) - Q
9 - ( 5 * ( 9 - Q ) )
( 9 * 5 ) - ( Q + 9 )
( ( 9 * 5 ) - Q ) - 9
9 + ( 5 * ( Q - 9 ) )
9 + ( ( Q - 9 ) * 5 )
( 5 * 9 ) - ( 9 + Q )
( ( 5 * 9 ) - 9 ) - Q
( 5 * 9 ) - ( Q + 9 )
( ( 5 * 9 ) - Q ) - 9
( 5 * ( Q - 9 ) ) + 9
( ( Q - 9 ) * 5 ) + 9
```

```
Time taken by program is : 0.016439 seconds
Apakah anda mau menyimpan jawaban anda?
Ketik 1 jika mau dan 2 jika tidak
2
============= Terima kasih sudah bermain =============
```

3.6 Testcase 6

Kasus dimana user meminta 4 kartu random dan jawaban disimpan dalam file .txt

```
=================== Welcome to ===================
================== 24 Game Solver==================
==================================================
Silakan masukkan nomor menu yang anda ingingkan :
1. Memasukkan kombinasi kartu secara manual
2. Mengambil angka random
3. Keluar
2
```

```
Ada sebanyak 20 solusi yang ditemukan
( Q * ( 7 - 6 ) ) * 2
Q * ( ( 7 - 6 ) * 2 )
( Q / ( 7 - 6 ) ) * 2
Q / ( ( 7 - 6 ) / 2 )
( Q * 2 ) * ( 7 - 6 )
Q * ( 2 * ( 7 - 6 ) )
( Q * 2 ) / ( 7 - 6 )
Q * ( 2 / ( 7 - 6 ) )
( ( 7 - 6 ) * Q ) * 2
( 7 - 6 ) * ( Q * 2 )
( ( 7 - 6 ) * 2 ) * Q
( 7 - 6 ) * ( 2 * Q )
( 2 * ( 7 - 6 ) ) * Q
2 * ( ( 7 - 6 ) * Q )
( 2 / ( 7 - 6 ) ) * Q
2 / ( ( 7 - 6 ) / Q )
( 2 * Q ) * ( 7 - 6 )
2 * ( Q * ( 7 - 6 ) )
( 2 * Q ) / ( 7 - 6 )
2 * ( Q / ( 7 - 6 ) )
```

```
Time taken by program is : 0.018554 seconds
Apakah anda mau menyimpan jawaban anda?
Ketik 1 jika mau dan 2 jika tidak
1
Masukkan nama file yang anda inginkan (tanpa .txt)
testcase6
```

test > testcase6.txt

```
1   ( Q * ( 7 - 6 ) ) * 2
2   Q * ( ( 7 - 6 ) * 2 )
3   ( Q / ( 7 - 6 ) ) * 2
4   Q / ( ( 7 - 6 ) / 2 )
5   ( Q * 2 ) * ( 7 - 6 )
6   Q * ( 2 * ( 7 - 6 ) )
7   ( Q * 2 ) / ( 7 - 6 )
8   Q * ( 2 / ( 7 - 6 ) )
9   ( ( 7 - 6 ) * Q ) * 2
10  ( 7 - 6 ) * ( Q * 2 )
11  ( ( 7 - 6 ) * 2 ) * Q
12  ( 7 - 6 ) * ( 2 * Q )
13  ( 2 * ( 7 - 6 ) ) * Q
14  2 * ( ( 7 - 6 ) * Q )
15  ( 2 / ( 7 - 6 ) ) * Q
16  2 / ( ( 7 - 6 ) / Q )
17  ( 2 * Q ) * ( 7 - 6 )
18  2 * ( Q * ( 7 - 6 ) )
19  ( 2 * Q ) / ( 7 - 6 )
20  2 * ( Q / ( 7 - 6 ) )
```

Bab 4

Source Code Program

3.1 Repository Github

Link : https://github.com/nathaniacalista01/Tucil1_13521139

3.2 Source Code dalam Bahasa C++

Dalam menyelesaikan permainan kartu 24, penulis menggunakan bahasa pemograman C++ dengan source code sebagai berikut :

```
#include <iostream>
using namespace std;
#include <fstream>
using std::ofstream;
#include <chrono>
#include <time.h>

void createEmptyArray(int length, string *results){
    for(int i = 0; i < length;i++){
        results[i] = "#";
    }
}

void insertFirst(int length, string *answers, string items){
    int i =0;
    while(true && i < length){
        if(answers[i] == "#"){
            answers[i] = items;
            break;
        }
        i += 1;
    }
}

int displayMenu(){
    int options;
```

```
    while(true){
        cout << "=================== Welcome to ==================="
<<endl;
        cout << "================== 24 Game Solver=================="
<<endl;
        cout << "=================================================="
<<endl;
        cout << "Silakan masukkan nomor menu yang anda ingingkan : "
<<endl;
        cout << "1. Memasukkan kombinasi kartu secara manual" <<endl;
        cout << "2. Mengambil angka random " <<endl;
        cout << "3. Keluar " <<endl;
        cin >> options;
        if(options == 1 || options == 2 || options == 3){
            break;
        }
    }
    return options;
}

void swapCards(string *a, string *b){
    string temp;
    temp = *a;
    *a = *b;
    *b = temp;
}

void permutation(string &cards, int start, int lastIndex, string
results[]){
    if(start == lastIndex){
        insertFirst(24,results,cards);
        return;
    }
    for(int i = start; i <= lastIndex;i++){
        swap(cards[start],cards[i]);
        permutation(cards,start+1,lastIndex,results);
        swap(cards[start],cards[i]);
    }
}
```

```
void cardCombination(string card,string results[24]){
    createEmptyArray(24,results);
    permutation(card,0,3,results);
}

void allCombinations(string operators, string answer[64]){
    for(int i = 0; i < 4 ; i++){
        string temp = "###";
        temp[0] = operators[i];
        for(int j = 0; j< 4;j++){
            temp[1] = operators[j];
            for(int k = 0; k < 4;k++){
                temp[2] = operators[k];
                insertFirst(64,answer,temp);
            }
        }
    }
}

void operators(string answer[64]){
    string operators = "*+-/";
    createEmptyArray(64,answer);
    allCombinations(operators,answer);
}

string getOperations(string operators, string card){
    string results = "#######";
    results[0] = card[0];
    int i = 1; int cardsCount = 1; int operatorsCount = 0;
    while(i < 7){
        if(i%2 == 0){
            results[i] = card[cardsCount];
            cardsCount += 1;
        }else{
            results[i] = operators[operatorsCount];
            operatorsCount += 1;
        }
        i += 1;
    }
    return results;
```

```
}

void getAllOperations(string cards[24], string operators[64], string
operations[1536]){
    createEmptyArray(1536,operations);
    for(int i = 0; i < 24; i++){
        for(int j = 0; j < 64; j ++){
            string results = getOperations(operators[j],cards[i]);
            insertFirst(1536,operations,results);
        }
    }
}

bool isValid(string cards[4]){
    for(int i =0; i < 4;i++){
        if(cards[i] != "A" && cards[i] != "Q" && cards[i] != "J"&&
cards[i] != "K"&& cards[i] != "9"&&cards[i] != "8"&&
        cards[i] != "7"&& cards[i] != "6"&& cards[i] != "5"&& cards[i] !=
"4"&& cards[i] != "3"&& cards[i] != "2" ){
            if(cards[i] == "10"){
                cards[i] = "0";
            }else{
                cout << "Input masih salah di kartu ";
                cout << cards[i] <<endl;
                return false;
            }

        }
    }
    return true;
}

string randomizeCard(){
    string results = "####";
    srand(time(0));
    for(int i = 0; i < 4; i++){
        int temp = rand()%12 + 1;
        if(temp == 1){
            results[i] = 'A';
        }else if(temp == 10){
```

```
            results[i] = '0';
        }else if(temp == 11){
            results[i] = 'J';
        }else if(temp == 12){
            results[i] = 'Q';
        }else if (temp == 13){
            results[i] = 'K';
        }else{
            results[i] = temp + '0';
        }
    }
    cout << "Kartu yang anda dapat : " <<endl;
    for(int i = 0; i < 4; i++){
        cout << results[i] << " ";
    }
    cout << "\n";
    return results;
}

string userInput(string cards[4]){
    bool allValid = true;
    string card;
    do
    {
        cout << "Masukkan kombinasi kartu yang anda inginkan : " <<endl;
        cin >> cards[0] >> cards[1] >>cards[2] >>cards[3];
        allValid = isValid(cards);
        card = "";
        for(int i =0; i < 4;i++){
            card += cards[i];
        }
    } while (!allValid);
    return card;
}

float charToNum(char c){
    switch (c){
    case 'A':
        return 1.0;
        break;
```

```
    case 'J':
        return 11.0;
        break;
    case 'Q' :
        return 12.0;
        break;
    case 'K' :
        return 13.0;
        break;
    case '0':
        return 10.0;
        break;
    default:
        return (float)((int)c - 48);
        break;
    }
}

float calculate(float first, char ops, float sec){
    switch (ops){
    case '+':
        return first + sec;
        break;
    case '-':
        return first - sec;
        break;
    case '/':
        if(sec == 0){
            return -9999.0;
        }
        return first/sec;
        break;
    default:
        return first*sec;
        break;
    }
}

void parantheses(string operations, float count[5], string results[5]){
    float firstNum = charToNum(operations[0]);
```

```
    float secondNum = charToNum(operations[2]);
    float thirdNum = charToNum(operations[4]);
    float fourthNum = charToNum(operations[6]);

    results[0] =
{'(','(',operations[0],operations[1],operations[2],')',operations[3],opera
tions[4],')',operations[5],operations[6]};
    count[0] =
calculate(calculate(calculate(firstNum,operations[1],secondNum),operations
[3],thirdNum),operations[5],fourthNum);

    results[1] =
{'(',operations[0],operations[1],'(',operations[2],operations[3],operation
s[4],')',')',operations[5],operations[6]};
    count[1] =
calculate(calculate(firstNum,operations[1],calculate(secondNum,operations[
3],thirdNum)),operations[5],fourthNum);

    results[2] =
{'(',operations[0],operations[1],operations[2],')',operations[3],'(',opera
tions[4],operations[5],operations[6],')'};
    count[2] =
calculate(calculate(firstNum,operations[1],secondNum),operations[3],calcul
ate(thirdNum,operations[5],fourthNum));

    results[3]=
{operations[0],operations[1],'(','(',operations[2],operations[3],operation
s[4],')',operations[5],operations[6],')'};
    count[3] =
calculate(firstNum,operations[1],calculate(calculate(secondNum,operations[
3],thirdNum),operations[5],fourthNum));

    /* Persamaan kelima */
    results[4] =
{operations[0],operations[1],'(',operations[2],operations[3],'(',operation
s[4],operations[5],operations[6],')',')'};
    count[4] =
calculate(firstNum,operations[1],calculate(secondNum,operations[3],calcula
te(thirdNum,operations[5],fourthNum)));
}
```

```
void exitMsg(){
    cout << "============= Terima kasih sudah bermain ============="
<<endl;
}

void displayAnswer(string finalAnswers[7680]){
    int i = 0;
    bool empty = false;
    while(finalAnswers[i] != "#" && !empty){
        if(finalAnswers[i] == "#"){
            empty = true;
        }else{
            for(int j = 0; j < 11; j++){
                if(finalAnswers[i][j] == '0'){
                    cout << "10 ";
                }else{
                    cout << finalAnswers[i][j] << " ";
                }
            }
            cout << "\n";
        }
        i += 1;
    }
}

void saveFile(string finalAnswers[7680]){
    int options;
    while(true){
        cout << "Apakah anda mau menyimpan jawaban anda? "<<endl;
        cout << "Ketik 1 jika mau dan 2 jika tidak " <<endl;
        cin >> options;
        if(options == 1 || options == 2){
            break;
        }else{
            cout << "Input anda masih kurang tepat, silakan ulangi lagi!"
<<endl;
        }
    }
    if(options == 1){
```

```
        string fileName;
        cout << "Masukkan nama file yang anda inginkan (tanpa .txt) "
<<endl;
        fstream newFile;
        cin >> fileName;
        newFile.open("../test/" + fileName + ".txt",ios::out);
        if(newFile.is_open()){
            int i = 0;
            bool empty = false;
            while(finalAnswers[i] != "#" && !empty){
                if(finalAnswers[i] == "#"){
                    empty = true;
                }else{
                    for(int j = 0; j < 11; j++){
                        if(finalAnswers[i][j] == '0'){
                            newFile << "10 ";
                        }else{
                            newFile << finalAnswers[i][j] << " ";
                        }
                    }
                    newFile << "\n";
                }
                i += 1;
            }
        }else{
            cout << "Tidak bisa membuka file "<<endl;
        }
    }else{
        exitMsg();
    }
}

bool isDuplicate(string answers[7680],string results){
    bool empty = false;
    int i = 0;
    while(!empty && i < 7680){
        if(answers[i] == results){
            return true;
        }else if(answers[i] == "#"){
            empty = true;
```

```
        }
        i += 1;
    }
    return false;
}

void calculateCard(string card){
    string allCardsCombination[24];
    string allOperators[64];
    string allOperations[1536];

    string finalAnswers[7680];
    int totalAnswer = 0;
    auto start = chrono :: high_resolution_clock::now();
    ios_base::sync_with_stdio(false);

    /* Semua kombinasi kartu (ada 24) */
    cardCombination(card,allCardsCombination);
    /* Semua permutasi dari operators dengan repetisi (ada 64) */
    operators(allOperators);

    getAllOperations(allCardsCombination,allOperators,allOperations);
    createEmptyArray(7680,finalAnswers);
    for(int i = 0; i < 1536; i++){
        float count[5];
        string results[5];
        parantheses(allOperations[i],count,results);
        for(int j = 0; j < 5; j++){
            if(count[j] == 24.00){
                if(!isDuplicate(finalAnswers,results[j])){
                    insertFirst(7680,finalAnswers,results[j]);
                    totalAnswer += 1;
                }
            }
        }
    }
    auto end = chrono :: high_resolution_clock :: now();
    double duration = chrono::duration_cast<chrono::nanoseconds>(end -
start).count();
```

```
    duration *= 1e-9;

    cout << "Ada sebanyak ";
    cout << totalAnswer;
    cout << " solusi yang ditemukan "<<endl;
    displayAnswer(finalAnswers);
    cout << "Time taken by program is : " << fixed << duration << "
seconds" <<endl;
    if(totalAnswer != 0){
        saveFile(finalAnswers);
    }else{
        exitMsg();
    }

}

int main(){
    string cards[4];
    int options = displayMenu();
    if(options == 1){
        string card = userInput(cards);
        calculateCard(card);
    }else if(options == 2){
        string card = randomizeCard();
        calculateCard(card);
    }else{
        exitMsg();
    }
    return 0;
}
```

Bab 5

Checklist

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

Referensi

https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2021-2022/Algoritma-Brute-Force-(2022)-Bag1.pdf
https://www.zenius.net/blog/rumus-kombinasi-dan-permutasi
https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2015-2016/Makalah-2016/MakalahStima-2016-038.pdf