LAPORAN TUGAS KECIL 1

IF2211 STRATEGI ALGORITMA

PENYELESAIAN PERMAINAN 24 DENGAN ALGORITMA *BRUTE FORCE*

DISUSUN OLEH :

DANIEL EGIANT SITANGGANG

13521056 – K02

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

Pada laporan ini akan dibuat program untuk mencari solusi permainan 24 dalam bahasa C dengan pendekatan *Brute Force.*

BAB II
ALGORITMA BRUTE FORCE PROGRAM

Brute Force adalah algoritma untuk mencapai suatu tujuan dengan mencoba segala kemungkinan yang ada. Algoritma ini ‘mahal’ karena dapat memakan waktu lama namun dapat memberikan hasil yang pasti.

Pada program ini, akan di implementasikan algoritma Brute Force dalam mencari solusi permainan 24.

Pada awal program, user akan diminta memasukkan 4 buah kartu atau men-*generate* kartu secara acak. Kemudian, program akan mengevaluasi seluruh kombinasi kemungkinan berdasarkan: permutasi 4 angka yang dimasukkan, kombinasi operator antar angka, dan kombinasi *parenthesis* / tanda kurung. Ketiga capaian ini dikomputasikan pada tujuh *nested iteration* pada line 54 hingga 119 sourcecode. Empat iterasi pertama untuk mempermutasikan angka, 3 iterasi selanjutnya untuk mengkombinasikan operator, dan kombinasi *parenthesis* dicek satu-satu (*hardcode*). Apabila ekspresi tersebut menghasilkan 24, maka ekspresi tersebut dimasukkan ke dalam array of string yang berisikan solusi-solusi yang memungkinkan.

Pada akhir algoritma, akan terbentuk sebuah list yang berisikan seluruh ekspresi kombinasi yang memungkinkan. Algoritma diakhiri dengan menampilkan banyaknya solusi dan seluruh solusi yang ada. User juga dapat menyimpan solusi pada file.

BAB III
KODE PROGRAM DALAM BAHASA C

File : **main.c**

```c
#include <string.h>
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <math.h>

int card[4] ;


double eval(double num1, char op, double num2){
    if (op == '+') {
        return num1 + num2;
    }
    else if (op == '-') {
        return num1 - num2;
    }
    else if (op == '*') {
        return num1 * num2;
    }
    else if (op == ':') {
        return num1 / num2;
    }
    else {
        printf("DEBUG");
    }
}

int card_to_num(char c){
    if (c == 'A' || c == 'a'){
        return 1;
    }
    if (c == 'J' || c == 'j'){
        return 11;
    }
    if (c == 'Q' || c == 'q'){
        return 12;
    }
    if (c == 'K' || c == 'k'){
        return 13;
    }
    else {
        return c - '0';
    }
}
```

```
void find(int * num){
    int goal = 24;
    char result [1000][30] = {"."};

    int count = 0;
    char op [] = {'+','-','*',':'};

    clock_t start = clock();
    /* Permutasi angka */
    for (int l = 0; l < 4; l ++){
        int num1 = num[l];
        for (int m = 0; m < 4; m ++){
            if (m == l) {
                continue;
            }
            int num2 = num[m];
            for (int n = 0; n < 4; n ++){
                if (n == l || n == m){
                    continue;
                }
                int num3 = num[n];
                for (int o = 0; o < 4; o ++){
                    if (o == m || o == n || o == l){
                        continue;
                    }
                    int num4 = num[o];

                    /* Iterasi operator */
                    for (int i = 0 ; i < 4; i ++ ){
                        char op1 = op[i];
                        for (int j = 0; j < 4; j ++){
                            char op2 = op[j];
                            for (int k = 0; k < 4; k ++){
                                char op3 = op[k];

                                /* Kombinasi Parenthesis */

                                /* ((A op B) op C) op D */
                                if ((eval(eval(eval(num1, op1, num2), op2,
num3), op3, num4)) == goal){
                                    sprintf(result[count], "((%d %c %d) %c
%d) %c %d\n", num1, op1, num2, op2, num3, op3, num4);
                                    count ++;
                                }

                                /* (A op (B op C)) op D */
                                if ((eval(eval(num1, op1, eval(num2, op2,
num3)), op3, num4)) == goal){
```

```
                                    sprintf(result[count], "(%d %c (%d %c
%d)) %c %d\n", num1, op1, num2, op2, num3, op3, num4);
                                    count ++;
                                }
                                /* A op ((B op C) op D) */
                                if ((eval(num1, op1, eval(eval(num2, op2,
num3), op3, num4))) == goal){
                                    sprintf(result[count], "%d %c ((%d %c
%d) %c %d)\n", num1, op1, num2, op2, num3, op3, num4);
                                    count ++;
                                }

                                /* A op (B op (C op D)) */
                                if ((eval(num1, op1, eval(num2, op2,
eval(num3, op3, num4)))) == goal){
                                    sprintf(result[count], "%d %c (%d %c (%d
%c %d))\n", num1, op1, num2, op2, num3, op3, num4);
                                    count ++;

                                }
                                /* (A op B) op (C op D) */
                                if ((eval(eval(num1, op1, num2), op2,
eval(num3, op3, num4))) == goal){
                                    sprintf(result[count], "(%d %c %d) %c
(%d %c %d)\n", num1, op1, num2, op2, num3, op3, num4);
                                    count ++;
                                }
                            }
                        }
                    }



                }
            }
        }
    }
    clock_t end = clock();
    printf("%d solutions found! \n",count);


    for (int i = 0; i < count; i ++){
        printf(result[i]);
    }

    double duration = (double) (end - start)/(CLOCKS_PER_SEC * 1000000);
    printf("Time taken to execute (microsecond): %f\n", duration);

    printf("Apakah anda ingin menyimpan hasil dalam file? (y/n) ");
```

```
    char cc;
    scanf(" %c",&cc);
    if (cc == 'y'){
        printf("Masukkan nama file :");
        char name[30];
        scanf("%s", name);
        strcat(name, ".txt");
        FILE *fp;
        fp = fopen(name, "w");
        for (int i = 0; i < count; i++){
            fputs(result[i], fp);
        }
        fclose(fp);
        printf("Hasil berhasil disimpan dalam file.\n");
    }
    else {
        printf("Hasil tidak disimpan pada file.\n");
    }
}

void menu(){
    int cards[4];
    printf("1. Masukkan Kartu\n");
    printf("2. Generate Random Cards\n");
    printf("3. EXIT\n");
    int cc;
    do {
        scanf("%d", &cc);
    } while (cc < 1 || cc > 3);

    if (cc == 1){
        printf("Silakan masukkan 4 buah kartu \neg: A 4 7 Q\n");
        char str[5][3] = {".",".",".",".","." };
        scanf("%s", str[0]);
        scanf("%s", str[1]);
        scanf("%s", str[2]);
        scanf("%s", str[3]);
        if (str[3][0] == '.'){
            printf("Kartu yang kamu masukkan kurang! Harap masukkan 4
kartu.\n");
            return;
        }
        else if (str[4][0] != '.'){
            printf("Kartu yang kamu masukkan terlalu banyak! Harap masukkan
4 kartu.\n");
            return;
        }
        else {
```

```
            for (int i = 0; i < 4; i ++){
                if (str[i][0] == 'J' || str[i][0] == 'j'){
                    cards[i] = 11;
                }
                else if (str[i][0] == 'Q' || str[i][0] == 'q'){
                    cards[i] = 12;
                }
                else if (str[i][0] == 'K' || str[i][0] == 'k'){
                    cards[i] = 13;
                }
                else if (str[i][0] == 'A' || str[i][0] == 'a'){
                    cards[i] = 1;
                }
                else {
                    cards[i] = atoi(str[i]);
                    if (cards[i] < 1 || cards[i] > 10) {
                        printf("Kartu yang kamu masukkan tidak valid! Harap
masukkan kartu yang valid\nKartu yang valid yaitu: A, 1, 2, 3, 4, 5, 6, 7,
8, 9, 10, J, Q, K\n");
                        return;
                    }
                }

            }
        }
        find(cards);
    }
    else if (cc == 2){
        srand((unsigned int) time(NULL));
        int temp;

        printf("Kartu Anda adalah :");
        for (int i = 0; i < 4; i ++){
            temp = 1 + (rand() % 13);
            if (temp == 1){
                printf(" A");
            }
            else if (temp == 11){
                printf(" J");
            }
            else if (temp == 12){
                printf(" Q");
            }
            else if (temp == 13){
                printf(" K");
            }
            else {
                printf(" %d", temp);
```

```
            }
            cards[i] = temp;
        }
        putchar('\n');

        find(cards);
    }
    else if (cc == 3){
        printf("Bye-bye ~\n");
        exit(0);
    }
}

void init(){
    printf("Welcome to 24 Solver !\n");
}

int main(){
    init();
    while (1){
        menu();
    }
}
```

BAB IV
INPUT/OUTPUT PROGRAM

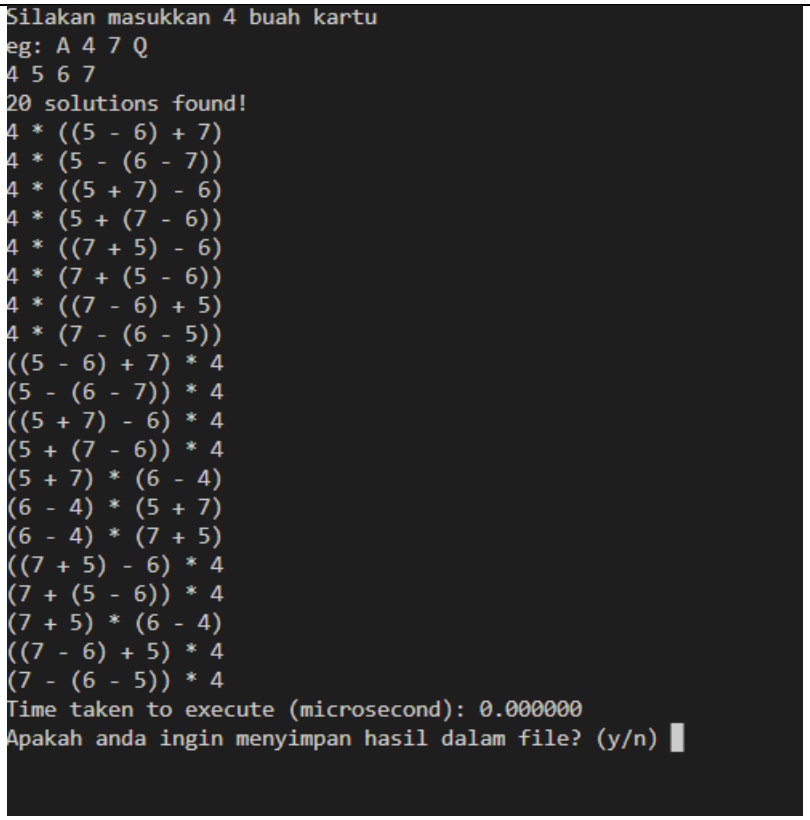
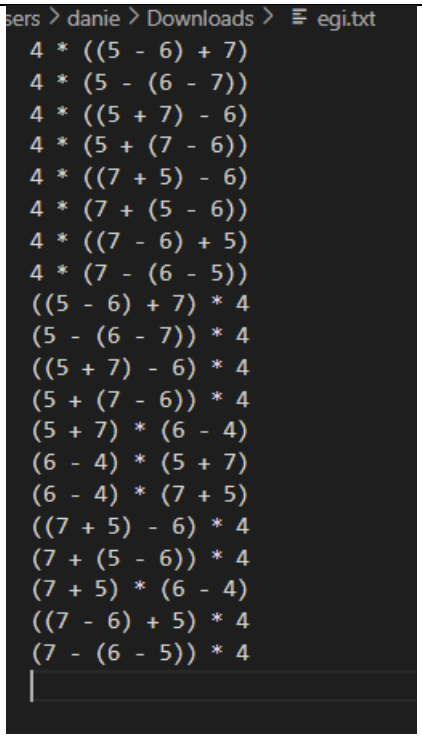
<table>
<tr><th colspan="2">Kondisi 1</th></tr>
<tr><td>Input dari Console</td><td>

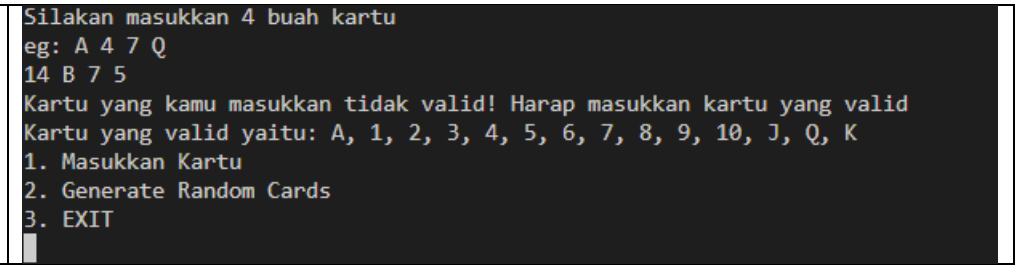
```
Welcome to 24 Solver !
1. Masukkan Kartu
2. Generate Random Cards
3. EXIT
1
Silakan masukkan 4 buah kartu
eg: A 4 7 Q
A 8 7 4
47 solutions found!
1 * (8 * (7 - 4))
(1 * 8) * (7 - 4)
((1 + 7) * 4) - 8
(1 - 7) * (4 - 8)
((1 * 7) - 4) * 8
(1 * (7 - 4)) * 8
1 * ((7 - 4) * 8)
8 * ((1 * 7) - 4)
8 * (1 * (7 - 4))
(8 * 1) * (7 - 4)
(8 : 1) * (7 - 4)
8 * (7 - (1 * 4))
8 * ((7 * 1) - 4)
```

</td></tr>
<tr><th colspan="2">Kondisi 2</th></tr>
<tr><td>Input dengan Generate Random Number</td><td>

```
Time taken to execute (microsecond): 0.000000
Apakah anda ingin menyimpan hasil dalam file? (y/n) n
Hasil tidak disimpan pada file.
1. Masukkan Kartu
2. Generate Random Cards
3. EXIT
2
Kartu Anda adalah : 3 Q Q 3
300 solutions found!
((3 + 12) + 12) - 3
(3 + (12 + 12)) - 3
3 + ((12 + 12) - 3)
3 + (12 + (12 - 3))
```

</td></tr>
<tr><th colspan="2">Kondisi 3</th></tr>
</table>

Output pada console	
Kondisi 4	
Output pada file	
Kondisi 5	

Validasi input	

BAB V
TABEL PENILAIAN

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

BAB IV
REPOSITORY GITHUB

https://github.com/egijago/Tucil1_13521056