LAPORAN TUGAS KECIL 1 IF2211 STRATEGI ALGORITMA

PENYELESAIAN PERMAINAN KARTU 24 DENGAN ALGORITMA BRUTE FORCE

Dosen : Dr. Ir. Rinaldi, M.T.

Disusun oleh :

Muhammad Hanan - 13521041

Sekolah Teknik Elektro dan Informatika

Institut Teknologi Bandung

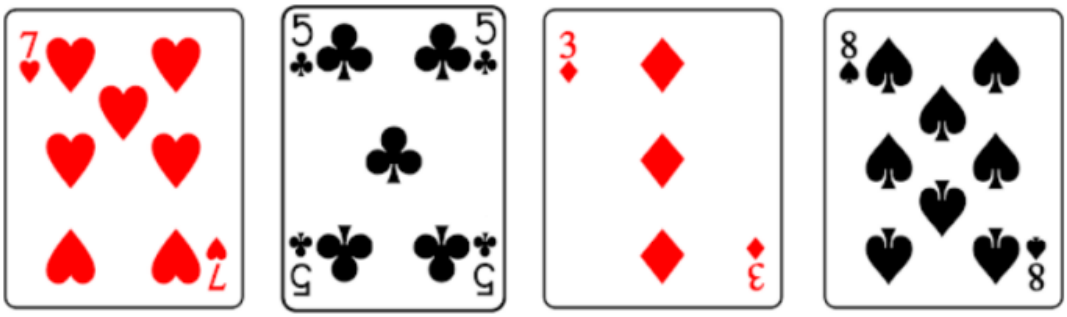
2022/2023

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



MAKE IT 24

BAB II

ALGORITMA

Dalam mengimplementasikan tugas kecil ini, digunakan Algoritma Brute Force untuk menemukan seluruh kemungkinan permutasi yang menghasilkan nilai 24 menggunakan bahasa C. Aspek-aspek yang akan dipermutasikan adalah keempat bilangan yang dipilih, operator-operator yang tersedia, dan sekat/tanda-kurung yang yang memungkinkan dari 4 bilangan.

Permutasi posisi dari keempat bilangan yang diterima/digunakan adalah sebesar 4! atau 24 kemungkinan. Kemudian total operasi yang dapat digunakan adalah sebanyak 4, yaitu tambah(+), kurang(-), kali(*), dan bagi(/). Sehingga total kemungkinan dari urutan operasi adalah sebesar 4^3 atau 64 kemungkinan. Kemudian sekat/tanda-kurung yang memungkinkan ada sebanyak 5, yaitu:

1. (A op B) op (C op D)
2. ((A op B) op C) op D
3. (A op (B op C)) op D
4. A op ((B op C) op D)
5. A op (B op (C op D))

Dimana A,B, C, dan D merupakan bilangan yang diterima/digunakan dan op adalah operator yang dapat digunakan.

Dari penjabaran diatas tadi, dapat ditentukan seluruh kemungkinan yang dapat dibentuk adalah sebesar 4! x 4^3 x 5 atau 7680 kemungkinan dan setiap kemungkinan nya akan dilakukan pengecekan apakah nilai dari setiap kemungkinan tersebut bernilai 24 atau tidak. Program yang disusun ini juga dapat dipastikan tidak memiliki hasil yang identik di setiap kemungkinannya.

BAB III

SOURCE CODE

A. Boolean.h

```c
#ifndef _BOOLEAN_h
#define _BOOLEAN_h

#define boolean unsigned char
#define true 1
#define false 0

#endif
```

Pada file ini adalah bentuk pendefinisian tipe data boolean yang mana tipe data ini tidak ada di dalam bahasa c.

B. Functemp.c

```c
typedef struct {
    float idx[4];
} List;
```

Pendefinisian struct List yang akan digunakan untuk menampung keempat bilangan yang diterima/digunakan.

```c
void welcome() {
    printf("\n");
    printf("\t___  ___      _          _ _       _____   ___  \n");
    printf("\t|  \\/  |      | |        (_) |    / __  \\ /   |\n");
    printf("\t| .  . | __ _| | _____   _| |_  `' / /'/ /| |\n");
    printf("\t| |\\/| |/ _` | |/ / _ \\ | | __|   / / / /_| |\n");
    printf("\t| |  | | (_| |   <  __/ | | |_  ./ /__\\___  |\n");
    printf("\t\\_|  |_/\\__,_|_|\\_\\___| |_|\\__| \\_____/   |_/\n");
    printf("\t                                             \n");
    printf("\t                                             \n");
    printf("\t         _____      _                        \n");
    printf("\t        /  ___|     | |                      \n");
    printf("\t        \\ `--.  ___ | |_  _____ _ __          \n");
    printf("\t         `--. \\/ _ \\| \\ \\ / / _ \\ '__|          \n");
    printf("\t        /\\__/ / (_) | |\\ V /  __/ |           \n");
    printf("\t        \\____/ \\___/|_| \\_/ \\___|_|           \n");
    printf("\t                                             \n\n");
}
```

Prosedur welcome() digunakan untuk memberikan tampilan awal pada saat program baru dijalankan.

```
int cekinp(char x) {
    int cek = false;
    if (x == 'A' || x == 'a' || x == '1') {
        cek = true;
    } else if (x == '2') {
        cek = true;
    } else if (x == '3') {
        cek = true;
    } else if (x == '4') {
        cek = true;
    } else if (x == '5') {
        cek = true;
    } else if (x == '6') {
        cek = true;
    } else if (x == '7') {
        cek = true;
    } else if (x == '8') {
        cek = true;
    } else if (x == '9') {
        cek = true;
    } else if (x == '0') {
        cek = true;
    } else if (x == 'J' || x == 'j') {
        cek = true;
    } else if (x == 'Q' || x == 'q') {
        cek = true;
    } else if (x == 'K' || x == 'k') {
        cek = true;
    }  else if (x == ' ') {
        cek = true;
    }  else if (x == '\n') {
        cek = true;
    } return cek;
}
```

Function cekinp adalah fungsi yang digunakan untuk mengecek inputan yang *available* pada program ini. Memberikan nilai true jika inputannya sesuai dan memberikan nilai false jika tidak.

```
List inputuser() {
    List inp;
    boolean sinput = true;
    int ann = 0;
    while (sinput) {
        char tmp = ' ';
        int cek = true;
        int urutan = 0;
        int skip = 0;
        if (ann != 0) {
            printf("\nSilahkan masukkan 4 kartu!\n>>> ");
        }
        while (tmp != '\n') {
            scanf("%c", &tmp);
            if (cekinp(tmp) == true) {
                if (skip == 1) {
                    skip = 0;
                } else {
                    if ('A' == toupper(tmp)) {
                        inp.idx[urutan] = 1;
                        urutan += 1;
                    } else if ('1' == tmp) {
                        inp.idx[urutan] = 10;
                        urutan += 1;
                        skip = 1;
                    } else if (toupper(tmp) == 'J') {
                        inp.idx[urutan] = 11;
                        urutan += 1;
                    } else if (toupper(tmp) == 'Q') {
                        inp.idx[urutan] = 12;
                        urutan += 1;
                    } else if (toupper(tmp) == 'K') {
                        inp.idx[urutan] = 13;
                        urutan += 1;
                    } else if ('2' == tmp || '3' == tmp || '4' == tmp || '5' == t
                        int tmpp = tmp - '0';
                        inp.idx[urutan] = (float) (tmpp);
                        urutan += 1;
                    }
                }
            } else {
                cek = false;
            }
        }
        if (cek) {
            if (urutan == 4) {
                sinput = false;
            } else {
                if (ann != 0) {
                    printf("Inputan tidak sesuai, silahkan input ulang!!\n");
                }
                ann += 1;
            }
        } else {
            printf("Inputan tidak sesuai, silahkan input ulang!!\n");
        }
    }
    return inp;
}
```

Function inputuser adalah fungsi untuk membaca dan menyimpan inputan 4 bilangan yang user pilih untuk dicarikan kemungkinan solusinya. Fungsi ini akan mengembalikan sebuat list yang berisi 4 bilangan yang diinput oleh user.

```
List randomin() {
    srand(time(0));
    int nilai[4];
    for (int i=0; i<4; i++) {
        nilai[i] = (rand() % 13) + 1;
    }

    List ran;
    printf("\nKartu yang didapatkan adalah ");
    for (int i=0; i<4; i++) {
        if (nilai[i] == 1) {
            printf("A ");
        } else if (nilai[i] == 11) {
            printf("J ");
        } else if (nilai[i] == 12) {
            printf("Q ");
        } else if (nilai[i] == 13) {
            printf("K ");
        } else {
            printf("%d ", nilai[i]);
        }
        ran.idx[i] = nilai[i];
    }
    printf("\n");
    return ran;
}
```

Function randomin adalah fungsi yang akan memberikan 4 bilangan random dari range 1-13 yang akan dicari untuk menemukan solusi yang tersedia.

```
void convtxt(List input) {
    char filename[20] = "../test/";
    char nama[20];
    printf("\nMasukkan nama file txt : ");
    scanf("%s", nama);
    strcat(filename, nama);
    FILE *fp = fopen(filename, "w");
    if (fp == NULL)
    {
        printf("Error opening the file %s", filename);
    }

    float arr[4];
    for (int i=0; i<4; i++) {
        arr[i] = input.idx[i];
    }
```

Prosedur convtxt adalah prosedur yang digunakan untuk membuat file txt yang digunakan untuk menyimpan hasil outputan. Nantinya user akan diminta masukan nama file yang diinginkan dan ditambah dengan format .txt.

```
fprintf(fp, "Kartu yang dimiliki %.0f %.0f %.0f %.0f\n", arr[0],arr[1],arr[2],arr[3]);
int nos = 0; // Number of Solution
int i, j, k, l;
int jml = 0;
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 (i != j && i != k && j != k && i != l && j != l && k != l) {

                    int op1,op2,op3;
                    for (op1 = 1; op1<5; op1++) {
                        for (op2 = 1; op2<5; op2++){
                            for (op3 = 1; op3<5; op3++){

                                float tmp1, tmp2, hasil;
                                char c1,c2,c3;
```

```
// (A op1 B) op2 (C op3 D)
switch (op1) {
    case 1 : tmp1 = arr[i] + arr[j]; c1 = '+'; break;
    case 2 : tmp1 = arr[i] - arr[j]; c1 = '-'; break;
    case 3 : tmp1 = arr[i] * arr[j]; c1 = '*'; break;
    case 4 : tmp1 = arr[i] / arr[j]; c1 = '/'; break;
    default : break;
}
switch (op3) {
    case 1 : tmp2 = arr[k] + arr[l]; c3 = '+'; break;
    case 2 : tmp2 = arr[k] - arr[l]; c3 = '-'; break;
    case 3 : tmp2 = arr[k] * arr[l]; c3 = '*'; break;
    case 4 : tmp2 = arr[k] / arr[l]; c3 = '/'; break;
    default : break;
}
switch (op2) {
    case 1 : hasil = tmp1 + tmp2 ; c2 = '+'; break;
    case 2 : hasil = tmp1 - tmp2 ; c2 = '-'; break;
    case 3 : hasil = tmp1 * tmp2 ; c2 = '*'; break;
    case 4 : hasil = tmp1 / tmp2 ; c2 = '/'; break;
    default : break;
}
if (hasil == 24) {
    fprintf(fp, "(%.0f %c %.0f) %c (%.0f %c %.0f)\n", arr[i], c1, arr[j], c2, arr[k], c3, arr[l]);
    nos += 1;
}
```

```
// ((A op1 B) op2 C) op3 D
switch (op1) {
    case 1 : tmp1 = arr[i] + arr[j]; c1 = '+'; break;
    case 2 : tmp1 = arr[i] - arr[j]; c1 = '-'; break;
    case 3 : tmp1 = arr[i] * arr[j]; c1 = '*'; break;
    case 4 : tmp1 = arr[i] / arr[j]; c1 = '/'; break;
    default : break;
}
switch (op2) {
    case 1 : tmp2 = tmp1 + arr[k]; c2 = '+'; break;
    case 2 : tmp2 = tmp1 - arr[k]; c2 = '-'; break;
    case 3 : tmp2 = tmp1 * arr[k]; c2 = '*'; break;
    case 4 : tmp2 = tmp1 / arr[k]; c2 = '/'; break;
    default : break;
}
switch (op3) {
    case 1 : hasil = tmp2 + arr[l]; c3 = '+'; break;
    case 2 : hasil = tmp2 - arr[l]; c3 = '-'; break;
    case 3 : hasil = tmp2 * arr[l]; c3 = '*'; break;
    case 4 : hasil = tmp2 / arr[l]; c3 = '/'; break;
    default : break;
}
if (hasil == 24) {
    fprintf(fp, "((%.0f %c %.0f) %c %.0f) %c %.0f\n", arr[i], c1, arr[j], c2, arr[k], c3, arr[l]);
    nos += 1;
}
```

```
// (A op1 (B op2 C)) op3 D
switch (op2) {
    case 1 : tmp1 = arr[j] + arr[k]; c2 = '+'; break;
    case 2 : tmp1 = arr[j] - arr[k]; c2 = '-'; break;
    case 3 : tmp1 = arr[j] * arr[k]; c2 = '*'; break;
    case 4 : tmp1 = arr[j] / arr[k]; c2 = '/'; break;
    default : break;
}
switch (op1) {
    case 1 : tmp2 = arr[i] + tmp1; c1 = '+'; break;
    case 2 : tmp2 = arr[i] - tmp1; c1 = '-'; break;
    case 3 : tmp2 = arr[i] * tmp1; c1 = '*'; break;
    case 4 : tmp2 = arr[i] / tmp1; c1 = '/'; break;
    default : break;
}
switch (op3) {
    case 1 : hasil = tmp2 + arr[l]; c3 = '+'; break;
    case 2 : hasil = tmp2 - arr[l]; c3 = '-'; break;
    case 3 : hasil = tmp2 * arr[l]; c3 = '*'; break;
    case 4 : hasil = tmp2 / arr[l]; c3 = '/'; break;
    default : break;
}
if (hasil == 24) {
    fprintf(fp, "(%.0f %c (%.0f %c %.0f)) %c %.0f\n", arr[i], c1, arr[j], c2, arr[k], c3, arr[l]);
    nos += 1;
}
```

```
// A op1 ((B op2 C) op3 D)
switch (op2) {
    case 1 : tmp1 = arr[j] + arr[k]; c2 = '+'; break;
    case 2 : tmp1 = arr[j] - arr[k]; c2 = '-'; break;
    case 3 : tmp1 = arr[j] * arr[k]; c2 = '*'; break;
    case 4 : tmp1 = arr[j] / arr[k]; c2 = '/'; break;
    default : break;
}
switch (op3) {
    case 1 : tmp2 = tmp1 + arr[l]; c3 = '+'; break;
    case 2 : tmp2 = tmp1 - arr[l]; c3 = '-'; break;
    case 3 : tmp2 = tmp1 * arr[l]; c3 = '*'; break;
    case 4 : tmp2 = tmp1 / arr[l]; c3 = '/'; break;
    default : break;
}
switch (op1) {
    case 1 : hasil = arr[i] + tmp2; c1 = '+'; break;
    case 2 : hasil = arr[i] - tmp2; c1 = '-'; break;
    case 3 : hasil = arr[i] * tmp2; c1 = '*'; break;
    case 4 : hasil = arr[i] / tmp2; c1 = '/'; break;
    default : break;
}
if (hasil == 24) {
    fprintf(fp, "%.0f %c ((%.0f %c %.0f) %c %.0f)\n", arr[i], c1, arr[j], c2, arr[k], c3, arr[l]);
    nos += 1;
}
```

```
// A op1 (B op2 (C op3 D))
switch (op3) {
    case 1 : tmp1 = arr[k] + arr[l]; c3 = '+'; break;
    case 2 : tmp1 = arr[k] - arr[l]; c3 = '-'; break;
    case 3 : tmp1 = arr[k] * arr[l]; c3 = '*'; break;
    case 4 : tmp1 = arr[k] / arr[l]; c3 = '/'; break;
    default : break;
}
switch (op2) {
    case 1 : tmp2 = arr[j] + tmp1; c2 = '+'; break;
    case 2 : tmp2 = arr[j] - tmp1; c2 = '-'; break;
    case 3 : tmp2 = arr[j] * tmp1; c2 = '*'; break;
    case 4 : tmp2 = arr[j] / tmp1; c2 = '/'; break;
    default : break;
}
switch (op1) {
    case 1 : hasil = arr[i] + tmp2; c1 = '+'; break;
    case 2 : hasil = arr[i] - tmp2; c1 = '-'; break;
    case 3 : hasil = arr[i] * tmp2; c1 = '*'; break;
    case 4 : hasil = arr[i] / tmp2; c1 = '/'; break;
    default : break;
}
if (hasil == 24) {
    fprintf(fp, "%.0f %c (%.0f %c (%.0f %c %.0f))\n", arr[i], c1, arr[j], c2, arr[k], c3, arr[l]);
    nos += 1;
}
```

```
                                    }
                                }
                            }
                        }
                    }
                }
            }
        }
    }
    fprintf(fp, "\nTotal kemungkinannya adalah sebanyak %i", nos);
    fclose(fp);
```

```
void printsol(List input) {
    float arr[4];
    for (int i=0; i<4; i++) {
        arr[i] = input.idx[i];
    }

    int nos = 0; // Number of Solution
    int i, j, k, l;
    int jml = 0;
    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 (i != j && i != k && j != k && i != l && j != l && k != l) {
```

```
int op1,op2,op3;
for (op1 = 1; op1<5; op1++) {
    for (op2 = 1; op2<5; op2++){
        for (op3 = 1; op3<5; op3++){

            float tmp1, tmp2, hasil;
            char c1,c2,c3;

            // (A op1 B) op2 (C op3 D)
            switch (op1) {
                case 1 : tmp1 = arr[i] + arr[j]; c1 = '+'; break;
                case 2 : tmp1 = arr[i] - arr[j]; c1 = '-'; break;
                case 3 : tmp1 = arr[i] * arr[j]; c1 = '*'; break;
                case 4 : tmp1 = arr[i] / arr[j]; c1 = '/'; break;
                default : break;
            }
            switch (op3) {
                case 1 : tmp2 = arr[k] + arr[l]; c3 = '+'; break;
                case 2 : tmp2 = arr[k] - arr[l]; c3 = '-'; break;
                case 3 : tmp2 = arr[k] * arr[l]; c3 = '*'; break;
                case 4 : tmp2 = arr[k] / arr[l]; c3 = '/'; break;
                default : break;
            }
            switch (op2) {
                case 1 : hasil = tmp1 + tmp2 ; c2 = '+'; break;
                case 2 : hasil = tmp1 - tmp2 ; c2 = '-'; break;
                case 3 : hasil = tmp1 * tmp2 ; c2 = '*'; break;
                case 4 : hasil = tmp1 / tmp2 ; c2 = '/'; break;
                default : break;
            }
            if (hasil == 24) {
                printf("(%.0f %c %.0f) %c (%.0f %c %.0f)\n", arr[i], c1, arr[j], c2, arr[k], c3, arr[l]);
                nos += 1;
            }
```

Prosedur printsol adalah prosedur yang digunakan untuk mengeprint semua solusi yang menungkinkan dari 4 bilangan yang diberikan.

```
// ((A op1 B) op2 C) op3 D
switch (op1) {
    case 1 : tmp1 = arr[i] + arr[j]; c1 = '+'; break;
    case 2 : tmp1 = arr[i] - arr[j]; c1 = '-'; break;
    case 3 : tmp1 = arr[i] * arr[j]; c1 = '*'; break;
    case 4 : tmp1 = arr[i] / arr[j]; c1 = '/'; break;
    default : break;
}
switch (op2) {
    case 1 : tmp2 = tmp1 + arr[k]; c2 = '+'; break;
    case 2 : tmp2 = tmp1 - arr[k]; c2 = '-'; break;
    case 3 : tmp2 = tmp1 * arr[k]; c2 = '*'; break;
    case 4 : tmp2 = tmp1 / arr[k]; c2 = '/'; break;
    default : break;
}
switch (op3) {
    case 1 : hasil = tmp2 + arr[l]; c3 = '+'; break;
    case 2 : hasil = tmp2 - arr[l]; c3 = '-'; break;
    case 3 : hasil = tmp2 * arr[l]; c3 = '*'; break;
    case 4 : hasil = tmp2 / arr[l]; c3 = '/'; break;
    default : break;
}
if (hasil == 24) {
    printf("((%.0f %c %.0f) %c %.0f) %c %.0f\n", arr[i], c1, arr[j], c2, arr[k], c3, arr[l]);
    nos += 1;
}
```

```
// (A op1 (B op2 C)) op3 D
switch (op2) {
    case 1 : tmp1 = arr[j] + arr[k]; c2 = '+'; break;
    case 2 : tmp1 = arr[j] - arr[k]; c2 = '-'; break;
    case 3 : tmp1 = arr[j] * arr[k]; c2 = '*'; break;
    case 4 : tmp1 = arr[j] / arr[k]; c2 = '/'; break;
    default : break;
}
switch (op1) {
    case 1 : tmp2 = arr[i] + tmp1; c1 = '+'; break;
    case 2 : tmp2 = arr[i] - tmp1; c1 = '-'; break;
    case 3 : tmp2 = arr[i] * tmp1; c1 = '*'; break;
    case 4 : tmp2 = arr[i] / tmp1; c1 = '/'; break;
    default : break;
}
switch (op3) {
    case 1 : hasil = tmp2 + arr[l]; c3 = '+'; break;
    case 2 : hasil = tmp2 - arr[l]; c3 = '-'; break;
    case 3 : hasil = tmp2 * arr[l]; c3 = '*'; break;
    case 4 : hasil = tmp2 / arr[l]; c3 = '/'; break;
    default : break;
}
if (hasil == 24) {
    printf("(%.0f %c (%.0f %c %.0f)) %c %.0f\n", arr[i], c1, arr[j], c2, arr[k], c3, arr[l]);
    nos += 1;
}
```

```
// A op1 ((B op2 C) op3 D)
switch (op2) {
    case 1 : tmp1 = arr[j] + arr[k]; c2 = '+'; break;
    case 2 : tmp1 = arr[j] - arr[k]; c2 = '-'; break;
    case 3 : tmp1 = arr[j] * arr[k]; c2 = '*'; break;
    case 4 : tmp1 = arr[j] / arr[k]; c2 = '/'; break;
    default : break;
}
switch (op3) {
    case 1 : tmp2 = tmp1 + arr[l]; c3 = '+'; break;
    case 2 : tmp2 = tmp1 - arr[l]; c3 = '-'; break;
    case 3 : tmp2 = tmp1 * arr[l]; c3 = '*'; break;
    case 4 : tmp2 = tmp1 / arr[l]; c3 = '/'; break;
    default : break;
}
switch (op1) {
    case 1 : hasil = arr[i] + tmp2; c1 = '+'; break;
    case 2 : hasil = arr[i] - tmp2; c1 = '-'; break;
    case 3 : hasil = arr[i] * tmp2; c1 = '*'; break;
    case 4 : hasil = arr[i] / tmp2; c1 = '/'; break;
    default : break;
}
if (hasil == 24) {
    printf("%.0f %c ((%.0f %c %.0f) %c %.0f)\n", arr[i], c1, arr[j], c2, arr[k], c3, arr[l]);
    nos += 1;
}
```

```
// A op1 (B op2 (C op3 D))
switch (op3) {
    case 1 : tmp1 = arr[k] + arr[l]; c3 = '+'; break;
    case 2 : tmp1 = arr[k] - arr[l]; c3 = '-'; break;
    case 3 : tmp1 = arr[k] * arr[l]; c3 = '*'; break;
    case 4 : tmp1 = arr[k] / arr[l]; c3 = '/'; break;
    default : break;
}
switch (op2) {
    case 1 : tmp2 = arr[j] + tmp1; c2 = '+'; break;
    case 2 : tmp2 = arr[j] - tmp1; c2 = '-'; break;
    case 3 : tmp2 = arr[j] * tmp1; c2 = '*'; break;
    case 4 : tmp2 = arr[j] / tmp1; c2 = '/'; break;
    default : break;
}
switch (op1) {
    case 1 : hasil = arr[i] + tmp2; c1 = '+'; break;
    case 2 : hasil = arr[i] - tmp2; c1 = '-'; break;
    case 3 : hasil = arr[i] * tmp2; c1 = '*'; break;
    case 4 : hasil = arr[i] / tmp2; c1 = '/'; break;
    default : break;
}
if (hasil == 24) {
    printf("%.0f %c (%.0f %c (%.0f %c %.0f))\n", arr[i], c1, arr[j], c2, arr[k], c3, arr[l]);
    nos += 1;
}
```

C. main.c

```
int main() {
    welcome();
    boolean game = true;
    while (game) {
        boolean cekpil = true;
        List inp;
        while (cekpil) {
            printf("Pilih metode memilih kartu: \n1. Input Manual\n2. Random\n>>> ");
            int pil;
            scanf("%d", &pil);
            if (pil == 1) {
                inp = inputuser();
                cekpil = false;
            } else if (pil == 2) {
                inp = randomin();
                cekpil = false;
            } else {
                printf("Inputan salah!\n");
            }
        }

        clock_t start, end;
        start = clock();
        float arr[4];
        for (int i=0; i<4; i++) {
            arr[i] = inp.idx[i];
        }
```

```
int nos = 0; // Number of Solution
int i, j, k, l;
int jml = 0;
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 (i != j && i != k && j != k && i != l && j != l && k != l) {

                    int op1,op2,op3;
                    for (op1 = 1; op1<5; op1++) {
                        for (op2 = 1; op2<5; op2++){
                            for (op3 = 1; op3<5; op3++){

                                float tmp1, tmp2, hasil;
                                char c1,c2,c3;
```

```
                    // (A op1 B) op2 (C op3 D)
                    switch (op1) {
                        case 1 : tmp1 = arr[i] + arr[j]; c1 = '+'; break;
                        case 2 : tmp1 = arr[i] - arr[j]; c1 = '-'; break;
                        case 3 : tmp1 = arr[i] * arr[j]; c1 = '*'; break;
                        case 4 : tmp1 = arr[i] / arr[j]; c1 = '/'; break;
                        default : break;
                    }
                    switch (op3) {
                        case 1 : tmp2 = arr[k] + arr[l]; c3 = '+'; break;
                        case 2 : tmp2 = arr[k] - arr[l]; c3 = '-'; break;
                        case 3 : tmp2 = arr[k] * arr[l]; c3 = '*'; break;
                        case 4 : tmp2 = arr[k] / arr[l]; c3 = '/'; break;
                        default : break;
                    }
                    switch (op2) {
                        case 1 : hasil = tmp1 + tmp2 ; c2 = '+'; break;
                        case 2 : hasil = tmp1 - tmp2 ; c2 = '-'; break;
                        case 3 : hasil = tmp1 * tmp2 ; c2 = '*'; break;
                        case 4 : hasil = tmp1 / tmp2 ; c2 = '/'; break;
                        default : break;
                    }
                    if (hasil == 24) {
                        nos += 1;
                    }
```

Fungsi main adalah fungsi yang akan menjalankan program utama.

```
// ((A op1 B) op2 C) op3 D
switch (op1) {
    case 1 : tmp1 = arr[i] + arr[j]; c1 = '+'; break;
    case 2 : tmp1 = arr[i] - arr[j]; c1 = '-'; break;
    case 3 : tmp1 = arr[i] * arr[j]; c1 = '*'; break;
    case 4 : tmp1 = arr[i] / arr[j]; c1 = '/'; break;
    default : break;
}
switch (op2) {
    case 1 : tmp2 = tmp1 + arr[k]; c2 = '+'; break;
    case 2 : tmp2 = tmp1 - arr[k]; c2 = '-'; break;
    case 3 : tmp2 = tmp1 * arr[k]; c2 = '*'; break;
    case 4 : tmp2 = tmp1 / arr[k]; c2 = '/'; break;
    default : break;
}
switch (op3) {
    case 1 : hasil = tmp2 + arr[l]; c3 = '+'; break;
    case 2 : hasil = tmp2 - arr[l]; c3 = '-'; break;
    case 3 : hasil = tmp2 * arr[l]; c3 = '*'; break;
    case 4 : hasil = tmp2 / arr[l]; c3 = '/'; break;
    default : break;
}
if (hasil == 24) {
    nos += 1;
}
```

```
// (A op1 (B op2 C)) op3 D
switch (op2) {
    case 1 : tmp1 = arr[j] + arr[k]; c2 = '+'; break;
    case 2 : tmp1 = arr[j] - arr[k]; c2 = '-'; break;
    case 3 : tmp1 = arr[j] * arr[k]; c2 = '*'; break;
    case 4 : tmp1 = arr[j] / arr[k]; c2 = '/'; break;
    default : break;
}
switch (op1) {
    case 1 : tmp2 = arr[i] + tmp1; c1 = '+'; break;
    case 2 : tmp2 = arr[i] - tmp1; c1 = '-'; break;
    case 3 : tmp2 = arr[i] * tmp1; c1 = '*'; break;
    case 4 : tmp2 = arr[i] / tmp1; c1 = '/'; break;
    default : break;
}
switch (op3) {
    case 1 : hasil = tmp2 + arr[l]; c3 = '+'; break;
    case 2 : hasil = tmp2 - arr[l]; c3 = '-'; break;
    case 3 : hasil = tmp2 * arr[l]; c3 = '*'; break;
    case 4 : hasil = tmp2 / arr[l]; c3 = '/'; break;
    default : break;
}
if (hasil == 24) {
    nos += 1;
}
```

```
// A op1 ((B op2 C) op3 D)
switch (op2) {
    case 1 : tmp1 = arr[j] + arr[k]; c2 = '+'; break;
    case 2 : tmp1 = arr[j] - arr[k]; c2 = '-'; break;
    case 3 : tmp1 = arr[j] * arr[k]; c2 = '*'; break;
    case 4 : tmp1 = arr[j] / arr[k]; c2 = '/'; break;
    default : break;
}
switch (op3) {
    case 1 : tmp2 = tmp1 + arr[l]; c3 = '+'; break;
    case 2 : tmp2 = tmp1 - arr[l]; c3 = '-'; break;
    case 3 : tmp2 = tmp1 * arr[l]; c3 = '*'; break;
    case 4 : tmp2 = tmp1 / arr[l]; c3 = '/'; break;
    default : break;
}
switch (op1) {
    case 1 : hasil = arr[i] + tmp2; c1 = '+'; break;
    case 2 : hasil = arr[i] - tmp2; c1 = '-'; break;
    case 3 : hasil = arr[i] * tmp2; c1 = '*'; break;
    case 4 : hasil = arr[i] / tmp2; c1 = '/'; break;
    default : break;
}
if (hasil == 24) {
    nos += 1;
}
```

```
                        // A op1 (B op2 (C op3 D))
                        switch (op3) {
                            case 1 : tmp1 = arr[k] + arr[l]; c3 = '+'; break;
                            case 2 : tmp1 = arr[k] - arr[l]; c3 = '-'; break;
                            case 3 : tmp1 = arr[k] * arr[l]; c3 = '*'; break;
                            case 4 : tmp1 = arr[k] / arr[l]; c3 = '/'; break;
                            default : break;
                        }
                        switch (op2) {
                            case 1 : tmp2 = arr[j] + tmp1; c2 = '+'; break;
                            case 2 : tmp2 = arr[j] - tmp1; c2 = '-'; break;
                            case 3 : tmp2 = arr[j] * tmp1; c2 = '*'; break;
                            case 4 : tmp2 = arr[j] / tmp1; c2 = '/'; break;
                            default : break;
                        }
                        switch (op1) {
                            case 1 : hasil = arr[i] + tmp2; c1 = '+'; break;
                            case 2 : hasil = arr[i] - tmp2; c1 = '-'; break;
                            case 3 : hasil = arr[i] * tmp2; c1 = '*'; break;
                            case 4 : hasil = arr[i] / tmp2; c1 = '/'; break;
                            default : break;
                        }
                        if (hasil == 24) {
                            nos += 1;
                        }
                    }
                }
            }
        }
    }
}
```

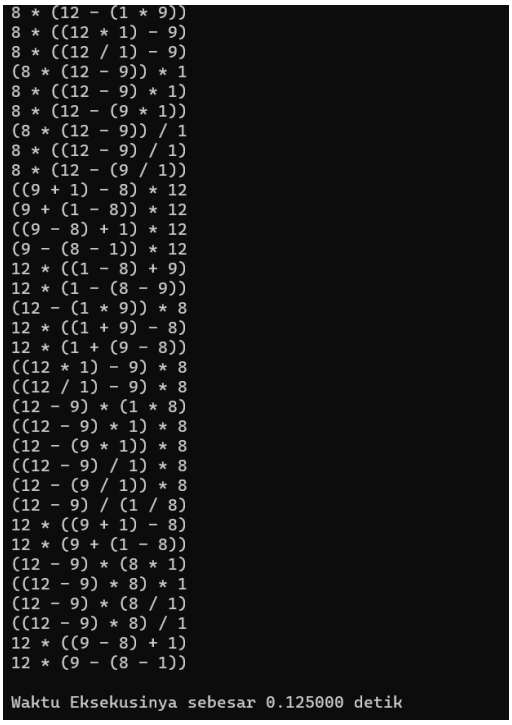
```
    printf("\nTotal kemungkinannya adalah sebanyak %i\n\n", nos);
    printsol(inp);
    end = clock();
    double timespent= (double)(end-start) / CLOCKS_PER_SEC;
    printf("\nWaktu Eksekusinya sebesar %f detik\n\n", timespent);

    boolean cekbf = true;
    while (cekbf) {
        printf("Apakah ingin menyimpan outputannya?\n1. Ya\n2. Tidak\n>>> ");
        int tmppil;
        scanf("%d", &tmppil);
        if (tmppil == 1) {
            convtxt(inp);
            cekbf = false;
        } else if (tmppil == 2) {
            cekbf = false;
        } else {
            printf("Inputan tidak sesuai, silahkan ulangi lagi!\n");
        }
    }
```

```
        boolean ceklas = true;
        while (ceklas) {
            printf("\nApakah ingin mencoba lagi?\n1. Ya\n2. Tidak\n>>> ");
            int tmppil;
            scanf("%d", &tmppil);
            if (tmppil == 1) {
                ceklas = false;
            } else if (tmppil == 2) {
                ceklas = false;
                game = false;
                printf("Terimakasih. Semoga bertemu dilain waktu :D\n");
            } else {
                printf("Inputan tidak sesuai, silahkan ulangi lagi!\n");
            }
        } printf("\n");
    }
}
```

BAB IV

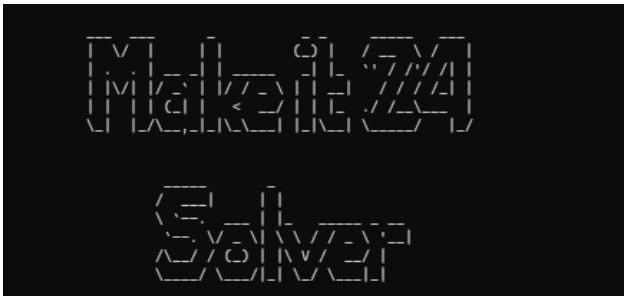
TEST CASE

1. Kasus Kontrol
   a. A 8 9 Q



```
Apakah ingin menyimpan outputannya?
1. Ya
2. Tidak
>>> 1

Masukkan nama file txt : testcase1.txt

Apakah ingin mencoba lagi?
1. Ya
2. Tidak
>>> 2
Terimakasih. Semoga bertemu dilain waktu :D
```

b. 3 5 9 K



```
Pilih metode memilih kartu:
1. Input Manual
2. Random
>>> 1

Silahkan masukkan 4 kartu!
>>> 3 5 9 K

Total kemungkinannya adalah sebanyak 102

(3 + (9 * 13)) / 5
(3 + (13 * 9)) / 5
(5 - 3) + (9 + 13)
((5 - 3) + 9) + 13
(5 - (3 - 9)) + 13
5 - (3 - (9 + 13))
5 - ((3 - 9) - 13)
(5 - 3) + (13 + 9)
((5 - 3) + 13) + 9
(5 - (3 - 13)) + 9
5 - (3 - (13 + 9))
5 - ((3 - 13) - 9)
((5 + 9) - 3) + 13
(5 + (9 - 3)) + 13
5 + ((9 - 3) + 13)
(5 + 9) - (3 - 13)
5 + (9 - (3 - 13))
(5 + 9) + (13 - 3)
((5 + 9) + 13) - 3
(5 + (9 + 13)) - 3
5 + ((9 + 13) - 3)
5 + (9 + (13 - 3))
((5 + 13) - 3) + 9
(5 + (13 - 3)) + 9
5 + ((13 - 3) + 9)
(5 + 13) - (3 - 9)
5 + (13 - (3 - 9))
(5 + 13) + (9 - 3)
((5 + 13) + 9) - 3
(5 + (13 + 9)) - 3
5 + ((13 + 9) - 3)
5 + (13 + (9 - 3))
(9 - 3) + (5 + 13)
((9 - 3) + 5) + 13
(9 - (3 - 5)) + 13
9 - (3 - (5 + 13))
9 - ((3 - 5) - 13)
(9 - 3) + (13 + 5)
((9 - 3) + 13) + 5
(9 - (3 - 13)) + 5
9 - (3 - (13 + 5))
9 - ((3 - 13) - 5)
(9 / 3) * (13 - 5)
9 / (3 / (13 - 5))
((9 + 5) - 3) + 13
(9 + (5 - 3)) + 13
```

```
(9 + (5 - 3)) + 13
9 + ((5 - 3) + 13)
(9 + 5) - (3 - 13)
9 + (5 - (3 - 13))
(9 + 5) + (13 - 3)
((9 + 5) + 13) - 3
(9 + (5 + 13)) - 3
9 + ((5 + 13) - 3)
9 + (5 + (13 - 3))
((9 + 13) - 3) + 5
(9 + (13 - 3)) + 5
9 + ((13 - 3) + 5)
(9 + 13) - (3 - 5)
9 + (13 - (3 - 5))
((9 * 13) + 3) / 5
(9 + 13) + (5 - 3)
((9 + 13) + 5) - 3
(9 + (13 + 5)) - 3
9 + ((13 + 5) - 3)
9 + (13 + (5 - 3))
(9 * (13 - 5)) / 3
9 * ((13 - 5) / 3)
(13 - 3) + (5 + 9)
((13 - 3) + 5) + 9
(13 - (3 - 5)) + 9
13 - (3 - (5 + 9))
13 - ((3 - 5) - 9)
(13 - 3) + (9 + 5)
((13 - 3) + 9) + 5
(13 - (3 - 9)) + 5
13 - (3 - (9 + 5))
13 - ((3 - 9) - 5)
((13 + 5) - 3) + 9
(13 + (5 - 3)) + 9
13 + ((5 - 3) + 9)
(13 + 5) - (3 - 9)
13 + (5 - (3 - 9))
((13 - 5) / 3) * 9
(13 - 5) / (3 / 9)
(13 + 5) + (9 - 3)
((13 + 5) + 9) - 3
(13 + (5 + 9)) - 3
13 + ((5 + 9) - 3)
13 + (5 + (9 - 3))
(13 - 5) * (9 / 3)
((13 - 5) * 9) / 3
((13 + 9) - 3) + 5
(13 + (9 - 3)) + 5
13 + ((9 - 3) + 5)
(13 + 9) - (3 - 5)
13 + (9 - (3 - 5))
((13 * 9) + 3) / 5
(13 + 9) + (5 - 3)
((13 + 9) + 5) - 3
(13 + (9 + 5)) - 3
13 + ((9 + 5) - 3)
13 + (9 + (5 - 3))

Waktu Eksekusinya sebesar 0.237000 detik

Apakah ingin menyimpan outputannya?
1. Ya
2. Tidak
>>> 1

Masukkan nama file txt : testcase2.txt

Apakah ingin mencoba lagi?
1. Ya
2. Tidak
>>> 2
Terimakasih. Semoga bertemu dilain waktu :D
```

c. 5 7 8 J

2. Kasus Random

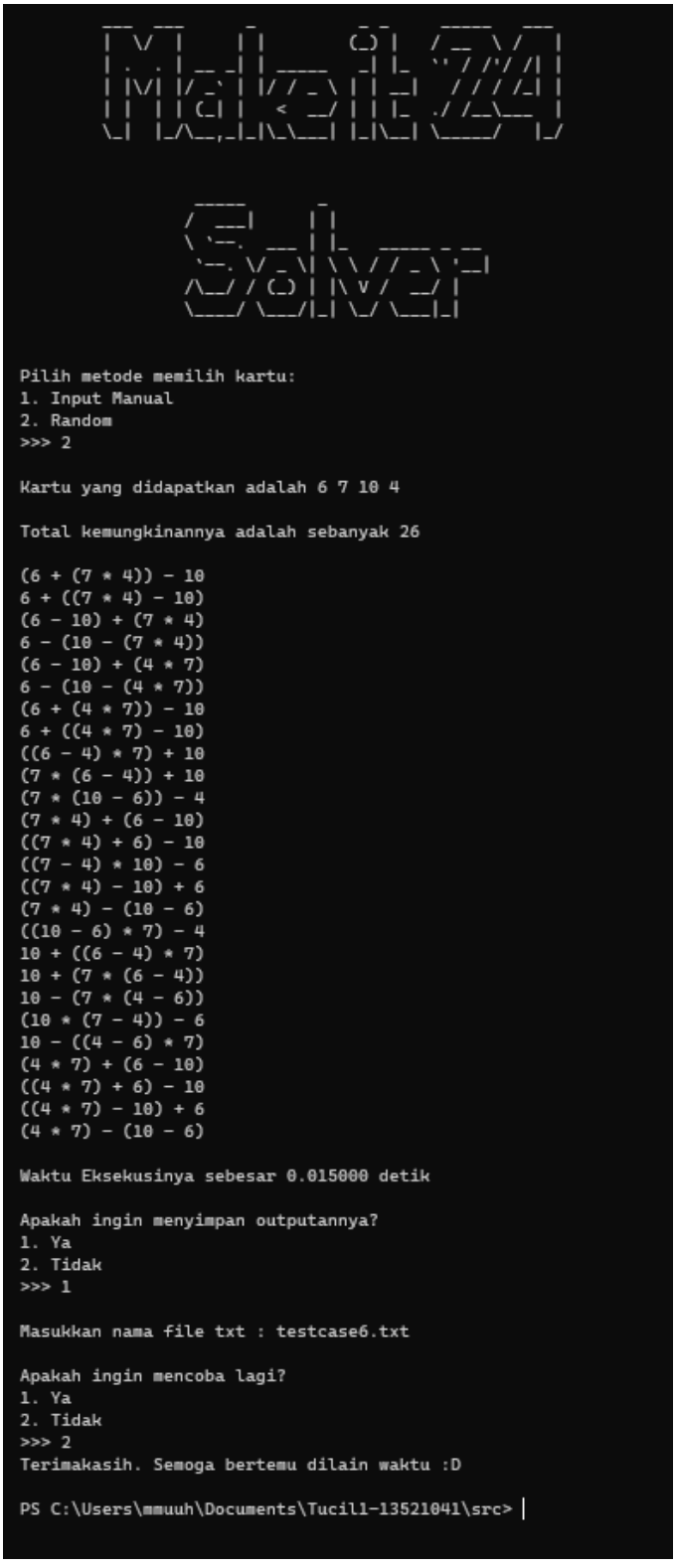
a. 6 5 8 3

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

Waktu Eksekusinya sebesar 0.015000 detik

Apakah ingin menyimpan outputannya?
1. Ya
2. Tidak
>>> 1

Masukkan nama file txt : testcase4.txt

Apakah ingin mencoba lagi?
1. Ya
2. Tidak
>>> 2
Terimakasih. Semoga bertemu dilain waktu :D

PS C:\Users\mmuuh\Documents\Tucil1-13521041\src>
```

b. J 5 Q A

```
 ___  ___      _          _ _      _____    ___
|  \/  |     | |        (_) |    / __  \ /   |
| .  . | __ _| | _____   _| |_   `' / /'/ /| |
| |\/| |/ _` | |/ / _ \ | | __|    / / / /_| |
| |  | | (_| |   <  __/ | | |_   ./ /___\___  |
\_|  |_/\__,_|_|\_\___| |_|\__|  \_____/    |_/


 _____       _
/  ___|     | |
\ `--.  ___ | |_   _____ _ __
 `--. \/ _ \| \ \ / / _ \ '__|
/\__/ / (_) | |\ V /  __/ |
\____/ \___/|_| \_/ \___|_|


Pilih metode memilih kartu:
1. Input Manual
2. Random
>>> 2

Kartu yang didapatkan adalah J 5 Q A

Total kemungkinannya adalah sebanyak 8

((11 - 1) / 5) * 12
(11 - 1) / (5 / 12)
(11 - 1) * (12 / 5)
((11 - 1) * 12) / 5
(12 * (11 - 1)) / 5
12 * ((11 - 1) / 5)
(12 / 5) * (11 - 1)
12 / (5 / (11 - 1))

Waktu Eksekusinya sebesar 0.000000 detik

Apakah ingin menyimpan outputannya?
1. Ya
2. Tidak
>>> 1

Masukkan nama file txt : testcase5.txt

Apakah ingin mencoba lagi?
1. Ya
2. Tidak
>>> 2
Terimakasih. Semoga bertemu dilain waktu :D

PS C:\Users\mmuuh\Documents\Tucil1-13521041\src>
```

c. 6 7 10 4


Pilih metode memilih kartu:
1. Input Manual
2. Random
>>> 2
Kartu yang didapatkan adalah 6 7 10 4
Total kemungkinannya adalah sebanyak 26
(6 + (7 * 4)) - 10
6 + ((7 * 4) - 10)
(6 - 10) + (7 * 4)
6 - (10 - (7 * 4))
(6 - 10) + (4 * 7)
6 - (10 - (4 * 7))
(6 + (4 * 7)) - 10
6 + ((4 * 7) - 10)
((6 - 4) * 7) + 10
(7 * (6 - 4)) + 10
(7 * (10 - 6)) - 4
(7 * 4) + (6 - 10)
((7 * 4) + 6) - 10
((7 - 4) * 10) - 6
((7 * 4) - 10) + 6
(7 * 4) - (10 - 6)
((10 - 6) * 7) - 4
10 + ((6 - 4) * 7)
10 + (7 * (6 - 4))
10 - (7 * (4 - 6))
(10 * (7 - 4)) - 6
10 - ((4 - 6) * 7)
(4 * 7) + (6 - 10)
((4 * 7) + 6) - 10
((4 * 7) - 10) + 6
(4 * 7) - (10 - 6)
Waktu Eksekusinya sebesar 0.015000 detik
Apakah ingin menyimpan outputannya?
1. Ya
2. Tidak
>>> 1
Masukkan nama file txt : testcase6.txt
Apakah ingin mencoba lagi?
1. Ya
2. Tidak
>>> 2
Terimakasih. Semoga bertemu dilain waktu :D
PS C:\Users\mmuuh\Documents\Tucil1-13521041\src>

BAB V

TABLE

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

Lampiran

Link Repository Github: https://github.com/tarsn/Tucil1_13521041