LAPORAN TUGAS KECIL 01

IF2211 STRATEGI ALGORITMA

"Penyelesaian Cyberpunk 2077 Breach Protocol dengan Algoritma *Brute Force*"

Disusun oleh:

Ignatius Jhon Hezkiel Chan

K-01 13522029

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

2023/2024

DAFTAR ISI

BAB 1

DESKRIPSI MASALAH

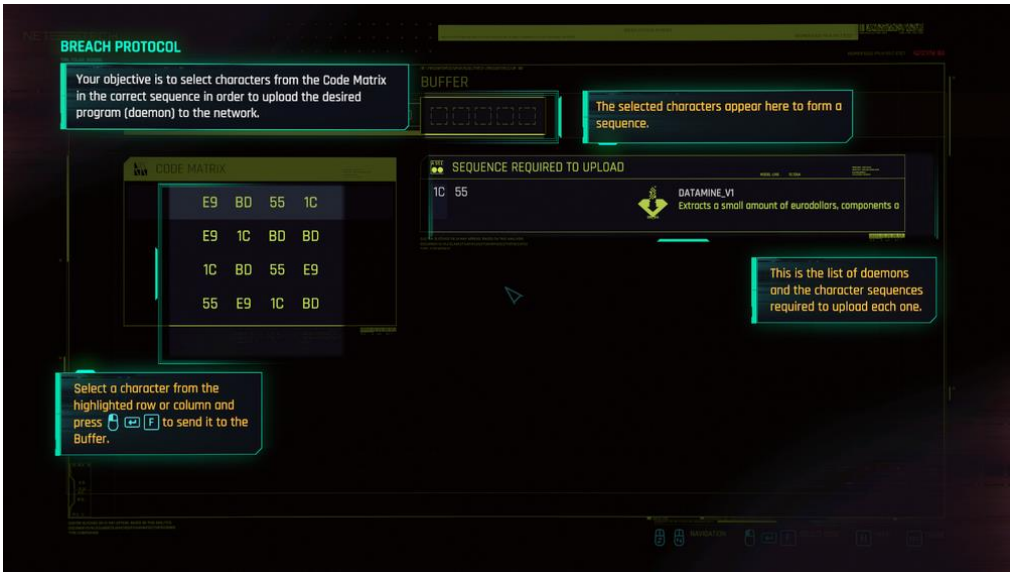
C**yberpunk 2077 Breach Protocol** adalah *minigame* meretas pada permainan video *Cyberpunk 2077*. *Minigame* ini merupakan simulasi peretasan jaringan local dari *ICE (Intrusion Counter-Measures Electronics)* pada permainan *Cyberpunk 2077*. Komponen pada permainan ini antara lain adalah:

1. Token – terdiri dari dua karakter alfanumerik seperti E9, BD, dan 55.
2. Matriks – terdiri atas token-token yang akan dipilih untuk menyusun urutan kode.
3. Sekuens – sebuah rangkaian token (dua atau lebih) yang harus dicocokkan.
4. Buffer – jumlah maksimal token yang dapat disusun secara sekuensial.

Aturan permainan Breach Protocol antara lain:

1. Pemain bergerak dengan pola horizontal, vertikal, horizontal, vertikal (bergantian) hingga semua sekuens berhasil dicocokkan atau buffer penuh.
2. Pemain memulai dengan memilih satu token pada posisi baris paling atas dari matriks.
3. Sekuens dicocokkan pada token-token yang berada di buffer.
4. Satu token pada buffer dapat digunakan pada lebih dari satu sekuens.
5. Setiap sekuens memiliki bobot hadiah atau *reward* yang variatif.
6. Sekuens memiliki panjang minimal berupa dua token.
7. Tiap sekuens hanya bisa dipakai tepat satu kali dalam buffer
8. Token pada suatu titik tidak bisa dipakai lebih dari satu kali dalam buffer

Tugas dalam Breach Protocol ini adalah menemukan solusi buffer yang paling optimal (dengan poin reward tertinggi) untuk suatu kombinasi matriks, sekuens, dan ukuran buffer. Jika didapatkan dua isi buffer dengan total reward yang sama, maka akan diambil isi buffer dengan jumlah token yang lebih sedikit.



Gambar 1 Tampilan Cyberpunk 2077 Breach Protocol
(Sumber: https://cyberpunk-hacker.com/)

BAB 2

STRATEGI PENYELESAIAN

Strategi yang digunakan untuk menyelesaikan *minigame* Cyberpunk 2077 Breach Protocol adalah dengan menggunakan pendekatan *Brute Force*. Program akan mengecek semua jenis kemungkinan isi buffer dari kombinasi matriks, sekuens, dan isi buffer yang diberikan. Kemudian program akan membandingkan nilai total reward tiap isi buffer dan memilih isi buffer dengan nilai total terbesar.

2.1 Fungsi Rekursi Brute Force

Fungsi rekursif akan menjadi sarana kita untuk *brute force* mencari semua kemungkinan isi buffer. Fungsi ini memiliki beberapa parameter sebagai berikut.

a. Isi buffer sekarang
b. Koordinat *tile* sekarang (x,y)
c. Jumlah buffer yang telah dipakai
d. Matriks yang menyimpan nilai Boolean tiap *tile* apakah suatu *tile* sudah dikunjungi atau belum
e. Status pergerakan sekarang, apakah horizontal atau vertical
f. Daftar pergerakan *tile* berisi koordinat-koordinat pergerakan

Fungsi ini kemudian akan melakukan aksi-aksi sebagai berikut.

1. Menambahkan titik sekarang pada daftar pergerakan *tile*
2. Menghitung total poin isi buffer sekarang, dengan melakukan string matching sekuens-sekuens hadiah dengan isi buffer, dan men-totalkan poin yang didapat dari tiap sekuens (jika ditemukan sekuens tersebut sebagai bagian dari isi buffer).
3. Membandingkan total poin tersebut dengan nilai maksimum poin sekarang. Jika total poin buffer lebih besar daripada total poin maksimum sementara, ataupun total poinnya sama tetapi banyak token buffer lebih sedikit, maka kita akan mengganti poin maksmimum sementara, isi buffer maksimum sementara, dan daftar pergerakan *tile* nya dengan isi buffer yang sekarang.
4. Kemudian akan dicek banyak buffer sekarang, apakah sama dengan jumlah maksimal buffer. Jika sama, maka fungsi kemudian berhenti.
5. Jika banyak buffer masih lebih kecil dibandingkan maksimum buffer, kita akan lanjut eksplorasi dengan menambah isi buffer sekarang, sesuai status pergerakan sekarang, horizontal atau vertical.
6. Jika status pergerakan sekarang horizontal, maka kita akan iterasi dari kolom awal sampai kolom terakhir pada baris tersebut. Jika titik tersebut belum dikunjungi, kita tandai titik tersebut menjadi telah dikunjungi, dan kemudian kita panggil fungsi secara rekursi dimana isi buffer ditambah dengan token pada titik tersebut, koordinat diubah menjadi koordinat titik tersebut, banyak buffer ditambah satu, dan status pergerakan diubah menjadi vertical.
7. Jika status pergerakan sekarang vertikal, maka kita akan iterasi dari baris awal sampai baris terakhir pada kolom tersebut. Jika titik tersebut belum dikunjungi, kita tandai titik tersebut menjadi telah dikunjungi, dan kemudian kita panggil fungsi secara rekursi dimana isi buffer ditambah dengan token pada titik tersebut, koordinat diubah menjadi koordinat titik tersebut, banyak buffer ditambah satu, dan status pergerakan diubah menjadi horizontal.

Dengan begini, tiap pemanggilan fungsi akan bercabang-cabang mencoba semua kemungkinan isi buffer dan ketika semua pemanggilan fungsi telah selesai, akan didapatkan isi buffer dengan total poin yang maksimal.

2.2 Algoritma String Matching

String Matching dilakukan secara brute force, dengan membandingkan satu per satu karakter pada isi buffer dan sekuens hadiah. Langkah-langkahnya adalah sebagai berikut.

1. Jika panjang buffer lebih kecil dari panjang sekuens hadiah, maka *strinig matching* gagal
2. Jika panjang buffer lebih besar, kemudian kita akan menetapkan suatu boolean “found” yang menjadi penanda apakah sekuens hadiah sudah ditemukan pada buffer atau belum. Nilai boolean ini akan kita inisialisasi false.
3. Kemudian kita akan iterasi isi buffer mulai dari token pertama, hingga ditemukannya sekuens hadiah pada buffer atau iterasi sudah mencapai token pada indeks ke-(panjang buffer dikurang panjang sekuens hadiah).
4. Dalam iterasi tersebut, kita akan melakukan iterasi lagi untuk pencocokan token per token sekuens hadiah dengan token buffer mulai dari indeks iterasi pertama sekarang. Iterasi kedua ini dilakukan dari awal sekuens hadiah hingga akhir indeks sekuens atau ditemukan ketidakcocokan antara token buffer dengan token sekuens hadiah.
5. Jika ditemukan ketidakcocokan, maka pencocokan sekarang berhenti dan dilanjutkan pencocokan baru dengan indeks iterasi pertama untuk buffer digeser ke kanan satu (indeks ditambah satu). Jika tidak ditemukan ketidakcocokan, maka *string matching* berhasil dan iterasi berhenti.

2.3 Pemanggilan Fungsi Rekursi

Kita akan melakukan pemanggilan fungsi untuk tiap *tile* pada baris pertama matriks. Kita iterasi untuk tiap kolom matriks, mulai dari kolom pertama hingga kolom terakhir, dan dalam tiap iterasi tersebut memanggil fungsi rekursi sesuai dengan parameter tiap *tile*.

BAB 3

IMPLEMENTASI PROGRAM

3.1 FOLDER PRIMITIF (src)

Seluruh fungsionalitas program berada pada main.cpp. Pada file ini berisi semua fungsi-fungsi untuk algoritma searching brute force, string matching, read file, dan randomize info permainan. Berikut adalah tampilan kodenya.

a. Fungsi main

Fungsi main berisi pemanggilan prosedur-prosedur yang menjadi event berjalannya program.

```
int main(){
    mainMenu();
    bruteForceAlgo();
    outputResult();
    saveToFile();
    cout<<"Terima kasih telah menggunakan program kami!"<<endl;
}
```

Gambar 2 Fungsi main

b. Struct dan Variabel Global

```
// Matriks boolean visited
struct StatusVisit{
    bool **visited;
};

// Struct koordinat pada matriks
struct Point{
    int x;
    int y;
};

/*Variabel Global :
    Integer:
    1. maxPoin  : nilai reward maksimal sementara
    2. maxBuffer: nilai maksimal isi buffer
    3. nReward  : banyak sekuens sebagai reward
    4. listPoin : array berisi reward-reward dari sekuens
    5. mtrxWidth: lebar matriks permainan (kolom)
    6. mtrxHeight: tinggi matriks permainan (baris)
    7. minRewLength: panjang sekuens reward yang paling kecil
    8. curMaxLength: panjang sekuens maksimal sementara
    9. exeTime  : waktu eksekusi program

    String:
    1. maxSequence  : sekuens maksimal sementara / isi buffer maksimal sementara
    2. rewardSeq    : array berisi sekuens-sekuens reward
    3. matrix       : matrix permainan

    vector:
    1. maxPoint     : vector (array dinamis) untuk menyimpan tahap-tahapan koordinat maksimal

*/
int maxPoin = -9999, maxBuffer, nReward, *listPoin, mtrxWidth, mtrxHeight, minRewLength = 9999,curMaxLength=0, exeTime;
string maxSequence, *rewardSeq, **matrix;
vector<Point> maxPoint;
```

Gambar 3 Struct dan Variabel Global

c. Fungsi findSeq dan countSeq

Berupa algoritma perhitungan poin buffer dan *string matching*

```
// Algoritma string matching, mencari apakah target ada pada seq
bool findSeq(string seq, string target){
    if(seq.length()<target.length()) return false;
    int targetLength = target.length(),endLim = seq.length()-targetLength,i=0;
    bool found = false;
    while(!found && i<=endLim){
        int j=0;
        bool match = true;
        while(match && j<targetLength){
            if(seq[i+j]!=target[j]){
                match = false;
            }
            j++;
        }
        if(match) found = true;
        i++;
    }
    return found;
}

// Menghitung total reward dari isi buffer seq
int countSeq(string seq){
    int bonus = 0;
    bool foundMatch = false;
    for(int i=0;i<nReward;i++){
        if(findSeq(seq,rewardSeq[i])){
            foundMatch = true;
            bonus += listPoin[i];
        }
    }
    if(!foundMatch) return -9999;
    return bonus;
}
```

Gambar 3 Fungsi findSeq dan countSeq

d. Prosedur searchSol dan bruteForceAlgo

Kedua prosedur ini menjadi prosedur penerapan algoritma brute force untuk mencoba semua kemungkinan isi buffer.

```
// Inisialisasi Matrix visited
void initVisited(StatusVisit *stat){
    stat->visited = new bool*[mtrxHeight];
    for(int i = 0; i < mtrxHeight; i++) {
        stat->visited[i] = new bool[mtrxWidth];
        for(int j = 0; j < mtrxWidth; j++) {
            stat->visited[i][j] = false;
        }
    }
}
```

```
// Brute Force rekursif untuk mencari semua kemungkinan solusi
void searchSol(string seq,int x,int y,int buffer,StatusVisit mat,bool horizontal,vector<Point> points){
    points.push_back({x+1,y+1});
    int curBonus = countSeq(seq);
    if(curBonus>maxPoin || (curBonus==maxPoin && seq.length()<curMaxLength)){
        maxPoin = curBonus;
        maxSequence = seq;
        maxPoint = points;
        curMaxLength = seq.length();
    }
    if(buffer==maxBuffer) return;
    if(horizontal){
        for(int i=0;i<mtrxWidth;i++){
            if(!mat.visited[y][i]){
                string temp = seq+matrix[y][i];
                mat.visited[y][i]=true;
                searchSol(temp,i,y,buffer+1,mat,false,points);
                mat.visited[y][i]=false;
            }
        }
    }else{
        for(int i=0;i<mtrxHeight;i++){
            if(!mat.visited[i][x]){
                string temp = seq+matrix[i][x];
                mat.visited[i][x]=true;
                searchSol(temp,x,i,buffer+1,mat,true,points);
                mat.visited[i][x]=false;
            }
        }
    }
}
```

```
// Initiate rekursif bruteforce
void bruteForceAlgo(){
    StatusVisit stat;
    vector<Point> points;
    initVisited(&stat);
    auto start = high_resolution_clock::now();
    // Memulai eksplorasi dari seluruh tile pada matriks baris pertama
    for(int i=0;i<mtrxWidth;i++){
        stat.visited[0][i] = true;
        searchSol(matrix[0][i],i,0,1,stat,false,points);
        stat.visited[0][i] = false;
    }
    auto stop = high_resolution_clock::now();
    auto duration = duration_cast<milliseconds>(stop-start);
    exeTime = duration.count();
}
```

Gambar 4 Prosedur SearchSol dan bruteForceAlgo

e. Fungsionalitas Pembacaan dari File

Berisi prosedur-prosedur yang meng-handle pembacaan informasi breach protocol dari file berekstensi txt.

```
// Pembacaan dari File
void readMatrixFile(ifstream& inFile){
    matrix = new string*[mtrxHeight];
    for(int i=0;i<mtrxHeight;i++){
        matrix[i] = new string[mtrxWidth];
        for(int j=0;j<mtrxWidth-1;j++){
            getline(inFile,matrix[i][j],' ');
        }
        getline(inFile,matrix[i][mtrxWidth-1]);
    }
}

void readFile(string fileName){
    ifstream inFile;
    string line;
    inFile.open(fileName);
    getline(inFile,line);
    maxBuffer = stoi(line);
    getline(inFile,line,' ');
    mtrxWidth = stoi(line);
    getline(inFile,line);
    mtrxHeight = stoi(line);

    readMatrixFile(inFile);
    getline(inFile,line);
    nReward = stoi(line);
    rewardSeq = new string[nReward];
    listPoin = new int[nReward];
    for(int i=0;i<nReward;i++){
        getline(inFile,line);
        line.erase(remove(line.begin(), line.end(), ' '), line.end());
        if(line.length()<minRewLength) minRewLength = line.length();
        rewardSeq[i] = line;
        getline(inFile,line);
        listPoin[i] = stoi(line);
    }
}
```

Gambar 5 Fungsionalitas Pembacaan dari File

f. Fungsionalitas Randomize Informasi Permainan

```
// Randomize informasi Permainan
void randomizeMatrix(int jmlhToken,string token[]){
    srand((unsigned)time(NULL));
    matrix = new string*[mtrxHeight];
    for(int i=0;i<mtrxHeight;i++){
        matrix[i] = new string[mtrxWidth];
        for(int j=0;j<mtrxWidth;j++){
            int idx = rand()%jmlhToken;
            matrix[i][j] = token[idx];
        }
    }
}

void randomizeSequence(int jmlhToken,string token[],int maxSizeSeq){
    srand((unsigned)time(NULL));
    rewardSeq = new string[nReward];
    listPoin = new int[nReward];
    for(int i=0;i<nReward;i++){
        listPoin[i] = 10 + rand()%91;
        rewardSeq[i]=token[rand()%jmlhToken];
        int sizeSeq = 2 + (rand()%(maxSizeSeq-2));
        for(int j=1;j<sizeSeq;j++){
            rewardSeq[i]+=" "+token[rand()%jmlhToken];
        }
    }
}
```

```
void readRandom(){
    int jmlhToken,maxSizeSeq;
    cout<<"Masukkan jumlah token unik: ";
    cin>>jmlhToken;
    string token[jmlhToken];
    cout<<"Masukkan token-token permainan: ";
    for(int i=0;i<jmlhToken;i++){
        cin>>token[i];
    }

    cout<<"Masukkan ukuran buffer: ";
    cin>>maxBuffer;
    cout<<"Masukkan ukuran matrix (baris dan kolom): ";
    cin>>mtrxHeight>>mtrxWidth;
    cout<<"Masukkan jumlah sekuens: ";
    cin>>nReward;
    cout<<"Masukkan ukuran maksimal sekuens: ";
    cin>>maxSizeSeq;

    randomizeMatrix(jmlhToken,token);
    randomizeSequence(jmlhToken,token,maxSizeSeq);
}
```

Gambar 6 Fungsionalitas Randomize Informasi Permainan

BAB 4

EKSPERIMEN

4.1. Test Case 1

src > tc1.txt

```
7
7 6
EE EE DD BB EE EE CC
DD EE CC BB DD BB CC
BB CC BB BB EE BB AA
AA DD BB BB EE AA EE
BB EE AA BB CC DD CC
DD BB AA BB BB BB BB
3
BB DD
-458
BB EE EE AA
41
AA EE
123
```

src > solTC1.txt

```
164
EE BB EE EE AA EE
1 1
1 3
5 3
5 4
6 4
6 1
```

```
PS C:\Users\user\Downloads\Stima\TUCIL\Tucil1_13522029\src> ./main
Selamat datang pada Cyberpunk 2077 Breach Protocol
Tentukan tipe permainan :
1. Pembacaan informasi permainan dari file txt
2. Informasi permainan otomatis dengan masukan via CLI
>> 1
Masukkan nama file permainan: tc1.txt

Reward maksimal yang didapatkan: 164
Isi buffer: EE BB EE EE AA EE
Tahapan-tahapan pemilihan kotak dalam koordinat:
1 1
1 3
5 3
5 4
6 4
6 1
Waktu eksekusi program: 104 ms
Apakah ingin menyimpan solusi? (y/n): y
Masukkan nama file: solTC1
Terima kasih telah menggunakan program kami!
```

Gambar 7 Test Case 1 melalui read File txt

4.2. Test Case 2

src > tc2.txt

```
8
7 7
1C 1C AS E9 7A XD AS
XD 1C 1C 87 1C AS XD
7A 87 E9 87 7A 87 E9
1C 7A 1C 7A 1C 1C 7A
XD 1C E9 7A BD E9 XD
E9 AS 87 87 XD AS 7A
AS XD BD AS 7A XD 1C
6
7A 87
44
E9 7A 87
60
XD 7A
7
7A 87 7A XD
39
1C BD 1C 87
27
87 AS 1C 87
52
```

src > solTC2.txt

```
156
AS E9 7A 87 AS 1C 87
3 1
3 5
4 5
4 6
2 6
2 2
4 2
```

```
PS C:\Users\user\Downloads\Stima\TUCIL\Tucil1_13522029\src> ./main
Selamat datang pada Cyberpunk 2077 Breach Protocol
Tentukan tipe permainan :
1. Pembacaan informasi permainan dari file txt
2. Informasi permainan otomatis dengan masukan via CLI
>> 1
Masukkan nama file permainan: tc2.txt

Reward maksimal yang didapatkan: 156
Isi buffer: AS E9 7A 87 AS 1C 87
Tahapan-tahapan pemilihan kotak dalam koordinat:
3 1
3 5
4 5
4 6
2 6
2 2
4 2
Waktu eksekusi program: 1910 ms
Apakah ingin menyimpan solusi? (y/n): y
Masukkan nama file: solTC2
Terima kasih telah menggunakan program kami!
```

Gambar 8 Test Case 2 melalui read File txt

4.3.Test Case 3

```
Selamat datang pada Cyberpunk 2077 Breach Protocol
Tentukan tipe permainan :
1. Pembacaan informasi permainan dari file txt
2. Informasi permainan otomatis dengan masukan via CLI
>> 2
Masukkan jumlah token unik: 5
Masukkan token-token permainan: A1 B2 C3 D4 E5
Masukkan ukuran buffer: 6
Masukkan ukuran matrix (baris dan kolom): 8 8
Masukkan jumlah sekuens: 6
Masukkan ukuran maksimal sekuens: 6

Berikut matriks permainan:
D4 A1 E5 D4 C3 C3 C3 B2
C3 C3 C3 E5 A1 B2 B2 C3
D4 A1 B2 D4 C3 C3 B2 A1
B2 D4 D4 B2 A1 B2 A1 D4
B2 E5 A1 B2 E5 C3 C3 E5
D4 A1 B2 C3 C3 D4 A1 D4
C3 E5 E5 C3 D4 B2 D4 B2
D4 B2 A1 D4 B2 B2 D4 D4
Berikut sequens beserta bobotnya masing-masing:
A1 D4 C3 C3 C3 B2
74
C3 E5 A1 B2
34
C3 A1 B2 D4 C3
73
B2 B2
87
D4 A1 B2
42
D4 E5 A1
55

Reward maksimal yang didapatkan: 142
Isi buffer: D4 E5 A1 B2 B2
Tahapan-tahapan pemilihan kotak dalam koordinat:
4 1
4 2
5 2
5 8
2 8
Waktu eksekusi program: 93 ms
Apakah ingin menyimpan solusi? (y/n): y
Masukkan nama file: solTC3
Terima kasih telah menggunakan program kami!
```

Gambar 9 Test Case 3 dengan randomize

4.4.Test Case 4

```
Selamat datang pada Cyberpunk 2077 Breach Protocol
Tentukan tipe permainan :
1. Pembacaan informasi permainan dari file txt
2. Informasi permainan otomatis dengan masukan via CLI
>> 2
Masukkan jumlah token unik: 7
Masukkan token-token permainan: aa bb cc dd ee ff gg
Masukkan ukuran buffer: 7
Masukkan ukuran matrix (baris dan kolom): 8 8
Masukkan jumlah sekuens: 6
Masukkan ukuran maksimal sekuens: 5

Berikut matriks permainan:
dd ff cc dd gg cc dd cc
gg cc ff ee bb bb dd ff
dd gg gg cc ff bb dd dd
ff dd gg dd bb gg ff gg
bb ee aa dd gg bb cc aa
cc ee ff dd bb gg bb ff
ee cc dd dd dd aa ff aa
ee dd gg gg ff gg gg ee
Berikut sequens beserta bobotnya masing-masing:
ff dd gg cc dd
69
gg ff
12
bb dd ff dd
98
gg ff bb
79
dd dd
13
dd gg ff gg
79

Reward maksimal yang didapatkan: 189
Isi buffer: ff gg ff bb dd ff dd
Tahapan-tahapan pemilihan kotak dalam koordinat:
2 1
2 3
5 3
5 2
7 2
7 4
2 4
Waktu eksekusi program: 680 ms
Apakah ingin menyimpan solusi? (y/n): y
Masukkan nama file: solTC4
Terima kasih telah menggunakan program kami!
```

Gambar 10 Test Case 4 dengan randomize

4.5.Test Case 5

src > tc5.txt

```
8
8 8
CC EE BB AA BB BB DD CC
CC CC AA CC EE DD CC AA
EE FF DD DD BB FF AA BB
BB BB BB AA FF BB EE CC
CC FF AA BB DD DD EE CC
DD AA EE EE AA AA AA DD
AA AA AA FF BB CC FF DD
EE BB DD EE AA CC DD FF
5
CC DD DD FF FF EE DD BB CC
639
FF AA AA AA
173
DD AA AA BB EE EE AA
61
FF AA AA EE CC CC
197
AA AA AA DD AA AA BB
995
```

src > solTC5.txt

```
995
CC AA AA AA DD AA AA BB
1 1
1 7
2 7
2 6
8 6
8 2
3 2
3 1
```

```
Selamat datang pada Cyberpunk 2077 Breach Protocol
Tentukan tipe permainan :
1. Pembacaan informasi permainan dari file txt
2. Informasi permainan otomatis dengan masukan via CLI
>> 1
Masukkan nama file permainan: tc5.txt

Reward maksimal yang didapatkan: 995
Isi buffer: CC AA AA AA DD AA AA BB
Tahapan-tahapan pemilihan kotak dalam koordinat:
1 1
1 7
2 7
2 6
8 6
8 2
3 2
3 1
Waktu eksekusi program: 5162 ms
Apakah ingin menyimpan solusi? (y/n): y
Masukkan nama file: solTC5
Terima kasih telah menggunakan program kami!
```

Gambar 11 Test Case 5 melalui read File txt

4.6. Test Case 6

```
PS C:\Users\user\Downloads\Stima\TUCIL\Tucil1_13522029\src> ./main
Selamat datang pada Cyberpunk 2077 Breach Protocol
Tentukan tipe permainan :
1. Pembacaan informasi permainan dari file txt
2. Informasi permainan otomatis dengan masukan via CLI
>> 2
Masukkan jumlah token unik: 6
Masukkan token-token permainan: AA BB CC DD EE FF
Masukkan ukuran buffer: 8
Masukkan ukuran matrix (baris dan kolom): 8 10
Masukkan jumlah sekuens: 6
Masukkan ukuran maksimal sekuens: 7

Berikut matriks permainan:
BB CC AA DD DD EE DD DD FF EE
AA BB AA FF BB EE BB DD EE AA
AA EE EE BB AA FF DD DD AA AA
BB FF DD FF DD AA CC EE CC EE
DD AA FF CC EE FF AA BB FF CC
DD AA BB EE AA DD AA BB FF CC
AA DD BB BB DD FF AA AA DD AA
CC EE AA CC DD BB FF EE FF FF
Berikut sequens beserta bobotnya masing-masing:
CC DD
14
EE DD FF EE AA
11
AA BB EE BB DD EE AA
37
EE BB AA FF DD DD
65
AA FF DD
38
DD CC
87

Reward maksimal yang didapatkan: 190
Isi buffer: EE BB AA FF DD DD CC
Tahapan-tahapan pemilihan kotak dalam koordinat:
6 1
6 8
3 8
3 5
1 5
1 6
10 6
Waktu eksekusi program: 17353 ms
Apakah ingin menyimpan solusi? (y/n): y
Masukkan nama file: solTC6
Terima kasih telah menggunakan program kami!
```

Gambar 12 Test Case 6 dengan randomize

BAB 5

PENUTUP

5.1 Kesimpulan

Melalui tugas besar ini, saya belajar banyak hal terkait algoritma *brute-force* dan bagaimana menerapkannya untuk menyelesaikan suatu permasalahan. Algoritma *brute-force* dapat menyelesaikan hampir semua permasalahan, dimana algoritma ini akan mencoba semua kemungkinan yang tersedia. Karena algoritma yang "nguli" dan mengandalkan kekuatan computer, *brute force* memiliki kompleksitas waktu yang lebih besar dibandingkan algoritma lainnya.

5.2 Saran

- Jangan deadliner dalam mengerjakan tugas

5.3 Link Repository

Link repository untuk tugas kecil 1 mata kuliah IF2211 Strategi Algoritma adalah sebagai berikut

Link : https://github.com/chankiel/Tucil1_13522029

5.4 Tabel Checkpoint Program

Poin	Ya	Tidak
1. Program berhasil dikompilasi tanpa kesalahan	✓	
2. Program berhasil dijalankan	✓	
3. Program dapat membaca masukan berkas .txt	✓	
4. Program dapat menghasilkan masukan secara acak	✓	
5. Solusi yang diberikan program optimal	✓	
6. Program dapat menyimpan solusi dalam berkas .txt	✓	
7. Program memiliki GUI		✓

DAFTAR REFERENSI

https://cplusplus.com/doc/Tutorial/files/

https://www.geeksforgeeks.org/naivealgorithm-for-pattern-searching/