Tugas Kecil 1 IF2211 Strategi Algoritma

Semester II tahun 2023/2024

Penyelesaian *Cyberpunk 2077 Breach Protocol* dengan Algoritma Brute Force

Disusun Oleh

Bryan Cornelius Lauwrence 13522033

PROGRAM STUDI TEKNIK INFORMATIKA

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

2024

DAFTAR ISI

BAB I
DESKRIPSI PERMASALAHAN

Cyberpunk 2077 Breach Protocol adalah *minigame* dalam sebuah permainan video *Cyberpunk* 2077. *Minigame* ini merupakan simulasi peretasan jaringan lokal dari *ICE* (*Intrusion Countermeasures Electronics*) pada permainan *Cyberpunk 2077*. Komponen pada permainan ini, antara lain:

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

Aturan permainan *minigame* ini sebagai berikut:

1. Pemain memulai dengan memilih salah satu token pada posisi baris paling atas pada matriks.
2. Pemain bergerak dengan pola horizontal, vertikal, horizontal, dan seterusnya secara bergantian hingga semua sekuens berhasil dicocokkan atau buffer sudah penuh.
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.

Tugas kecil ini bertujuan untuk merancang algoritma *brute force* untuk menemukan solusi dengan hadiah terbesar dari permainan *Cyberpunk 2077 Breach Protocol*. Pada tugas kali ini, akan digunakan bahasa pemrograman C.

BAB II
ALGORITMA BRUTE FORCE

Algoritma brute force adalah pendekatan yang *straightforward* untuk memecahkan suatu persoalan. Algoritma ini didasarkan pada problem statement dan konsep yang dilibatkan. Algoritma ini akan memecahkan persoalan dengan langsung, jelas, dan sangat sederhana. Pada permasalahan *minigame Cyberpunk 2077 Breach Protocol*, konsep yang digunakan adalah pengecekan hadiah dari seluruh kemungkinan urutan buffer berdasarkan matriks yang tersedia. Kemudian, algoritma akan mengambil buffer dengan hadiah tertinggi yang pertama ditemukan.

Algoritma *brute force* yang dibuat untuk menyelesaikan *minigame* tersebut memiliki langkah-langkah sebagai berikut:

1. Buffer menggunakan konsep *stack*, ketika buffer kosong *push* elemen pada baris pertama kolom pertama.
2. Selama buffer belum penuh, *push* elemen terdekat di arah vertikal atau horizontal (sesuai urutan), yang tidak ada di dalam buffer, dan hitung *reward* dari buffer, sambil membandingkan dengan *reward* maksimal yang telah dihitung.
3. Jika buffer belum penuh, tetapi tidak bisa *push* elemen lain karena sudah terpakai pada buffer, *pop* buffer dan lakukan *push* elemen setelehnya, lalu hitung *reward* dari buffer, sambil membandingkan dengan *reward* maksimal yang telah dihitung.
4. Jika buffer sudah penuh, *pop* buffer dan lakukan *push* elemen setelahnya, lalu hitung *reward* dari buffer, sambil membandingkan dengan *reward* maksimal yang telah dihitung.
5. Jika buffer belum penuh dan kolom atau baris sebelumnya telah dicek semua, *pop* buffer dan *push* elemen selanjutnya, lalu hitung *reward* dari buffer, sambil membandingkan dengan *reward* maksimal yang telah dihitung.
6. Jika seluruh kolom atau baris sudah dicek dan buffer kosong kembali, *push* elemen baris pertama pada kolom berikutnya.
7. Ulangi langkah-langkah sampai seluruh baris pertama telah dicek pada buffer.

BAB III
SOURCE CODE

Definisi ADT

```c
/* Tipe data point */
typedef struct {
    int X;
    int Y;
} Point;
/* Tipe data token */
typedef struct {
    char F; /* Huruf/Angka pertama token */
    char B; /* Huruf/Angka kedua token */
} Token;
/* Tipe data matriks */
typedef struct {
   Token mem[CAP][CAP];
   int rowEff; /* banyaknya/ukuran baris yg terdefinisi */
   int colEff; /* banyaknya/ukuran kolom yg terdefinisi */
} Matrix;
/* Tipe data sekuens */
typedef struct {
    Token contents[CAP];
    int skor; /* Hadiah untuk suatu sekuens */
    int nEff; /* Panjang sekuens */
} Sequence;

typedef struct {
    Sequence fill[CAP];
    int n;
} ListSequence;
/* Tipe data buffer */
typedef struct {
    Point container[CAP]; /* Stack untuk menyimpan point */
    int length;
    int max;
} Buffer;
```

Selektor

```c
/* Selektor Point */
#define ABSIS(P) (P).X
#define ORDINAT(P) (P).Y
/* Selektor Token */
#define FRONT(T) (T).F
#define BACK(T) (T).B
/* Selektor Matriks */
#define ROW_EFF(M) (M).rowEff
#define COL_EFF(M) (M).colEff
#define ELMT_MATRIX(M, i, j) (M).mem[(i)][(j)]
```

```
/* Selektor Sekuens */
#define ELMT_SEQUENCE(S, i) (S).contents[(i)]
#define SKOR(S) (S).skor
#define LENGTH_SEQUENCE(S) (S).nEff

#define ELMT_LS(L, i) (L).fill[(i)]
#define LENGTH_LS(L) (L).n
/* Selektor Buffer */
#define ELMT_BUFFER(B, i) (B).container[(i)]
#define LENGTH_BUFFER(B) (B).length
#define MAX_CAP(B) (B).max
```

Konstruktor dan Primitif Lain

```
void createPoint (int a, int b, Point *p);
/* Membuat sebuah point sebagai koordinat matriks */
/* I.S. titik p sembarang, a merupakan baris dan b merupakan kolom yang
terdefinisi pada suatu matriks */
/* F.S. p terdefinisi sebagai titik (a,b) */

void createToken (char f, char b, Token *T);
/* Membentuk token baru */
/* I.S. f dan b adalah karakter yang valid, T sembarang */
/* F.S. Token T terdefinisi sebagai "fb" */

void createMatrix (int nRows, int nCols, Matrix *m);
/* Membentuk sebuah Matrix "kosong" yang siap diisi berukuran nRow x nCol di
"ujung kiri" memori */
/* I.S. nRow dan nCol adalah valid untuk memori matriks yang dibuat */
/* F.S. Matriks m sesuai dengan definisi di atas terbentuk */

void createEmptySequence (Sequence *s);
/* Membentuk sebuah sekuens "kosong" yang siap diisi */
/* I.S. Sekuens s sembarang */
/* F.S. Sekeuns s terdefinisi dan kosong */

void inputToken (Token *T);
/* Membuat point dari input user */
/* I.S. P sembarang */
/* F.S. P terdefinisi sesuai input user */

void fillSequence (Sequence *s, Token t);
/* Mengisi sekuens s dengan suatu token t di ujung sekuens */
/* I.S. Sequence s terdefinisi kosong atau sudah berisi dan tidak penuh */
/* F.S. Token t tergabung dalam Sequence s */

void fillScoreSequence (Sequence *s, int skor);
/* Mengisi nilai sekuens dengan skor */
/* I.S. Sequence s terdefinisi dengan skor 0 */
/* F.S. Skor sequence didefinisikan sebagai skor */
```

```
void createListOfSequence (ListSequence *l);
/* Membuat list untuk menampung sekuens */
/* I.S. List sequence sembarang */
/* F.S. List sequence terdefinisi kosong */

void pushListOfSequence (ListSequence *l, Sequence s);
/* Mengisi list of sequence dengan sequence s */
/* I.S. List l kosong atau berisi tetapi tidak penuh, s terdefinisi */
/* F.S. s ditambahkan pada list l */

void createEmptyBuffer (Buffer *b, int max_capaxity);
/* Membuat buffer kosong */
/* I.S. Buffer b sembarang */
/* F.S. Buffer b terdefinisi dan kosong */

void pushBuffer (Buffer *b, Point p);
/* Point ditempatkan pada buffer pada posisi paling atas */
/* I.S. Buffer b kosong atau berisi tetapi tidak penuh, point p terdefinisi pada
suatu matriks */
/* F.S. Point p ditambahkan menjadi elemen paling atas dari Buffer b*/

void popBuffer (Buffer *b);
/* Point di posisi palin atas Buffer dikeluarkan */
/* I.S. Buffer tidak kosong */
/* F.S. Elemen buffer berkurang 1, yaitu elemen paling atas */

boolean checkBuffer (Buffer b, Matrix m, Sequence s);
/* Menghasilkan true jika point pada buffer b, yang menunjuk poisisi matriks m,
terdapat sekuens s */

int checkBufferListSequence (Buffer b, Matrix m, ListSequence l);
/* Mengembalikan skor total dari buffer b */

boolean sameSequence (Sequence S1, Sequence S2);
/* Mengembalikan true jika S1 dan S2 adalah sekuens yang sama */

boolean sequenceExist (Sequence S, ListSequence L);
/* Mengembalikan true jika terdapat sequence S pada list L */

boolean samePointToken (Token t, Matrix m, Point p);
/* Mengembalikan true jika point p pada matriks sama dengan token */

boolean pointNotExist (Buffer b, int x, int y);
/* Menghasilkan true jika titik (x,y) belum ada pada buffer b */

void printPoint (Point p);
/* Menampilkan point */

void printToken (Token t);
```

```
/* Menampilkan token */

void printSequence (Sequence s);
/* Menampilkan sekuens*/

void printListOfSequence (ListSequence l);
/* Menampilkan list of sequence */

void printBuffer (Buffer b);
/* Menampilkan buffer */

void printBufferMatrix (Buffer b, Matrix m);
/* Menampilkan elemen buffer b pada matriksnya */

void printMatriks (Matrix m);
/* Menampilkan matriks */
```

Program Pembaca Masukkan dan Penyimpanan

```
void inputFileTXT (int *buffer_size, Matrix *m, ListSequence *l);
/* Membaca masukkan melalui file berkestensi .txt */
/* I.S. buffer_size, Matrix m, dan ListSequence l sembarang */
/* F.S. buffer_size, Matrix m, dan ListSequence l terdefinisi sesuai bacaan file
*/

int findMinimumSequenceSize (int nToken, int nSequence);
/* Mengmebalikan panjang sekuens minimum untuk menghasilkan sekuens yang unik
dengan jumlah token tertentu */

void inputCLI (int *buffer_size, Matrix *m, ListSequence *l);
/* Membaca masukkan melalui command line interface */
/* I.S. buffer_size, Matrix m, dan ListSequence l sembarang */
/* F.S. buffer_size, Matrix m, dan ListSequence l terdefinisi sesuai bacaan
command line */

void saveToTXT (Buffer b, Matrix m, int skor, int time);
/* Menyimpan solusi ke dalam file txt */
```

Program Utama Mencari Solusi Terbaik

```
void findBestOption (Matrix m, ListSequence l, Buffer *b, int *skorAkhir)
/* Mencari solusi optimal */
/* I.S. buffer_size, Matrix m, dan ListSequence l terdefinisi. Buffer b
sembarang */
/* F.S. Buffer b sebagai solusi paling optimal dari matriks m */
{
    /* KAMUS LOKAL */
    int i;
    int idx = 0;
    int ctr[MAX_CAP(*b)];
    Point p;
    Buffer tempBuffer;
```

```
    int skorMaks = 0;
    int skorTemp;
    /* ALGORITMA */
    if (MAX_CAP(*b) > 0) {
        for (i = 0; i < MAX_CAP(*b); i++) {
            ctr[i] = 0;
        }

        for (i = 0; i < LENGTH_LS(l); i++) {
            skorMaks += SKOR(ELMT_LS(l,i));
        }

        createEmptyBuffer(&tempBuffer,MAX_CAP(*b));

        while (ctr[0] < COL_EFF(m) && *skorAkhir != skorMaks) {
            if (idx == 0) {
                createPoint(0,ctr[idx],&p);
                pushBuffer(&tempBuffer,p);
                idx++;
            } else if (idx < MAX_CAP(*b)-1) {
                if (idx % 2 != 0) {
                    if (ctr[idx] != ROW_EFF(m)) {
                        if (pointNotExist(tempBuffer,ctr[idx],ctr[idx-1])) {
                            createPoint(ctr[idx],ctr[idx-1],&p);
                            pushBuffer(&tempBuffer,p);
                            skorTemp = checkBufferListSequence(tempBuffer,m,l);
                            if (skorTemp > *skorAkhir) {
                                *skorAkhir = skorTemp;
                                *b = tempBuffer;
                            }
                            idx++;
                        } else {
                            ctr[idx]++;
                        }
                    } else {
                        popBuffer(&tempBuffer);
                        ctr[idx] = 0;
                        idx--;
                        ctr[idx]++;
                    }
                } else {
                    if (ctr[idx] != COL_EFF(m)) {
                        if (pointNotExist(tempBuffer,ctr[idx-1],ctr[idx])) {
                            createPoint(ctr[idx-1],ctr[idx],&p);
                            pushBuffer(&tempBuffer,p);
                            skorTemp = checkBufferListSequence(tempBuffer,m,l);
                            if (skorTemp > *skorAkhir) {
                                *skorAkhir = skorTemp;
                                *b = tempBuffer;
```

```
                        }
                        idx++;
                    } else {
                        ctr[idx]++;
                    }
                } else {
                    popBuffer(&tempBuffer);
                    ctr[idx] = 0;
                    idx--;
                    ctr[idx]++;
                }
            }
        } else {
            if (idx % 2 != 0) {
                while (ctr[idx] < ROW_EFF(m) && *skorAkhir != skorMaks) {
                    if (pointNotExist(tempBuffer,ctr[idx],ctr[idx-1])) {
                        createPoint(ctr[idx],ctr[idx-1],&p);
                        pushBuffer(&tempBuffer,p);

                        skorTemp = checkBufferListSequence(tempBuffer,m,l);
                        if (skorTemp > *skorAkhir) {
                            *skorAkhir = skorTemp;
                            *b = tempBuffer;
                        }
                        ctr[idx]++;
                        popBuffer(&tempBuffer);
                    } else {
                        ctr[idx]++;
                    }
                }
            } else {
                while (ctr[idx] < COL_EFF(m) && *skorAkhir != skorMaks) {
                    if (pointNotExist(tempBuffer,ctr[idx-1],ctr[idx])) {
                        createPoint(ctr[idx-1],ctr[idx],&p);
                        pushBuffer(&tempBuffer,p);
                        skorTemp = checkBufferListSequence(tempBuffer,m,l);
                        if (skorTemp > *skorAkhir) {
                            *skorAkhir = skorTemp;
                            *b = tempBuffer;
                        }
                        ctr[idx]++;
                        popBuffer(&tempBuffer);
                    } else {
                        ctr[idx]++;
                    }
                }
            }
            ctr[idx] = 0;
            idx--;
```

```
                ctr[idx]++;
                popBuffer(&tempBuffer);
            }
        }
    } else {
        *skorAkhir = 0;
    }
}
```

Program Utama

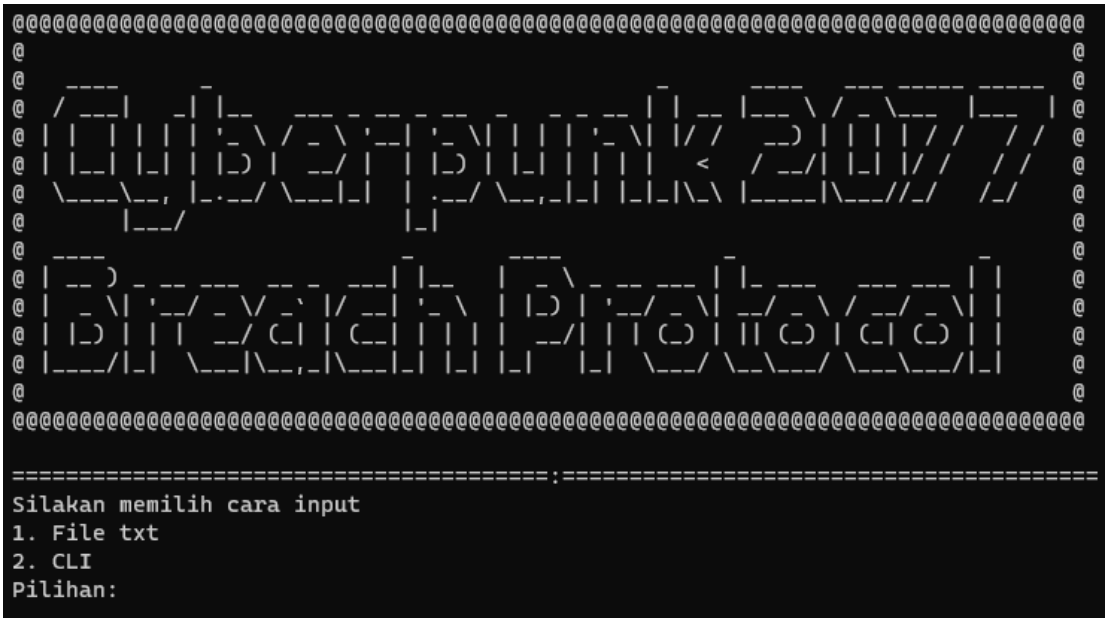
```
#include "data.h"

#ifdef _WIN32
#define CLEAR "cls"
#else
#define CLEAR "clear"
#endif

/* KAMUS */
int pilih;
char yesno;
char kalang;
Matrix m;
ListSequence l;
Buffer b;
Point p;
int bufferSize;
int totalSkor;
/* ALGORITMA */
int main () {
    system(CLEAR);


    printf("@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@@@@@@@@@@@@\n");
    printf("@
          @\n");
    printf("@   ____     _                                 _      ____   ___
_____ _____  @\n");
    printf("@  / ___|   _| |__   ___ _ __ _ __  _   _ _ __ | | __ |___ \\ / _
\\___  |___  | @\n");
    printf("@ | |  | | | | '_ \\ / _ \\ '__| '_ \\| | | | '_ \\| |/ /   __) | |
| | / /   / /  @\n");
    printf("@ | |__| |_| | |_) |  __/ |  | |_) | |_| | | | |   <   / __/| |_| |/
/   / /   @\n");
    printf("@  \\____\\__, |_.__/ \\___|_|  | .__/ \\__,_|_| |_|_|\\_\\
|_____|\\___//_/   /_/    @\n");
    printf("@       |___/                 |_|
         @\n");
```

```
    printf("@  ____                     _       ____            _
  _      @\n");
    printf("@ | __ ) _ __ ___  __ _  ___| |__   |  _ \\ _ __ ___ | |_ ___   ___
___ | |     @\n");
    printf("@ |  _ \\| '__/ _ \\/ _` |/ __| '_ \\  | |_) | '__/ _ \\| __/ _ \\ /
__/ _ \\| |     @\n");
    printf("@ | |_) | | |  __/ (_| | (__| | | | |  __/| | | (_) | || (_) | (_|
(_) | |     @\n");
    printf("@ |____/|_|  \\___|\\__,_|\\___|_| |_| |_|   |_|  \\___/ \\__\\___/
\\___\\___/|_|     @\n");
    printf("@
          @\n");
    printf("@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@@@@@@@@@@@@\n");


    printf("\n========================================:=========================
===============\n");
    do {
        printf("Silakan memilih cara input\n1. File txt\n2. CLI\nPilihan: ");
        scanf("%d",&pilih);
        while (pilih > 2 || pilih < 1) {
            printf("\nTidak ada pilihan tersebut\n");
            printf("Silakan memilih cara input kembali\n1. File txt\n2.
CLI\nPilihan: ");
            scanf("%d",&pilih);
        }
        printf("\n");

        if (pilih == 1) {
            inputFileTXT(&bufferSize,&m,&l);
        } else {
            inputCLI(&bufferSize,&m,&l);
        }
        createEmptyBuffer(&b,bufferSize);

        printf("\nMelakukan perhitungan...\n");
        totalSkor = 0;
        clock_t begin = clock();
        findBestOption(m,l,&b,&totalSkor);
        clock_t end = clock();

        printf("\n\nSkor optimal: %d\n", totalSkor);
        if (totalSkor == 0) {
            printf("Tidak ada solusi\n\n");
        } else {
            printf("Buffer: ");printBufferMatrix(b,m);printf("\n");
            printf("Lokasi pada matriks:\n");printBuffer(b);printf("\n");
        }
        int time = (int)(((double)(end - begin) / CLOCKS_PER_SEC)*1000);
```

```
        printf("Perhitungan dilakukan selama %d ms\n\n", time);

        printf("Apakah ingin menyimpan solusi? (y/n)\n");
        scanf(" %c", &yesno);
        if (yesno == 'y' || yesno == 'Y') {
            saveToTXT(b,m, totalSkor,time);
            printf("\nPenyimpanan selesai.\n");
        } else {
            printf("Tidak melakukan penyimpanan\n");
        }

        printf("\nApakah ingin melakukan perhitungan lain? (y/n)\n");
        scanf(" %c", &kalang);
    } while (kalang == 'y' || kalang == 'Y');
    printf("\nProgram selesai dijalankan\nTerima kasih.\n");
    return 0;
}
```

BAB IV
UJI COBA PROGRAM

Program untuk menyelesaikan permainan *Cyberpunk 2077 Breach Protocol* dikompilasi dengan menjalankan "gcc -o bin/main src/main.c src/data.c" pada direktori utama, jika digunakan sistem operasi Windows. Setelah dikompilasi kemudian dijalankan dengan perintah ".\bin\main.exe", program akan mengeluarkan tampilan awal sebagai berikut.



Gambar 5.1 Tampilan Awal Program

Berikutnya pengguna dapat memilih sesuai pilihan yang tersedia pada menu. Berikut beberapa data uji sebagai contoh input output program.

1. Data Uji 1

```
7
6 6
7A 55 E9 E9 1C 55
55 7A 1C 7A E9 55
55 1C 1C 55 E9 BD
BD 1C 7A 1C 55 BD
BD 55 BD 7A 1C 1C
1C 55 55 7A 55 7A
3
BD E9 1C
15
BD 7A BD
20
BD 1C BD 55
```

Gambar 5.2 Masukkan Data Uji 1

```
Skor optimal: 50
Buffer: 7A BD 7A BD 1C BD 55
Lokasi pada matriks:
1, 1
1, 4
3, 4
3, 5
6, 5
6, 3
1, 3

Perhitungan dilakukan selama 16199 ms

Apakah ingin menyimpan solusi? (y/n)
y

Masukkan nama file untuk penyimpanan (tanpa ekstensi): percobaan1-answer

Melakukan penyimpanan pada file percobaan1-answer.txt

Penyimpanan selesai.

Apakah ingin melakukan perhitungan lain? (y/n)
n

Program selesai dijalankan
Terima kasih.
```

Gambar 5.3 Keluaran Data Uji 1

2. Data Uji 2

```
6
5 5
55 E9 1C 55 E9
BD 7A BD 1C 55
E9 55 E9 BD 1C
7A 7A 1C 55 BD
1C 7A 1C 1C 55
2
E9 55 1C
10
1C 55 7A
15
```

Gambar 5.4 Masukkan Data Uji 2

```
Skor optimal: 25
Buffer: E9 55 1C 55 7A
Lokasi pada matriks:
2, 1
2, 3
5, 3
5, 2
2, 2

Perhitungan dilakukan selama 188 ms

Apakah ingin menyimpan solusi? (y/n)
y

Masukkan nama file untuk penyimpanan (tanpa ekstensi): percobaan2-answer

Melakukan penyimpanan pada file percobaan2-answer.txt

Penyimpanan selesai.
```

Gambar 5.5 Keluaran Data Uji 2

3. Data Uji 3

```
3
5 5
55 1C 55 55 BD
BD 7A BD 1C 55
E9 55 E9 BD 1C
7A 7A 1C 55 BD
1C 7A 1C 1C BD
2
E9 55 1C
10
1C 55 7A
15
```

Gambar 5.6 Masukkan Data Uji 3

```
Skor optimal: 0
Tidak ada solusi

Perhitungan dilakukan selama 15 ms

Apakah ingin menyimpan solusi? (y/n)
y

Masukkan nama file untuk penyimpanan (tanpa ekstensi): percobaan3-answer

Melakukan penyimpanan pada file percobaan3-answer.txt

Penyimpanan selesai.
```

Gambar 5.7 Keluaran Data Uji 3

4. Data Uji 4

```
Silakan memilih cara input
1. File txt
2. CLI
Pilihan: 2

Input dilakukan melalui command line
Masukkan jumlah token unik: 5
Masukkan seluruh token (pisahkan dengan spasi atau enter): a1 b2 c3
d4 e5
Masukkan ukuran buffer: 7
Masukkan ukuran matriks (kolom baris): 5 7
Masukkan jumlah sekuens: 3
Masukkan ukuran maksimal sekuens (harap lebih dari sama dengan 2): 4
```

Gambar 5.8 Masukkan Data Uji 4

```
Matriks yang terbentuk:
c3 a1 c3 a1 d4
a1 a1 c3 d4 d4
e5 e5 a1 b2 e5
d4 e5 c3 b2 b2
d4 d4 d4 c3 a1
b2 d4 c3 a1 c3
d4 b2 c3 a1 b2

Sekuens-sekuens yang terbentuk:
1. b2 e5 e5 (Reward: 10)
2. e5 c3 (Reward: 20)
3. e5 b2 (Reward: 15)

Skor optimal: 45
Buffer: a1 e5 b2 e5 e5 c3
Lokasi pada matriks:
2, 1
2, 4
5, 4
5, 3
1, 3
1, 1

Perhitungan dilakukan selama 3753 ms

Apakah ingin menyimpan solusi? (y/n)
y

Masukkan nama file untuk penyimpanan (tanpa ekstensi): percobaan4-answer

Melakukan penyimpanan pada file percobaan4-answer.txt

Penyimpanan selesai.
```

Gambar 5.9 Keluaran Data Uji 4

5. Data Uji 5

```
10
3 3
55 1C 55
BD 7A BD
E9 BD 7A
2
1C 7A BD E9 7A 55 55
10
55 BD
5
```

Gambar 5.10 Masukkan Data Uji 5

```
Skor optimal: 10
Buffer: 1C 7A BD E9 7A 55 55
Lokasi pada matriks:
2, 1
2, 2
1, 2
1, 3
3, 3
3, 1
1, 1

Perhitungan dilakukan selama 9 ms

Apakah ingin menyimpan solusi? (y/n)
y

Masukkan nama file untuk penyimpanan (tanpa ekstensi): percobaan5-answer

Melakukan penyimpanan pada file percobaan5-answer.txt

Penyimpanan selesai.
```

Gambar 5.11 Keluaran Data Uji 5

6. Data Uji 6

```
Silakan memilih cara input
1. File txt
2. CLI
Pilihan: 2

Input dilakukan melalui command line
Masukkan jumlah token unik: 5
Masukkan seluruh token (pisahkan dengan spasi atau enter): BD 1C 7A 55 E9
Masukkan ukuran buffer: 7
Masukkan ukuran matriks (kolom baris): 6 6
Masukkan jumlah sekuens: 3
Masukkan ukuran maksimal sekuens (harap lebih dari sama dengan 2): 4
```

Gambar 5.12 Masukkan Data Uji 6

```
Matriks yang terbentuk:
1C 1C 55 55 E9 E9
E9 E9 55 7A 55 55
1C E9 7A 55 E9 BD
7A 7A E9 E9 BD BD
7A 1C BD E9 1C 55
BD 1C 1C 7A BD 7A

Sekuens-sekuens yang terbentuk:
1. E9 55 (Reward: 10)
2. 7A E9 1C (Reward: 5)
3. BD 55 BD (Reward: 5)
```

```
Skor optimal: 15
Buffer: 1C E9 55 7A E9 1C
Lokasi pada matriks:
1, 1
1, 2
3, 2
3, 3
2, 3
2, 1

Perhitungan dilakukan selama 18781 ms

Apakah ingin menyimpan solusi? (y/n)
y

Masukkan nama file untuk penyimpanan (tanpa ekstensi): percobaan6-answer

Melakukan penyimpanan pada file percobaan6-answer.txt

Penyimpanan selesai.
```

Gambar 5.13 Keluaran Data Uji 6

7. Data Uji 7

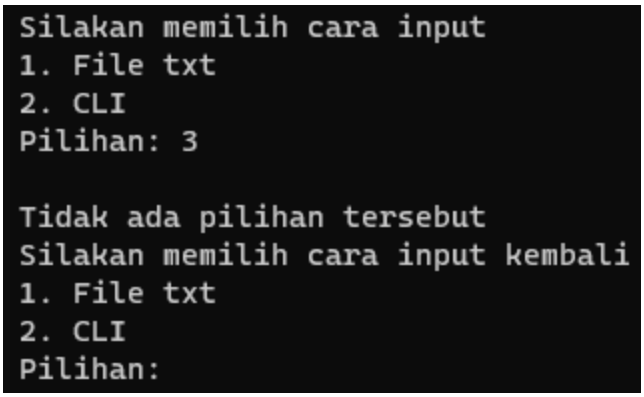
```
0
3 3
55 1C 55
BD 7A BD
E9 BD 7A
2
1C 7A BD E9 7A 55 55
10
55 BD
5
```

Gambar 5.14 Masukkan Data Uji 7

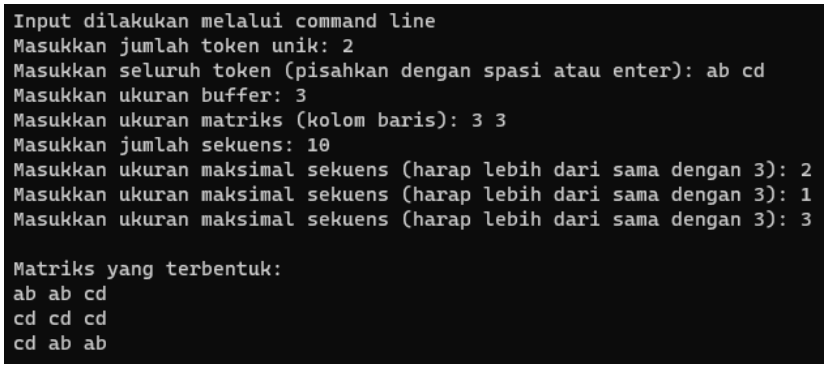
```
Skor optimal: 0
Tidak ada solusi

Perhitungan dilakukan selama 0 ms

Apakah ingin menyimpan solusi? (y/n)
y

Masukkan nama file untuk penyimpanan (tanpa ekstensi): percobaan7-answer

Melakukan penyimpanan pada file percobaan7-answer.txt

Penyimpanan selesai.
```

Gambar 5.15 Keluaran Data Uji 7

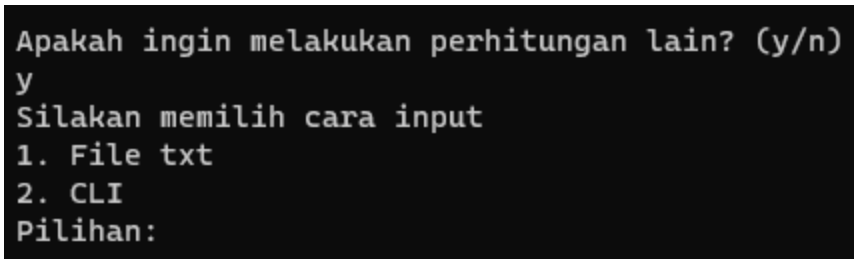
Program juga melakukan validasi terhadap input user. Berikut beberapa tampilan validasi yang dilakukan program.



Gambar 5.16 Validasi Pilihan *Input*



Gambar 5.17 Validasi Ukuran Sekuens Agar Sekuens Unik



Gambar 5.18 Validasi Jika *User* Ingin Melakukan Perhitungan Lain

DAFTAR REFERENSI

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

LAMPIRAN

Tabel Hasil

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

Pranala Github

Repository Github dapat diakses melalui pranala berikut:
https://github.com/BryanLauw/Tucil1_13522033