

**LAPORAN TUGAS KECIL 1
IF2211 STRATEGI ALGORITMA
ALGORITMA BRUTEFORCE
SEMESTER I TAHUN 2023/2024**



Disusun oleh:

Tazkia Nizami

13522032

**PROGRAM STUDI TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG**

2024

BAB I

PENJELASAN ALGORITMA *BRUTE FORCE*

Langkah-langkah algoritma brute force (exhaustive search) untuk mencari solusi penyelesaian Cyberpunk 2077 Breach Protocol Minigame adalah sebagai berikut:

1. Ambil masukkan dari pengguna, bisa melalui file atau gabungan input manual pengguna dan acak.
2. Menentukan setiap *buffer* yang mungkin. Karena ukuran *buffer* berbeda untuk setiap pengujian, maka dapat dimisalkan sebagai sebuah variabel n .
3. Setiap elemen dari *buffer* yang merupakan 2 angka koordinat memiliki kemungkinan dengan jumlah yang bergantung pada ukuran lebar dan tinggi dari matrix. Misalkan untuk ukuran $m \times m$.
4. Kemudian untuk kemungkinan *buffer* selanjutnya, akan dihasilkan setiap kemungkinan koordinat yang mungkin, yaitu sebanyak $m \times m$ kemungkinan
5. Untuk setiap *buffer*, *buffer* akan diperiksa untuk memastikan *buffer* sesuai dengan aturan Cyberpunk 2077 Breach Protocol Minigame.
6. Untuk setiap *buffer* yang *valid* sesuai aturan, akan dihitung reward yang diperoleh dari *buffer* tersebut.
7. Reward tersebut kemudian dibandingkan dengan reward tertinggi yang sudah diperoleh. Jika *buffer* memiliki reward yang lebih tinggi, maka *buffer* tersebut akan menjadi solusi baru untuk kemudian dibandingkan dengan *buffer* selanjutnya
8. Langkah 3 sampai 7 kemudian diulang hingga setiap kemungkinan *buffer* selesai diperiksa dan dibandingkan.

Perhitungan kompleksitas waktu dari algoritma brute force dapat dihitung dengan perhitungan berikut:

1. Setiap koordinat akan memiliki kemungkinan sebanyak m^2 kemungkinan koordinat pada matriks.
2. Untuk setiap sepasang koordinat, akan mengalikan banyak kemungkinan dengan m^2 , sehingga kompleksitas waktunya adalah $m^2 \times m^2$ atau $(m^2)^2$ atau m^{2^2}
3. Untuk ukuran *buffer* sebanyak n , maka kompleksitas waktunya akan menjadi m^{2n}

Karena selama pengujian dan praktik penggunaan program untuk menyelesaikan Cyberpunk 2077 Breach Protocol Minigame bilangan m dan n cenderung dekat (tidak memiliki selisih sampai ratusan dan perbandingan rasio hingga puluhan), maka dapat disimpulkan bahwa $m = n$, sehingga $O(n) = n^{2n}$

BAB II

SOURCE CODE PROGRAM

Kode Program terbagi menjadi beberapa file yang menyimpan fungsi yang memiliki tujuan dan penggunaan yang berbeda. Berikut adalah *source program* untuk menyelesaikan Cyberpunk 2077 Breach Protocol dalam bahasa Go:

A. File main.go

```
package main

import (
    "fmt"
    "time"
)

type coor struct {
    X, Y int
}

// KAMUS GLOBAL
var (
    bufferSize          int
    matrixWidth         int
    matrixHeight        int
    mat                 [][]string
    seqs                [][]string
    rewardSeqs          []int
    banyakSeq           int
    bufferTertinggi     []coor
    bufferTertinggiReward int
    bufferTemp          []coor
    bufferTempReward    int
    stopper             int
    elapsedTime         time.Duration
)

func main() {
    printAsciiArt()
    mainInput()

    // inisiasi variabel
    bufferTertinggi = make([]coor, bufferSize)
    bufferTemp = make([]coor, bufferSize)
    bufferStopper := make([]coor, bufferSize)
    bufferTertinggiReward = 0
}
```

```

bufferTempReward = 0
stopper = 0

fmt.Println("\033[33mMemulai kalkulasi...\033[0m")
startTime := time.Now()
for stopper < 1 {
    // cek apakah bufferTemp valid
    if isUnique(bufferTemp) && isValid(bufferTemp) {
        bufferTempReward = hitungReward(bufferTemp)
        if bufferTempReward > bufferTertinggiReward {
            bufferTertinggiReward = bufferTempReward
            copy(bufferTertinggi, bufferTemp)
        }
    }
    // fmt.Println("bufferTemp: ", bufferTemp)
    bufferTemp = nextBufferTemp(bufferTemp)
    if compareBuffer(bufferTemp, bufferStopper) {
        stopper++
    }
}

elapsedTime = time.Since(startTime)
displayHasil(elapsedTime)
fmt.Println("\033[31mKeluar dari program...\033[0m")
}

```

B. File countReward.go

```

package main

func hitungReward(bufferTemp []coord) int {
    reward := 0
    for i := 0; i < len(bufferTemp); i++ {
        // hitung reward dari bufferTemp
        for j := 0; j < len(seqs); j++ {
            if mat[bufferTemp[i].Y][bufferTemp[i].X] == seqs[j][0] {
                // jika huruf pertama cocok, cek apakah sisa huruf juga cocok
                if cekSeq(i, j) {
                    reward += rewardSeqs[j]
                }
            }
        }
    }
    return reward
}

```

```

}

func cekSeq(buffPos int, seqIdx int) bool {
    seqPos := 1
    // cek apakah sisa huruf juga cocok
    for i := buffPos + 1; i < len(bufferTemp); i++ {
        if mat[bufferTemp[i].Y][bufferTemp[i].X] != seqs[seqIdx][seqPos] {
            return false
        }
        seqPos++
        if seqPos == len(seqs[seqIdx]) {
            return true
        }
    }
    return false
}

```

C. File nextBufferGenerator.go

```

package main

func nextCoor(koor coor, horizontal bool) coor {

    if horizontal {
        if koor.X+1 < matrixWidth {
            koor.X++
        } else {
            return coor{-1, -1}
        }
    } else {
        if koor.Y+1 < matrixHeight {
            koor.Y++
        } else {
            return coor{-1, -1}
        }
    }

    return koor
}

func nextBufferTemp(bufferTemp []coor) []coor {
    // generate bufferTemp selanjutnya
    // validasi bufferTemp
    for j := 0; j < len(bufferTemp)-1; j++ {

```

```

        if bufferTemp[j].X != bufferTemp[j+1].X && j%2 == 1 {
            bufferTemp[j+1] = coor{bufferTemp[j].X, bufferTemp[j+1].Y}
            return bufferTemp
        } else if bufferTemp[j].Y != bufferTemp[j+1].Y && j%2 == 0 {
            bufferTemp[j+1] = coor{bufferTemp[j+1].X, bufferTemp[j].Y}
            return bufferTemp
        }
    }

    // jika sudah valid, generate bufferTemp selanjutnya
    for i := len(bufferTemp) - 1; i >= 0; i-- {
        if i%2 == 1 { // jika i ganjil, maka bufferTemp[i] adalah horizontal
            bufferTemp[i] = nextCoor(bufferTemp[i], true)
            if bufferTemp[i].X == -1 && bufferTemp[i].Y == -1 {
                bufferTemp[i] = coor{0, 0}
            } else {
                return bufferTemp
            }
        } else { // jika i genap, maka bufferTemp[i] adalah vertikal
            bufferTemp[i] = nextCoor(bufferTemp[i], false)
            if bufferTemp[i].X == -1 && bufferTemp[i].Y == -1 {
                bufferTemp[i] = coor{0, 0}
            } else {
                return bufferTemp
            }
        }
    }

    return bufferTemp
}

```

D. outputDriver.go

```

package main

import (
    "fmt"
    "log"
    "os"
    "strconv"
    "time"
)

func displayHasil(elapsedTime time.Duration) {

```

```

// tampilkan hasil
fmt.Println("\033[33m\n--- MENAMPILKAN HASIL ---\033[0m")
if bufferTertinggiReward != 0 {
    fmt.Print("\033[35mReward Tertinggi: \033[0m")
    fmt.Println(bufferTertinggiReward)
    fmt.Print("\033[35mToken Buffer Terbaik: \033[0m")
    for i := 0; i < len(bufferTertinggi); i++ {
        fmt.Print(mat[bufferTertinggi[i].Y][bufferTertinggi[i].X], " ")
    }
    fmt.Println("\033[35\nmBuffer Terbaik: \033[0m")
    for i := 0; i < len(bufferTertinggi); i++ {
        fmt.Printf("%d, %d\n", bufferTertinggi[i].Y+1,
bufferTertinggi[i].X+1)
    }

} else {
    fmt.Println("\033[31mTidak ada hasil yang ditemukan\n\033[0m")
}
fmt.Print("\033[35mWaktu eksekusi: \n\033[0m")
fmt.Println(elapsedTime)

// tawarkan untuk menyimpan hasil ke file txt
fmt.Print("\033[34mApakah Anda ingin menyimpan hasil ke file? (y/n):
\033[0m")
var input string
fmt.Scanln(&input)
if input == "y" {
    saveToFile()
}
}

func saveToFile() {
    // minta user untuk memasukkan nama file .txt
    fmt.Print("\033[34mMasukkan nama file .txt (tanpa .txt): \033[0m")
    var input string
    fmt.Scanln(&input)

    // validasi keberadaan file
    _, err := os.Stat("../test/" + input + ".txt")
    if err == nil {
        fmt.Println("\033[31mFile sudah ada\033[0m")
        saveToFile()
    }

    // buat file
    file, err := os.Create("../test/" + input + ".txt")

```

```

    if err != nil {
        log.Fatal(err)
    }
    defer file.Close()
    // tulis hasil ke file
    if bufferTertinggiReward != 0 {
        file.WriteString(strconv.Itoa(bufferTertinggiReward) + "\n")
        for i := 0; i < len(bufferTertinggi); i++ {
            file.WriteString(mat[bufferTertinggi[i].X][bufferTertinggi[i].Y] + "
")
        }
        file.WriteString("\n")
        for i := 0; i < len(bufferTertinggi); i++ {
            file.WriteString(strconv.Itoa(bufferTertinggi[i].Y+1) + ", " +
strconv.Itoa(bufferTertinggi[i].X+1) + "\n")
        }
        file.WriteString("\n")
        file.WriteString(elapsedTime.String())
    } else {
        file.WriteString("Tidak ada hasil yang ditemukan\n")
        file.WriteString("\n")
        file.WriteString(elapsedTime.String())
    }
    fmt.Println("\033[33mMenyimpan file di folder test...\033[0m")
}

```

E. readInput.go

```

package main

import (
    "bufio"
    "fmt"
    "log"
    "math/rand"
    "os"
    "strconv"
    "strings"
)

func readFile(input string) {
    // buka file input.txt
    f, err := os.Open("../test/" + input + ".txt")

```



```

if err != nil {
    log.Fatal(err)
}
defer f.Close()

scanner := bufio.NewScanner(f)

// baca bufferSize
scanner.Scan()
bufferSize, _ = strconv.Atoi(scanner.Text())

// baca matrixWidth dan matrixHeight
scanner.Scan()
matrixHeight, _ = strconv.Atoi(strings.Split(scanner.Text(), " ")[0])
matrixWidth, _ = strconv.Atoi(strings.Split(scanner.Text(), " ")[1])

// baca matrix
for i := 0; i < matrixHeight; i++ {
    scanner.Scan()
    row := strings.Split(scanner.Text(), " ")
    for j, token := range row {
        if len(token) != 2 {
            fmt.Printf("\033[31mterdapat token yang tidak terdiri dari 2
karakter pada posisi [%d, %d] : %s\n\033[0m", i, j, token)
        }
    }
    mat = append(mat, row)
}

// baca banyakSeq, seqs, dan rewardSeqs
scanner.Scan()
banyakSeq, _ = strconv.Atoi(strings.Split(scanner.Text(), " ")[0])
seqs = make([][]string, banyakSeq)
rewardSeqs = make([]int, banyakSeq)

// baca sekuens dan reward sekuens
for i := 0; i < banyakSeq; i++ {
    scanner.Scan()
    seqs[i] = strings.Split(scanner.Text(), " ")
    scanner.Scan()
    rewardSeqs[i], _ = strconv.Atoi(scanner.Text())
}

if err := scanner.Err(); err != nil {
    log.Fatal(err)
}

```

```

    }
}

func readManualInput() {
    fmt.Print("\033[34mMasukkan banyak token unik: \033[0m")
    banyakTokenUnik := 0
    fmt.Scanln(&banyakTokenUnik)

    fmt.Print("\033[34mMasukkan token-token unik: \033[0m")
    tokens := make([]string, banyakTokenUnik)
    for i := 0; i < banyakTokenUnik; i++ {
        fmt.Scan(&tokens[i])
        for len(tokens[i]) != 2 {
            fmt.Print("\033[31mToken harus terdiri dari 2 karakter. Masukkan
lagi: \033[0m")
            fmt.Scan(&tokens[i])
        }
    }
    fmt.Scanln() // untuk mengakhiri newline

    fmt.Print("\033[34mMasukkan Ukuran Buffer: \033[0m")
    fmt.Scanln(&bufferSize)

    fmt.Print("\033[34mMasukkan Ukuran Matrix (baris kolom): \033[0m")
    fmt.Scanln(&matrixHeight, &matrixWidth)

    fmt.Print("\033[34mMasukkan banyak sekuens: \033[0m")
    fmt.Scanln(&banyakSeq)

    fmt.Print("\033[34mMasukkan panjang maksimal sekuens: \033[0m")
    panjangMaksimalSekuens := 0
    fmt.Scanln(&panjangMaksimalSekuens)

    // bentuk matrix secara acak dengan setiap elemen adalah salah satu token
    unik
    mat = make([][]string, matrixHeight)
    for i := 0; i < matrixHeight; i++ {
        mat[i] = make([]string, matrixWidth)
        for j := 0; j < matrixWidth; j++ {
            mat[i][j] = tokens[rand.Intn(banyakTokenUnik)]
        }
    }
}

// bentuk sekuens secara acak dengan panjang maksimal sekuens yang dimasukkan
seqs = make([][]string, banyakSeq)

```

```

rewardSeqs = make([]int, banyakSeq)
for i := 0; i < banyakSeq; i++ {
    seqLength := rand.Intn(panjangMaksimalSekuens-1) + 2
    seqs[i] = make([]string, seqLength)
    for j := 0; j < seqLength; j++ {
        seqs[i][j] = tokens[rand.Intn(banyakTokenUnik)]
    }
    rewardSeqs[i] = rand.Intn(99) + 1
}

// cetak matrix ke layar
fmt.Println("\033[35mMatrix: \033[0m")
for i := 0; i < matrixHeight; i++ {
    for j := 0; j < matrixWidth; j++ {
        fmt.Print(mat[i][j], " ")
    }
    fmt.Println()
}

// cetak sekuens ke layar
fmt.Println("\033[35mSekuens: \033[0m")
for i := 0; i < banyakSeq; i++ {
    fmt.Print("\033[32mSekuens ", i+1, ": \033[0m")
    for j := 0; j < len(seqs[i]); j++ {
        fmt.Print(seqs[i][j], " ")
    }
    fmt.Println("Reward:", rewardSeqs[i])
}
}

func printAsciiArt() {
    // Open the file
    file, err := os.Open("ascii.txt")
    if err != nil {
        log.Fatal(err)
    }
    defer file.Close()

    // Read the file line by line
    scanner := bufio.NewScanner(file)
    for scanner.Scan() {
        line := scanner.Text()
        fmt.Println("\033[33m" + line + "\033[0m")
    }
}

```

```

    if err := scanner.Err(); err != nil {
        log.Fatal(err)
    }
}

func mainInput() {
    // tanya kepada user apakah ingin menggunakan file input.txt atau tidak
    fmt.Println("\033[34mApakah anda ingin menggunakan file input (.txt)?\n(y/n)\033[0m")
    var input string
    fmt.Scanln(&input)
    if input == "y" {
        // minta user untuk memasukkan nama file .txt
        fmt.Println("\033[33mberalih ke penggunaan input file...\033[0m")
        fmt.Println("\033[34m\npastikan file yang akan dibaca berada di folder test\033[0m")
        fmt.Print("\033[34mMasukkan nama file .txt (tanpa .txt): \033[0m")
        fmt.Scanln(&input)
        // validasi keberadaan file
        _, err := os.Stat("../test/" + input + ".txt")
        if err != nil {
            if os.IsNotExist(err) {
                // pembacaan gagal, keluar dari program
                fmt.Println("\033[31mFile tidak ditemukan\033[0m")
                fmt.Println("\033[31mKeluar dari program...\033[0m")
                os.Exit(1)
            }
        }
        fmt.Println("\033[33mMembaca file...\033[0m")
        readFile(input)
    } else {
        fmt.Println("\033[33mberalih ke penggunaan input manual...\033[0m")
        readManualInput()
    }
}

```

F. validator.go

```

package main

func isUnique(buf []coord) bool {
    //cek apakah setiap koordinat di buf unik
    for i := 0; i < len(buf); i++ {
        for j := i + 1; j < len(buf); j++ {

```

```

        if buf[i] == buf[j] {
            return false
        }
    }
}
return true
}

func isValid(buf [][]coord) bool {
    // cek apakah buf valid
    for i := 1; i < len(buf); i++ {
        if i%2 == 1 {
            if buf[i].Y != buf[i-1].Y {
                return false
            }
        } else {
            if buf[i].X != buf[i-1].X {
                return false
            }
        }
    }
    return true
}

func compareBuffer(buffer1 [][]coord, buffer2 [][]coord) bool {
    //bandingkan bufferTemp dengan bufferTertinggi
    for i := 0; i < len(bufferTemp); i++ {
        if bufferTemp[i] != bufferTertinggi[i] {
            return false
        }
    }
    return true
}

```

HASIL INPUT / OUTPUT

Semua file yang digunakan sebagai input pada bagian ini terdapat pada folder test pada link repository (terlampir)

A. Tampilan Program

[illegible]

B. Test 1

Input:

```
test > test_1.txt
1 6
2 7 7
3 F4 8J B7 KK KK KK B7
4 KK KK B7 B7 8J KK B7
5 B7 F4 8J B7 KK 8J F4
6 F4 F4 KK KK 8J B7 8J
7 8J F4 KK 8J 8J KK 8J
8 8J B7 8J 8J B7 F4 F4
9 KK B7 B7 8J B7 8J 8J
10 3
11 8J 8J F4
12 98
13 B7 8J B7
14 10
15 B7 8J
16 82
```

Output:

```

pastikan file yang akan dibaca berada di folder test
Masukkan nama file .txt (tanpa .txt): test_1
Membaca file...
Memulai kalkulasi...

— MENAMPILKAN HASIL —
Reward Tertinggi: 272
Token Buffer Terbaik: B7 8J F4 F4 8J KK
Buffer Terbaik:
3, 1
3, 3
2, 3
2, 5
4, 5
4, 1
Waktu eksekusi:
12.1115ms
Apakah Anda ingin menyimpan hasil ke file? (y/n):

```

C. Test 2

Input:

```

test > test_2.txt
1 7
2 5 6
3 7U 7U 55 J9 A1 7U
4 K0 A1 A1 AH 7U 7U
5 K0 7U 7U 55 AH K0
6 J9 A1 A1 J9 AH A1
7 A1 K0 55 A1 AH 55
8 3
9 7U 55 A1 7U A1 7U J9 AH
10 69
11 K0 55 AH J9 J9 K0
12 84
13 A1 A1 7U AH
14 20

```

Output:

```

— MENAMPILKAN HASIL —
Reward Tertinggi: 20
Token Buffer Terbaik: 7U K0 A1 7U 7U 55 A1
Buffer Terbaik:
1, 1
1, 2
2, 2
2, 3
3, 3
3, 5
1, 5
Waktu eksekusi:
8.6169ms
Apakah Anda ingin menyimpan hasil ke file? (y/n):

```

D. Test 3

Input:

```

test > test_3.txt
1 6
2 4 4
3 K2 B3 B3 K2
4 B3 K2 A3 K2
5 NN B3 B3 K2
6 K2 A3 A3 NN
7 5
8 NN NN NN NN
9 82
10 K2 K2 NN NN B3
11 52
12 B3 K2
13 42
14 B3 B3
15 9
16 A3 A3
17 44

```

Output:


```

— MENAMPILKAN HASIL —
Reward Tertinggi: 130
Token Buffer Terbaik: K2 B3 K2 B3 K2 K2
Buffer Terbaik:
1, 1
1, 2
2, 2
2, 3
4, 3
4, 2
Waktu eksekusi:
599.9µs
Apakah Anda ingin menyimpan hasil ke file? (y/n): |

```

E. Test 4

Input:

```

test > test_4.txt
1 6
2 6 6
3 8P AB 8P AB 8P K2
4 AB AB 8P AB K2 AB
5 AB AB AB K2 AB K2
6 AB K2 8P K2 AB K2
7 AB AB AB K2 8P AB
8 K2 8P 8P 8P 8P AB
9 3
10 K2 8P 8P AB AB
11 5
12 K2 AB 8P 8P
13 68
14 K2 AB AB 8P
15 98

```

Output:

— MENAMPILKAN HASIL —

Reward Tertinggi: 98

Token Buffer Terbaik: 8P AB AB AB AB AB

Buffer Terbaik:

1, 1

1, 2

4, 2

4, 1

2, 1

2, 3

Waktu eksekusi:

8.3137ms

Apakah Anda ingin menyimpan hasil ke file? (y/n): |

F. Test 5

Input:

test > test_5.txt

1 8

2 8 10

3 A1 A1 B2 C3 A1 C3 A1 C3 B2 A1

4 B2 B2 A1 A1 C3 B2 C3 C3 B2 A1

5 A1 A1 C3 C3 B2 B2 A1 C3 A1 C3

6 A1 B2 B2 A1 A1 A1 B2 C3 B2 C3

7 A1 B2 C3 C3 C3 B2 C3 B2 A1 A1

8 C3 C3 C3 B2 C3 A1 B2 B2 B2 B2

9 A1 B2 B2 A1 A1 B2 A1 B2 B2 B2

10 A1 A1 C3 A1 B2 B2 A1 C3 B2 A1

11 3

12 C3 C3 C3 B2 A1 B2

13 100

14 J3 AB

15 13

16 J3 AB AB

17 26

Output:

```

— MENAMPILKAN HASIL —
Reward Tertinggi: 100
Token Buffer Terbaik: A1 B2 B2 B2 C3 C3 A1 A1
Buffer Terbaik:
1, 1
1, 2
6, 2
6, 3
3, 3
3, 5
1, 5
1, 3
Waktu eksekusi:
5.8266189s
Apakah Anda ingin menyimpan hasil ke file? (y/n):

```

G. Test 6

Input:

```

test > test_6.txt
1      8
2      6 6
3      BB M3 BB LL J4 LL
4      BB M3 J4 J2 BB J2
5      LL LL M3 M3 J4 J4
6      J4 J4 M3 M3 J2 BB
7      LL BB J4 J2 J2 LL
8      BB LL BB M3 LL LL
9      4
10     J4 M3
11     62
12     J2 LL J2 BB J2
13     72
14     J2 J2
15     31
16     LL J4 J4 J2 M3 J2
17     76

```

Output:

— MENAMPILKAN HASIL —

Reward Tertinggi: 217

Token Buffer Terbaik: J4 M3 J4 M3 J4 M3 J2 J2

Buffer Terbaik:

4, 1

4, 3

2, 3

2, 2

4, 2

4, 4

2, 4

2, 6

Waktu eksekusi:

250.7629ms

Apakah Anda ingin menyimpan hasil ke file? (y/n): y

Masukkan nama file .txt (tanpa .txt): testSimpan1

Menyimpan file di folder test...

Keluar dari program...

Uji Penyimpanan:

```
test > testSimpan1.txt
1 217
2 LL M3 LL M3 J2 M3 J4 LL
3 4, 1
4 4, 3
5 2, 3
6 2, 2
7 4, 2
8 4, 4
9 2, 4
10 2, 6
11
12 250.7629ms
```

H. Test 7 (tidak ada solusi)

Input:

```

test > test_7.txt
1 7
2 4 6
3 7A 55 E9 E9 1C 55
4 55 7A 1C 7A E9 55
5 55 1C 1C 55 E9 BD
6 BD 1C 7A 1C 55 BD
7 2
8 BD 7A BD
9 20
10 BD 1C BD 55
11 30

```

Output:

```

— MENAMPILKAN HASIL —
Tidak ada hasil yang ditemukan

Waktu eksekusi:
3.246ms
Apakah Anda ingin menyimpan hasil ke file? (y/n): y
Masukkan nama file .txt (tanpa .txt): testSimpan2
Menyimpan file di folder test...
Keluar dari program...

```

Uji Penyimpanan:

```

test > testSimpan2.txt
1 Tidak ada hasil yang ditemukan
2
3 3.246ms

```

I. Test 8 (random input)

Input:

(wanna be)

n

Masukkan banyak token unik: 6

Masukkan token-token unik: AB K2 NN H3 P0 D3

Masukkan Ukuran Buffer: 7

Masukkan Ukuran Matrix (baris kolom): 7 5

Masukkan banyak sekuens: 3

Masukkan panjang maksimal sekuens: 6

Matrix:

NN NN AB H3 AB

H3 NN D3 AB H3

NN H3 D3 H3 P0

D3 K2 AB K2 K2

P0 K2 NN H3 H3

K2 AB K2 D3 D3

D3 K2 H3 NN D3

Sekuens:

Sekuens 1: NN AB AB AB H3 Reward: 77

Sekuens 2: H3 AB K2 K2 D3 Reward: 82

Sekuens 3: NN AB AB D3 Reward: 98

Memulai kalkulasi ...

Output:

— MENAMPILKAN HASIL —

Reward Tertinggi: 98

Token Buffer Terbaik: NN NN K2 NN AB AB D3

Buffer Terbaik:

1, 1

1, 2

5, 2

5, 3

1, 3

1, 5

6, 5

Waktu eksekusi:

25.2593ms

Apakah Anda ingin menyimpan hasil ke file? (y/n):

BAB IV

LINK REPOSITORY

Berikut link repository Github: https://github.com/TazakiN/Tucil1_13522032

LAMPIRAN

A. Checklist 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		✓