**TUGAS KECIL 1**

**IF2211 STRATEGI ALGORITMA**

Penyelesaian Cyberpunk 2077 Breach Protocol Dengan Algoritma Brute Force

A picture containing text

Description automatically generated

Disusun oleh :

M. Hanief Fatkhan Nashrullah 13522100

**PROGRAM STUDI TEKNIK INFORMATIKA**

**SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA**

**INSTITUT TEKNOLOGI BANDUNG**

**2023**

**Daftar Isi**

[**BAB I : Algoritma Brute Force dalam Breach Protocol** 3](#_Toc158590593)

[**BAB II : Source Program** 5](#_Toc158590594)

[**BAB III : Pengujian dan Hasil Pengujian** 12](#_Toc158590595)

[**Lampiran** 16](#_Toc158590596)

# **BAB I Algoritma Brute Force dalam Breach Protocol**

Breach protocol merupakan minigame yang menjadi salah satu mekanik dalam permainan video Cyberpunk 2077. *Minigame* ini memiliki empat komponen utama, yaitu token, dua karakter alfanumerik, kemudian ada *code matrix*, rangkaian token berbentuk matrix yang akan dicocokkan dengan *sequence*, lalu ada *sequence*, rangkaian yang terdiri dari lebih dari satu token, dan *buffer* sebagai batas jumlah token yang dapat digunakan secara sekuensial.

*Minigame* ini memiliki aturan sebagai berikut :

1. Pencocokkan harus dimulai dari token yang ada pada baris pertama.
2. Token (jalur) yang digunakan harus bergantian dari vertikal dan horizontal dengan jalur pertama dari baris pertama bergerak secara vertikal.
3. Token yang digunakan tidak harus bersebelahan, token bisa digunakan selama tidak bertentangan dengan aturan no.2 dan berada pada baris atau kolom yang sama.
4. Token yang sudah diambil tidak bisa diambil lagi dalam satu kali pencocokan.
5. *Sequence* dicocokkan pada token yang telah digunakan pada *buffer*.
6. Satu *buffer* bisa digunakan oleh beberapa *sequence*.
7. Token yang digunakan tidak boleh melebihi ukuran *buffer.*
8. Setiap *sequence* memiliki bobot *reward* yang dapat bervariasi.



Gambar 1 Minigame Breach Protocol  
( sumber : <https://cyberpunk.fandom.com/wiki/Quickhacking?file=Breach_Protocol_Information_Screenshot_02.png> )

Algoritma *brute force* merupakan algoritma yang cukup sederhana. Algoritma ini memerika satu per satu dari seluruh kemungkinan yang ada untuk mencari solusi yang optimal. Algoritma ini akan digunakan untuk menyelesaikan *Minigame Breach Protocol* tersebut. Langkah langkahnya adalah :

1. Periksa apakah *buffer* yang digunakan sudah penuh dengan token atau belum. Jika sudah maka tidak ada token yang dapat diambil lagi. Jika belum, dapat lanjut ke langkah selanjutnya.
2. Periksa seluruh token dan koordinat yang dapat digunakan dari posisi koordinat awal.
3. Jika ada token yang digunakan, cocokkan dengan *sequence* yang tersedia.
4. Jika terdapat *sequence* yang cocok, periksa apakah *reward* yang diberikan lebih besar dari *reward* awal (default *reward* = 0).
5. Ulangi langkah no.1 dengan posisi koordinat terakhir yang digunakan.
6. Jika sudah tidak ada koordinat dan token yang dapat digunakan dari koordinat terakhir, kembali ke koordinat sebelumnya (*backtrack)* dan lanjut ke koordinat dan token lain.
7. Algoritma berakhir saat semua kemungkinan sudah diperiksa.

# **BAB II Source Program**

Program ini ditulis dengan menggunakan bahasa pemrograman Python

Daftar Module/Library :

* Tkinter
* CustomTkinter
* OS
* sys
* Random
* datetime
* webbrowser

Source code :















# **BAB III: Pengujian dan Hasil Pengujian**

|  |  |
| --- | --- |
| Input File | |
| Input | Output |
| **tc1.txt**  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  30 |  |
| **tc2.txt**  7  6 6  69 42 00 69 00 EF  BE AD BE AD EF 42  42 42 DE 69 EF AD  BE 69 DE BE AD 42  BE 42 EF BE BE DE  00 00 00 EF 00 BE  4  AD 69 42 69 BE  17  69 AD DE AD  100  42 AD 00  97  DE EF AD 00 42  3 |  |
| **tc3.txt**  8  7 7  01 F0 B8 F0 DE DE F0  F0 B8 DE E5 DE F0 AD  E5 DE AD DE DE 01 AD  AD F0 B8 AD E5 E5 01  01 E5 01 F0 01 F0 F0  E5 F0 F0 AD B8 B8 DE  B8 B8 E5 DE E5 DE 01  7  F0 01 B8 F0  38  E5 F0 AD E5 E5  61  E5 B8 B8  25  DE DE 01 DE F0  83  DE 01 F0 F0 DE 01 B8  17  F0 AD DE AD E5 01  -12  F0 01 DE B8 E5 DE  -46 |  |
| **tc4.txt**  8  7 7  B8 69 EE E3 B8 FF E3  EE EA EA 99 FF 69 FF  CC 69 CC EA CC B8 CC  FF FF CA 42 42 42 CC  EE EA CA CA CA B8 99  FF B8 B8 99 69 99 69  EA CC CA B8 CC 69 CA  5  E3 69 B8 69 FF  -61  B8 69 E3 CC B8 42  87  42 EE 42 CA  -79  E3 E3 CC  -17  EE B8 69 FF  -7 |  |
| **tc5.txt**  8  9 9  BE EF DE 69 BE 99 DE AD DE  69 BE DE AD DE DE EF DE 69  69 69 EF EF EE DE EF 99 69  BE EE DE 69 BE 99 EE 99 EF  99 AD EF AD AD EF 69 99 69  BE 69 99 AD EE EF 99 69 69  BE 99 AD 99 99 EF 99 DE BE  69 EE BE DE EF AD 99 AD DE  EF 69 99 DE EE 99 AD DE BE  6  EE EE AD AD  83  EE DE  6  AD BE 99 69 AD 99  99  BE DE  26  EF AD AD DE DE AD 69  22  EF EF  9 |  |
| **invalidmatrixtc.txt**  7  6 6  7A 55 E9 E9  55 7A 1C 7A  55 1C 1C 55  BD 1C 7A 1C  3  BD E9 1C  15  BD 7A BD  20  BD 1C BD 55  30 |  |
| Randomized Input | |
| Input | Output |
|  |  |
|  |  |

# **Lampiran**

1. *Checklist*

|  |  |  |
| --- | --- | --- |
| 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 | ✔ |  |

1. Pranala *repository*

<https://github.com/hannoobz/Tucil1_13522100>