

IMPLEMENTASI ALGORITMA GREEDY DALAM PEMECAHAN BOT PERMAINAN DIAMOND

LAPORAN TUGAS BESAR



Oleh:

Kelompok 13 - KODE KARMA

Aditya Hot Martua Sihite 123140114

Andre Prasetya Daely 123140131

Exaudi Amin Hutasoit 123140161

Dosen Pengampu: Winda Yulita, M.Cs.

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS TEKNOLOGI INDUSTRI

INSTITUT TEKNOLOGI SUMATERA

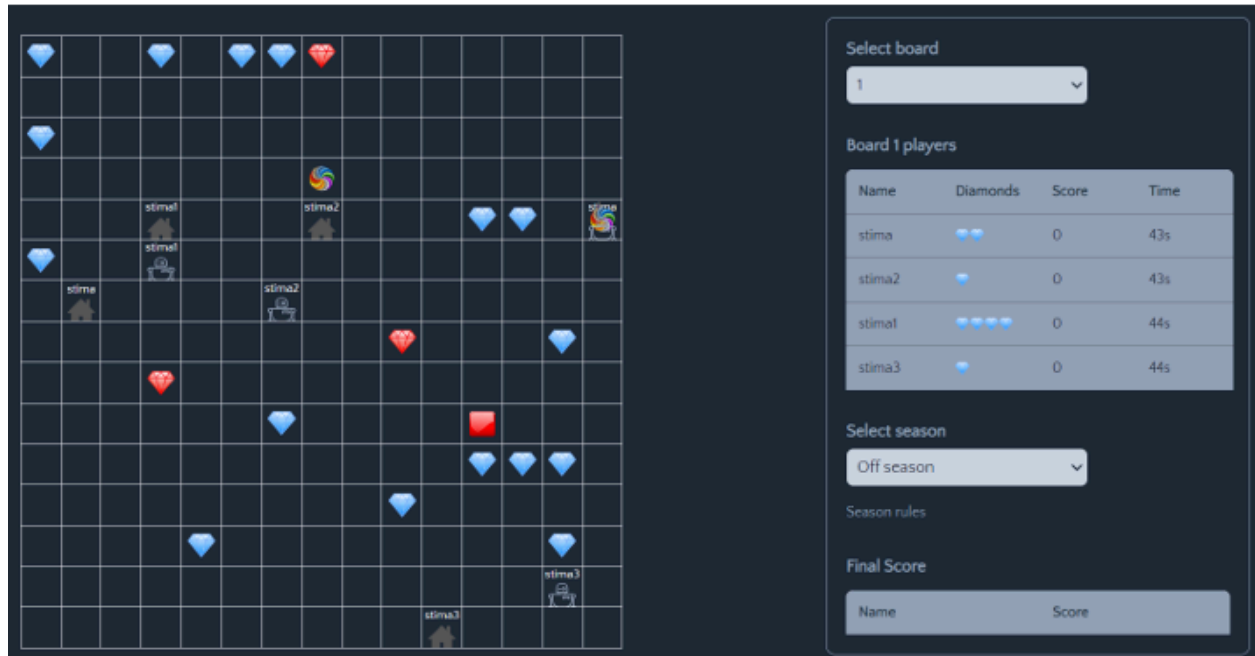
2025

DAFTAR ISI

BAB I DESKRIPSI TUGAS.....	3
BAB II LANDASAN TEORI.....	9
2.1 Dasar Teori.....	9
2.2 Game Engine Diamonds.....	9
2.3 Struktur Game Engine.....	10
2.4 Bot Engine Diamonds.....	10
2.5 Alur Permainan Diamonds.....	11
2.5.1 Menjalankan Game Engine Diamonds.....	11
2.5.2 Mengembangkan Bot Engine Permainan Diamonds.....	12
2.6 Cara Kerja Program.....	13
BAB III APLIKASI STRATEGI GREEDY.....	16
3.1 Proses Mapping Persoalan Diamonds ke Elemen-Elemen Algoritma Greedy.....	16
3.2 Eksplorasi Alternatif Solusi Greedy.....	17
3.3 Analisis Efisiensi dan Efektivitas Solusi Greedy.....	18
3.4 Strategi Greedy yang Dipilih.....	19
BAB IV IMPLEMENTASI DAN PENGUJIAN.....	20
4.1 Implementasi Algoritma Greedy pada Program Bot.....	20
4.2 Penjelasan Struktur Data pada Program.....	21
4.3 Analisis Desain Solusi Algoritma.....	22
BAB V KESIMPULAN DAN SARAN.....	24
5.1 Kesimpulan.....	24
5.2 Saran.....	24
LAMPIRAN.....	25
DAFTAR PUSTAKA.....	26

BAB I

DESKRIPSI TUGAS



Gambar 1.1. Permainan Diamonds

Diamonds merupakan suatu programming challenge yang mempertandingkan bot yang anda buat dengan bot dari para pemain lainnya. Setiap pemain akan memiliki sebuah bot dimana tujuan dari bot ini adalah mengumpulkan diamond sebanyak-banyaknya. Cara mengumpulkan diamond tersebut tidak akan sesederhana itu, tentunya akan terdapat berbagai rintangan yang akan membuat permainan ini menjadi lebih seru dan kompleks. Untuk memenangkan pertandingan, setiap pemain harus mengimplementasikan strategi tertentu pada masing-masing bot-nya. Penjelasan lebih lanjut mengenai aturan permainan akan dijelaskan di bawah.

Pada tugas pertama Strategi Algoritma ini, mahasiswa diminta untuk membuat sebuah bot yang nantinya akan dipertandingkan satu sama lain. Tentunya mahasiswa harus menggunakan strategi greedy dalam membuat bot ini.

Program permainan Diamonds terdiri atas:

1. Game engine, yang secara umum berisi:
 - a. Kode backend permainan, yang berisi logic permainan secara keseluruhan serta API yang disediakan untuk berkomunikasi dengan frontend dan program bot
 - b. Kode frontend permainan, yang berfungsi untuk memvisualisasikan permainan
2. Bot starter pack, yang secara umum berisi:
 - a. Program untuk memanggil API yang tersedia pada backend
 - b. Program bot logic (bagian ini yang akan kalian implementasikan dengan algoritma greedy untuk bot kelompok kalian)
 - c. Program utama (main) dan utilitas lainnya.

Untuk mengimplementasikan algoritma pada bot tersebut, mahasiswa dapat menggunakan game engine dan membuat bot dari bot starter pack yang telah tersedia pada pranala berikut.

- **Game engine :**
<https://github.com/haziqam/tubes1-IF2211-game-engine/releases/tag/v1.1.0>
- **Bot starter pack :**
<https://github.com/haziqam/tubes1-IF2211-bot-starter-pack/releases/tag/v1.0.1>

Komponen-komponen dari permainan Diamonds antara lain:

1. Diamond



Gambar 1.2. Komponen Diamonds

Untuk memenangkan pertandingan, kita harus mengumpulkan diamond ini sebanyak-banyaknya dengan melewati/melangkahnya. Terdapat 2 jenis diamond yaitu diamond biru dan diamond merah. Diamond merah bernilai 2 poin, sedangkan yang biru bernilai 1 poin. Diamond akan di-regenerate secara berkala dan rasio antara diamond merah dan biru ini akan berubah setiap regeneration.

IF2211	Strategi Algoritma	Tugas Besar	4
--------	--------------------	-------------	---

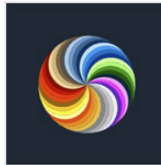
2. Red Button/Diamond Button



Gambar 1.3. Komponen Reset Button

Ketika red button ini dilewati/dilangkahi, semua diamond (termasuk red diamond) akan di-generate kembali pada board dengan posisi acak. Posisi red button ini juga akan berubah secara acak jika red button ini dilangkahi.

3. Teleporters



Gambar 1.3. Komponen Teleporters

Terdapat 2 teleporter yang saling terhubung satu sama lain. Jika bot melewati sebuah teleporter maka bot akan berpindah menuju posisi teleporter yang lain.

4. Bots and Bases







Gambar 1.4. Komponen Teleporters

Pada game ini kita akan menggerakkan bot untuk mendapatkan diamond sebanyak banyaknya. Semua bot memiliki sebuah Base dimana Base ini akan digunakan untuk menyimpan diamond yang sedang dibawa. Apabila diamond disimpan ke base, score bot akan bertambah senilai diamond yang dibawa dan inventory (akan dijelaskan di bawah) bot menjadi kosong.

IF2211	Strategi Algoritma	Tugas Besar	5
--------	--------------------	-------------	---

5. Inventory

Name	Diamonds	Score	Time
stima		0	43s
stima2		0	43s
stima1		0	44s
stima3		0	44s

Gambar 1.5. Inventory Bots

Bot memiliki inventory yang berfungsi sebagai tempat penyimpanan sementara diamond yang telah diambil. Inventory ini memiliki kapasitas maksimum sehingga sewaktu waktu bisa penuh. Agar inventory ini tidak penuh, bot bisa menyimpan isi inventory ke base agar inventory bisa kosong kembali.

Untuk mengetahui flow dari game ini, berikut ini adalah cara kerja permainan Diamonds.

1. Pertama, setiap pemain (bot) akan ditempatkan pada board secara random. Masing-masing bot akan mempunyai home base, serta memiliki score dan inventory awal bernilai nol.
2. Setiap bot diberikan waktu untuk bergerak, waktu yang diberikan semua sama untuk setiap pemain.
3. Objektif utama bot adalah mengambil diamond-diamond yang ada di peta sebanyak-banyaknya. Seperti yang sudah disebutkan di atas, diamond yang berwarna merah memiliki 2 poin dan diamond yang berwarna biru memiliki 1 poin.
4. Setiap bot juga memiliki sebuah inventory, dimana inventory berfungsi sebagai tempat penyimpanan sementara diamond yang telah diambil. Inventory ini sewaktu-waktu bisa penuh, maka dari itu bot harus segera kembali ke home base.
5. Apabila bot menuju ke posisi home base, score bot akan bertambah senilai diamond yang tersimpan pada inventory dan inventory bot akan menjadi kosong kembali.
6. Usahakan agar bot anda tidak bertemu dengan bot lawan. Jika bot A menempa posisi bot B, bot B akan dikirim ke home base dan semua diamond pada inventory bot B akan hilang, diambil masuk ke inventory bot A (istilahnya tackle).
7. Selain itu, terdapat beberapa fitur tambahan seperti teleporter dan red button yang dapat digunakan apabila anda menuju posisi objek tersebut.
8. Apabila waktu seluruh bot telah berakhir, maka permainan berakhir. Score masing-masing pemain akan ditampilkan pada tabel Final Score di sisi kanan layar.

Spesifikasi Tugas Besar 1

- Buatlah program sederhana dalam bahasa Python yang mengimplementasikan algoritma Greedy pada bot permainan Diamonds dengan tujuan memenangkan permainan.
- Tugas dikerjakan berkelompok dengan anggota minimal 2 orang dan maksimal 3 orang, boleh lintas kelas dan lintas kampus.
- Strategi greedy yang diimplementasikan setiap kelompok harus dikaitkan dengan fungsi objektif dari permainan ini, yaitu memenangkan permainan dengan memperoleh diamond sebanyak banyak nya dan jangan sampai diamond tersebut diambil oleh bot lain. Buatlah strategi greedy terbaik, karena setiap bot dari masing-masing kelompok akan diadu dalam kompetisi Tubes 1.
- Strategi greedy yang kelompok anda buat harus dijelaskan dan ditulis secara eksplisit pada laporan, karena akan diperiksa saat demo apakah strategi yang dituliskan sesuai dengan yang diimplementasikan. Tiap kelompok dapat menggunakan kreativitas yang bermacam macam dalam menyusun strategi greedy untuk memenangkan permainan. Implementasi pemain harus dapat dijalankan pada game engine yang telah disebutkan diatas serta dapat dikompetisikan dengan bot dari kelompok lain.
- Program harus mengandung komentar yang jelas, dan untuk setiap strategi greedy yang disebutkan, harus dilengkapi dengan kode sumber yang dibuat.
- Mahasiswa dilarang menggunakan kode program yang diunduh dari Internet. Mahasiswa harus membuat program sendiri, diperbolehkan untuk belajar dari program yang sudah ada.
- Mahasiswa dianggap sudah melihat dokumentasi dari game engine, sehingga tidak terjadi kesalahpahaman spesifikasi antara mahasiswa dan asisten.
- BONUS (maks 10): Membuat video tentang aplikasi greedy pada bot serta simulasinya pada game kemudian mengunggahnya di Youtube. Video dibuat harus memiliki audio dan menampilkan wajah dari setiap anggota kelompok. Untuk contoh video tubes stima tahun-tahun sebelumnya dapat dilihat di Youtube dengan kata kunci “Tubes Stima”, “strategi algoritma”, “Tugas besar stima”, dll.
- Jika terdapat kesulitan selama mengerjakan tugas besar sehingga memerlukan bimbingan, maka dapat melakukan asistensi tugas besar kepada asisten (opsional). Dengan catatan asistensi hanya bersifat membimbing, bukan memberikan “jawaban”.
- Terdapat juga demo dari program yang telah dibuat. Pengumuman tentang demo menunggu pemberitahuan lebih lanjut dari asisten.

IF2211	Strategi Algoritma	Tugas Besar	7
--------	--------------------	-------------	---

- Bot yang telah dibuat akan dikompetisikan dengan kelompok lain dan disaksikan oleh seluruh peserta kuliah. Terdapat hadiah menarik bagi kelompok yang memenangkan kompetisi.
- Setiap kelompok harap melaporkan nama kelompok dan anggotanya dan diserahkan kepada asisten untuk didata.
- Kelompok yang terindikasi melakukan kecurangan akan diberikan nilai 0 pada tugas besar.
- Program disimpan dalam repository yang bernama Tubes1_NamaKelompok dengan nama kelompok.

IF2211	Strategi Algoritma	Tugas Besar	8
--------	--------------------	-------------	---

BAB II

LANDASAN TEORI

2.1 Dasar Teori

Algoritma Greedy adalah metode yang dapat menyelesaikan masalah secara bertahap dan merupakan salah satu pendekatan yang dipakai dalam optimasi. Prinsip dasar dari algoritma Greedy adalah mengambil sebanyak mungkin keuntungan yang bisa didapat pada saat ini. Algoritma ini tergolong heuristik dan susunan langkah-langkahnya diatur dengan sistematis dalam rangka menyelesaikan masalah.

Algoritma greedy tidak selalu menghasilkan jawaban yang terbaik meskipun ia memilih opsi yang paling baik pada saat itu. Algoritma ini sering dipakai untuk memperoleh solusi perkiraan di mana kecepatan lebih diprioritaskan dibandingkan dengan ketepatan. Contohnya, untuk masalah Travelling Salesman Person, mencari solusi yang pasti menggunakan metode bruteforce memerlukan waktu yang cukup lama. Dengan memanfaatkan algoritma greedy, kita bisa mendapatkan solusi yang tidak selalu ideal tetapi bisa dianggap sebagai perkiraan dari solusi terbaik. Untuk memastikan bahwa algoritma greedy benar-benar optimal, perlu ada pembuktian secara matematis.

Algoritma ini memiliki sifat serakah, di mana setiap langkahnya memilih solusi terbaik yang tersedia saat ini berdasarkan karakteristik dari masalah tersebut. Fokus utama dari pendekatan ini adalah untuk mendapatkan solusi terbaik secara keseluruhan dari solusi terbaik yang tersedia saat ini. Kelebihan utama dari algoritma serakah adalah kesederhanaan, efektivitas, dan kemudahan dalam penerapan. Meskipun algoritma serakah memiliki kepraktisan dan efisiensi yang tinggi, ada juga beberapa keterbatasan dan kekurangan yang perlu diperhatikan.

2.2 Game Engine Diamonds

Game engine dari permainan Diamonds dapat ditemukan pada tautan <https://github.com/haziqam/tubes1-IF2211-game-engine/releases/tag/v1.1.0>. Game engine ini telah dipersiapkan sehingga kita hanya perlu melakukan konfigurasi untuk menggunakannya. Ada beberapa dependencies yang digunakan pada game engine ini seperti:

1. Node.js
2. Docker desktop
3. Yarn

IF2211	Strategi Algoritma	Tugas Besar	9
--------	--------------------	-------------	---

Game engine Diamonds ini tidak menggunakan engine compose sehingga kita memerlukan Docker untuk memproses dan menjalankan seluruh komponen dari game engine ini. Pada game engine ini, Docker juga membantu dalam proses berjalannya aplikasi secara independen dan konsisten di dalam mesin local.

2.3 Struktur Game Engine

Penting untuk mengetahui komponen-komponen dari game engine Diamonds karena akan sangat berguna bagi pengembangan bot dalam permainan Diamonds tersebut. Game engine berdasarkan tautan releases pada subbab 2.2. memiliki beberapa komponen yang strukturnya sendiri terdiri dari:

1. Folder *frontend* – berisi tampilan antarmuka pengguna untuk permainan Diamonds
2. Folder *backend* – berisi konfigurasi pengolahan data dari folder types dengan frontend seperti komponen-komponen dalam game yaitu Diamonds, Teleporter, Reset Button, Base, dan juga bots.
3. Folder *type* – penyimpanan game objects seperti base, bot, diamond, dummy-bot, teleport, teleport relocation, dan reset button.

2.4 Bot Engine Diamonds

Terdapat dua (2) file penting yang terdapat pada direktori utama file bot engine diamonds berdasarkan tautan releases pada subbab 2.2. File yang dibutuhkan penting untuk diketahui karena memberikan kita pemahaman tentang cara kerja bot engine diluar logika pemrograman yang kita rancang. File pendukung yang penting dalam bot engine tersebut adalah:

1. *main.py* – Membantu bot dalam permainan Diamonds untuk berkomunikasi dengan server melalui API. Bot ini menerima beberapa argumen baris perintah untuk mengkonfigurasi cara bermainnya.
2. *decode.py* – Menerima data dalam bentuk dictionary atau daftar dictionary. File ini mengonversi kunci-kunci dalam data tersebut menjadi snake case menggunakan `_keys_to_snake_case()` dan melakukan rekursi untuk mengonversi kunci-kunci dalam dictionary yang bersarang (nested) jika ada.

Pada bot engine ini terdapat juga folder game yang akan mendukung dan menopang keberjalanan logika program yang telah dirancang. Pada laporan ini akan dijelaskan file yang penting dalam folder game seperti util dan juga folder logic. Pada tugas besar ini, kita akan diminta untuk mengimplementasikan logika program algoritma greedy yang akan dirancang.

IF2211	Strategi Algoritma	Tugas Besar	10
--------	--------------------	-------------	----

2.5 Alur Permainan Diamonds

Terdapat beberapa prasyarat/dependencies yang harus diunduh dan/atau diinstalasi sebelum mengembangkan bot permainan diamonds dan juga menjalankan game engine Diamonds by Etime. Prasyarat/dependencies tersebut adalah:

1. Game Engine
 - Node.js (<https://nodejs.org/en>)
 - Docker desktop (<https://www.docker.com/products/docker-desktop/>)
 - Yarn
2. Bot Engine
 - Python

2.5.1 Menjalankan Game Engine Diamonds

Pada permainan diamonds, kita tidak memerlukan melakukan konfigurasi maupun perubahan apapun pada folder game engine. Sebelum menjalankan game engine, pastikan bahwa Node.js, Docker desktop, dan juga yarn telah terinstall pada perangkat anda. Setelah prasyarat untuk game engine telah terinstall silahkan mengikuti langkah-langkah berikut untuk menjalani game engine:

1. Download source code .zip pada tautan
<https://github.com/haziqam/tubes1-IF2211-game-engine/releases/tag/v1.1.0>
2. Extract zip tersebut, lalu masuk ke folder hasil extractnya dan buka terminal.

3. Masuk ke root directory dari project

```
cd tubes1-IF2110-game-engine-1.1.0
```

4. Instalasi dependencies

```
yarn
```

5. Konfigurasi environment variable dengan menjalankan script berikut

```
./scripts/copy-env.bat
```

6. Setup local database (buka aplikasi docker desktop terlebih dahulu, lalu jalankan command berikut di terminal)

```
docker compose up -d database
```

IF2211	Strategi Algoritma	Tugas Besar	11
--------	--------------------	-------------	----

7. Jalankan script berikut untuk windows

```
./scripts/setup-db-prisma.bat
```

8. Lakukan perintah build

```
npm run build
```

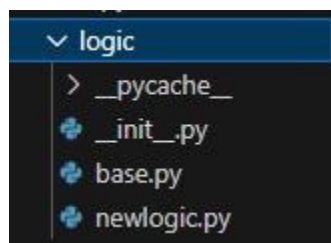
9. Lakukan perintah start

```
npm run start
```

2.5.2 Mengembangkan Bot Engine Permainan Diamonds

Sesuai dengan folder bot engine yang dapat diunduh melalui tautan yang terdapat pada Bab 1, pengembangan bot engine dapat dilakukan melalui file `./game/logic/random.py`. Kita juga dapat mengatur waktu setiap langkah bot berjalan pada file `./main.py` dengan mengubah value pada fungsi `Pustaka` yaitu `sleep(value)`.

Pada tugas pengembangan bot dengan algoritma greedy memiliki beberapa batasan seperti kita tidak diperbolehkan untuk mengubah struktur program maupun program selain file yang berisi program bot yang dikembangkan pada folder `./game/logic` oleh kelompok kami. Saat mengembangkan bot dalam folder `./game/logic` harus dipastikan bahwa terdapat fungsi `next move` dalam class yang kita rancang. Fungsi `next move` yang kita rancang juga harus menerima 2 parameter yaitu variabel dengan class `'GameObject'` dan juga `'Board'`. Fungsi `next move` yang telah dibuat juga harus mengembalikan atau return dua value yaitu `delta_x` dan `delta_y` yang memiliki variasi nilai yaitu -1, 0, dan 1.



Gambar 2.5.2.2 File yang terdapat pada program logic

Untuk menjalankan satu bot (pada contoh ini, kita menjalankan satu bot dengan logic yang terdapat pada file `game/logic/random.py`) kita dapat menjalankan perintah

IF2211	Strategi Algoritma	Tugas Besar	12
--------	--------------------	-------------	----

```
python main.py --logic Random --email=your_email@example.com --name=your_name  
--password=your_password --team etimo
```

dari root directory program bot engine. Kita dapat menjalankan beberapa bot sekaligus dengan cara menjalankan perintah

```
./run-bots.bat
```

untuk windows pada root directory program bot engine. Kita diperbolehkan untuk melakukan konfigurasi pada file ./run-bots.bat sesuai dengan nama logika yang telah dikembangkan dan juga nama bot yang kita inginkan pada program.

2.6 Cara Kerja Program

1) Inisialisasi Arah Pergerakan

(kanan, bawah, kiri, atas), yang masing-masing didefinisikan sebagai pasangan koordinat (x, y).

```
self.directions = [(1, 0), (0, 1), (-1, 0), (0, -1)]
```

2) Menemukan berlian terbaik yang dapat dikumpulkan oleh bot berdasarkan rasio nilai berlian terhadap jaraknya.

```
def _find_best_diamond(self, bot_pos: Position, diamonds: List[GameObject]) ->  
Optional[Position]:  
    if not diamonds:  
        return None  
  
    best_ratio = float('-inf')  
    best_position = None  
  
    for diamond in diamonds:  
        diamond_value = getattr(diamond.properties, "value", 1)  
        dist = abs(bot_pos.x - diamond.position.x) + abs(bot_pos.y -  
diamond.position.y)  
  
        if dist == 0:  
            # Jangan hitung berlian yang posisinya sama dengan bot  
            continue
```

```

        ratio = diamond_value / dist
        if ratio > best_ratio:
            best_ratio = ratio
            best_position = diamond.position

    return best_position

```

3) Menentukan target pergerakan

Jika bot telah mengumpulkan 4 berlian atau lebih, bot akan bergerak kembali ke base:

```

if props.diamonds >= 4:
    if (self.current_target is None or
        self.current_target.x != props.base.x or
        self.current_target.y != props.base.y):
        self.current_target = props.base

```

Jika bot belum cukup mengumpulkan berlian, bot akan mencari berlian terdekat yang sesuai dengan kapasitas inventory-nya:

```

diamond_objs = [obj for obj in board.game_objects if obj.type ==
"DiamondGameObject"]
    if diamond_objs:
        best_target = self._find_best_diamond(pos, diamond_objs)

```

4) Menghitung pergerakan bot menuju target berdasarkan koordinat,

Jika tidak ada target, bot memilih arah secara acak dengan probabilitas tertentu.

```

if self.current_target:
    move_x, move_y = get_direction(pos.x, pos.y,
self.current_target.x, self.current_target.y)
    else:
        # Kalau tidak ada target, pilih arah gerak secara acak dengan
probabilitas tertentu
        rand_val = random.random()
        if rand_val < 0.25:
            self.active_direction_index =
random.randrange(len(self.directions))
        elif rand_val < 0.85:

```

```

        # Kadang-kadang ubah arah secara sedikit bergantian
        if random.random() < 0.3:
            shift = random.choice([-1, 1])
            self.active_direction_index =
(self.active_direction_index + shift) % len(self.directions)

        chosen_direction =
self.directions[self.active_direction_index]
        move_x, move_y = chosen_direction

```

- 5) Mengganti indeks arah bot secara acak dengan probabilitas tertentu.

```

if random.random() > 0.7:
    self.active_direction_index = (self.active_direction_index
+ 1) % len(self.directions)

```

IF2211	Strategi Algoritma	Tugas Besar	15
--------	--------------------	-------------	----

BAB III

APLIKASI STRATEGI *GREEDY*

3.1 Proses Mapping Persoalan Diamonds ke Elemen-Elemen Algoritma Greedy

Permainan Diamonds merupakan permasalahan optimasi yang cocok diterapkan dengan algoritma greedy karena memiliki karakteristik pilihan lokal yang dapat mengarah pada solusi global yang baik. Berikut adalah elemen-elemen greedy:

- Himpunan kandidat

Himpunan kandidat terdiri dari semua target berlian yang tersedia dan belum diambil pada saat itu. Artinya, setiap berlian di area permainan yang bisa dicapai oleh bot merupakan elemen dalam himpunan kandidat. Bot mempertimbangkan semua berlian yang terlihat di peta sebagai pilihan selanjutnya.

- Himpunan Solusi

Solusi adalah hasil akhir yang ingin dicapai, yaitu urutan langkah atau rute yang ditempuh bot sehingga mengumpulkan berlian dan mengantarkannya ke base. Dalam implementasi greedy, solusi terbentuk dengan menambahkan satu per satu target berlian ke rute perjalanan bot hingga memenuhi kriteria penghentian. Solusi sementara yang dibangun meliputi riwayat titik-titik yang dikunjungi atau nilai total berlian yang terkumpul.

- Fungsi Solusi

Fungsi solusi mengambil hasil pilihan-pilihan greedy dan menghasilkan keluaran akhir dalam format yang diinginkan. Fungsi ini bisa berupa penghitungan total nilai berlian yang diangkut atau penyusunan urutan lintasan akhir bot.

- Fungsi Seleksi

Fungsi seleksi adalah kriteria untuk memilih kandidat terbaik berikutnya. Pada bot ini, kriteria tersebut adalah perhitungan rasio *value/distance* untuk setiap berlian: nilai berlian dibagi jarak tempuh menuju berlian tersebut. Kemudian berlian dengan rasio terbesar dipilih sebagai target selanjutnya. Prinsip ini mirip dengan metode greedy pada masalah knapsack fraksional, di mana item dengan rasio profit/berat tertinggi dipilih terlebih dahulu.

IF2211	Strategi Algoritma	Tugas Besar	16
--------	--------------------	-------------	----

- Fungsi Kelayakan

Fungsi kelayakan memeriksa apakah kandidat terpilih dapat dimasukkan ke dalam solusi tanpa melanggar batasan. Dalam implementasi bot, fungsi ini memastikan kapasitas inventori bot belum penuh sebelum memilih berlian baru. Misalnya, jika inventori telah mencapai batas maksimum (misalnya ≥ 4), bot tidak akan memilih berlian baru dan sebaliknya akan kembali ke base.

- Fungsi Objektif

Fungsi objektif mengevaluasi kualitas solusi akhir. Tujuan akhir bot adalah memaksimalkan skor total yang diperoleh dari nilai berlian yang diambil dan dikembalikan ke base. Oleh karena itu, fungsi objektif dapat berupa jumlah keseluruhan nilai berlian yang terkumpul atau skor akhir permainan. Strategi greedy berusaha meningkatkan nilai objektif ini dengan memilih berlian bernilai tinggi dan jarak dekat secara lokal.

3.2 Eksplorasi Alternatif Solusi Greedy

Dalam pengembangan strategi greedy untuk bot pengumpul diamond, terdapat beberapa pendekatan yang dapat dipertimbangkan untuk meningkatkan performa bot dalam pengumpulan diamonds seperti:

1. Greedy by Value per Distance

Strategi utama yang diterapkan bot adalah memilih target berlian berdasarkan rasio nilai terhadap jarak (*value/distance*). Setiap berlian yang tersedia di papan akan dihitung rasionya, kemudian bot memilih target dengan rasio tertinggi. Strategi ini memungkinkan bot secara efisien mengambil berlian yang paling “menguntungkan” secara lokal – yakni, bernilai tinggi dan mudah dijangkau. Pendekatan ini sangat serupa dengan strategi greedy pada persoalan fractional knapsack, di mana item dengan rasio profit terhadap berat tertinggi diprioritaskan terlebih dahulu.

2. Greedy by Inventory Capacity

Strategi ini memastikan bahwa bot tidak terus mengambil berlian jika kapasitas inventory telah penuh. Dalam program, jika bot sudah membawa minimal 4 berlian, maka target langsung dialihkan ke base, tanpa mempertimbangkan keberadaan berlian lain. Hal ini merupakan implementasi dari fungsi kelayakan (*feasibility*) untuk mencegah kelebihan muatan, serta mengamankan poin sebelum terjadi risiko (misalnya tackle oleh bot musuh).

IF2211	Strategi Algoritma	Tugas Besar	17
--------	--------------------	-------------	----

3. Greedy by Exploration

Jika tidak ditemukan berlian di sekitar, atau semua target sudah diambil oleh pemain lain, maka bot mengaktifkan strategi eksplorasi secara acak namun terarah. Bot memilih arah secara dinamis berdasarkan pola rotasi dan probabilitas, untuk terus menjelajahi area baru. Meskipun bukan bagian dari pemilihan kandidat nilai terbaik, strategi ini menjaga agar bot tetap aktif dan tidak stagnan saat peluang greedy tidak tersedia. Eksplorasi ini menjadi bagian penting dalam menjaga performa jangka panjang dan menemukan peluang baru.

3.3 Analisis Efisiensi dan Efektivitas Solusi Greedy

Dalam menganalisis berbagai pendekatan greedy untuk bot pengumpul diamond, kita dapat melihat trade-off yang menarik antara efisiensi komputasi dan efektivitas strategi seperti:

1. Greedy by Value per Distance

Efisiensi:

- Kompleksitas tetap $O(n)$ untuk mengevaluasi semua diamond
- Perhitungan rasio value/distance sederhana dan cepat
- Optimal untuk papan dengan distribusi diamond acak

Efektivitas:

- Maksimalisasi poin per langkah
- Sangat cocok untuk early game saat inventory masih kosong

2. Greedy by Inventory Capacity

Efisiensi:

- Pemeriksaan kondisi sangat ringan ($O(1)$)
- Tidak membutuhkan perhitungan tambahan

Efektivitas:

- Mencegah kehilangan skor karena tackle
- Menjamin stabilitas pengumpulan nilai

3. Greedy by Exploration

Efisiensi:

- Kompleksitas $O(1)$ untuk pemilihan arah acak
- Minim resource, cukup rotasi dan probabilitas

Efektivitas:

- Membantu bot tetap aktif saat tidak ada target
- Potensial menemukan diamond tersembunyi

IF2211	Strategi Algoritma	Tugas Besar	18
--------	--------------------	-------------	----

3.4 Strategi Greedy yang Dipilih

Strategi utama yang diterapkan dalam bot ini mengadopsi pendekatan greedy berbasis jarak terdekat dengan mempertimbangkan keamanan terhadap bot lawan dan efisiensi penggunaan teleportasi. Bot secara konsisten memilih diamond terdekat yang sesuai dengan kapasitas inventory saat ini, dengan mekanisme fallback ke penghindaran musuh ketika terdeteksi bahaya di sekitar.

Implementasi strategi ini menggabungkan tiga lapisan utama:

1. Prioritas keamanan, yaitu pengecekan keberadaan musuh dalam radius 2 unit. Jika musuh berada terlalu dekat saat inventory penuh, maka bot menghindar dan menghindari konfrontasi.
2. Optimasi rute dengan teleport, yaitu pemanfaatan teleportasi hanya jika secara perhitungan jarak dan tujuan memberikan keuntungan signifikan, seperti lebih cepat menuju diamond atau base.
3. Fallback ke greedy konvensional, yaitu pemilihan diamond terdekat berdasarkan rasio value/distance , jika tidak ada ancaman atau opsi teleport yang menguntungkan.

Dari sisi efektivitas, strategi ini mengutamakan:

1. Manajemen risiko dengan melakukan penghindaran musuh secara dinamis, terutama saat inventory penuh dan rawan tackle.
2. Optimisasi gerakan melalui analisis biaya-manfaat dari penggunaan teleport, dengan mempertimbangkan jarak ke teleport, jarak teleport ke target (diamond atau base), dan kondisi papan.
3. Pengambilan keputusan kontekstual, yaitu strategi berubah tergantung kondisi inventory. Bot mengambil risiko lebih tinggi saat inventory kosong (agresif mengambil diamond), dan menjadi lebih defensif saat inventory penuh (prioritaskan pulang aman).

IF2211	Strategi Algoritma	Tugas Besar	19
--------	--------------------	-------------	----

BAB IV

IMPLEMENTASI DAN PENGUJIAN

4.1 Implementasi Algoritma Greedy pada Program Bot

Berikut adalah implementasi program utama kami dalam bentuk *pseudocode*.

```
CLASS GreedyDiamondLogic EXTENDS BaseLogic

METHOD __init__():
    SET directions TO [(1, 0), (0, 1), (-1, 0), (0, -1)] // kanan, bawah, kiri, atas
    SET current_target TO None
    SET active_direction_index TO 0

METHOD _find_best_diamond(bot_pos, diamonds) RETURNS Position OR None:
    IF diamonds IS EMPTY:
        RETURN None
    SET best_ratio TO -∞
    SET best_position TO None
    FOR EACH diamond IN diamonds:
        SET value TO diamond.value
        SET dist TO |bot_pos.x - diamond.x| + |bot_pos.y - diamond.y|
        IF dist == 0:
            CONTINUE
        SET ratio TO value / dist
        IF ratio > best_ratio:
            best_ratio = ratio
            best_position = diamond.position
    RETURN best_position

METHOD next_move(bot, board) RETURNS (delta_x, delta_y):
    SET props TO bot.properties
    SET pos TO bot.position

    IF props.diamonds >= 4:
        // Bot kembali ke base
        IF current_target IS None OR current_target ≠ props.base:
            SET current_target TO props.base

    ELSE:
        SET diamond_objs TO semua objek "DiamondGameObject" di board
        IF diamond_objs IS NOT EMPTY:
            SET best_target TO CALL _find_best_diamond(pos, diamond_objs)
            IF best_target IS NOT None AND best_target ≠ current_target:
```

```

    SET current_target TO best_target
    ELSE IF best_target IS None:
        SET current_target TO None
    ELSE:
        SET current_target TO None

// Reset target jika sudah sampai
IF current_target IS NOT None AND pos == current_target:
    SET current_target TO None

// Tentukan arah gerak
IF current_target IS NOT None:
    (delta_x, delta_y) = arah dari pos ke current_target
ELSE:
    // Eksplorasi acak jika tidak ada target
    SET rand_val TO random(0,1)
    IF rand_val < 0.25:
        SET active_direction_index TO random index dari directions
    ELSE IF rand_val < 0.85 AND random(0,1) < 0.3:
        SET shift TO random -1 atau 1
        UPDATE active_direction_index (modulo panjang directions)
    SET (delta_x, delta_y) TO directions[active_direction_index]
    IF random(0,1) > 0.7:
        ROTASI arah aktif (active_direction_index +1 mod 4)

RETURN (delta_x, delta_y)

```

4.2 Penjelasan Struktur Data pada Program

Struktur data yang digunakan dalam program ini mengandalkan dua objek utama yang sangat penting dalam membangun algoritma secara langsung. Objek-objek tersebut adalah sebagai berikut:

1. Objek GameObject

GameObject merupakan representasi dari setiap elemen dalam permainan, baik itu diamond, bot pemain, maupun teleportasi. Masing-masing objek memiliki sejumlah properti penting seperti type (jenis objek), id (identitas unik), position (koordinat x dan y), serta properties yang berisi atribut tambahan. Untuk objek bot, properti tersebut antara lain adalah jumlah diamond yang sedang dibawa (diamonds), kapasitas maksimum (inventory_size), serta points untuk nilai total diamond yang dibawa. Objek diamond memiliki nilai tertentu, sedangkan objek teleport tidak memiliki nilai, tetapi dapat digunakan untuk berpindah lokasi.

IF2211	Strategi Algoritma	Tugas Besar	21
--------	--------------------	-------------	----

2. Objek Board

Board berfungsi sebagai representasi peta permainan secara keseluruhan. Objek ini menyimpan semua elemen dalam permainan ke dalam daftar `game_objects`. Dalam prosesnya, objek-objek ini dikelompokkan kembali oleh program ke dalam beberapa kategori seperti `diamond`, `bot`, dan `teleport` untuk memudahkan proses evaluasi dan pengambilan keputusan. Melalui objek `Board`, `bot` dapat mengakses semua objek yang relevan dan melakukan penghitungan jarak, deteksi keberadaan musuh, serta perhitungan strategi optimal.

4.3 Analisis Desain Solusi Algoritma

Dalam program ini, digunakan dua struktur data utama sebagai fondasi pengambilan keputusan `bot`, yaitu `GameObject` dan `Board`.

`GameObject` berfungsi untuk merepresentasikan semua elemen yang ada dalam game. Objek-objek ini memiliki berbagai tipe: mulai dari `diamond` yang bisa dikumpulkan (`DiamondGameObject`), `bot` pemain (`BotGameObject`), hingga portal teleportasi (`TeleportGameObject`). Setiap objek memiliki informasi penting seperti:

- `type` untuk mengetahui jenis objek,
- `id` sebagai identitas unik,
- `position` yang menyimpan koordinat `x` dan `y` di papan permainan,
- serta atribut tambahan seperti jumlah `diamond` yang dibawa, kapasitas `inventory`, dan nilai poin dari `diamond`.

Sementara itu, `Board` merupakan representasi peta permainan secara menyeluruh. Di dalamnya terdapat daftar semua `GameObject`, dan program akan memproses daftar ini untuk mengelompokkan objek ke dalam kategori seperti `diamond`, `bot`, atau `teleport`. Pengelompokan ini sangat membantu saat `bot` harus mengambil keputusan secara cepat dan tepat.

Dari dua struktur ini, dibentuk logika permainan yang kompleks namun efisien. `Bot` akan menghitung jarak ke setiap `diamond`, mempertimbangkan ketersediaan `teleport`, dan mengecek kondisi `inventory` sebelum menentukan langkah. Jika `inventory` penuh, `bot` langsung diarahkan ke base. Namun jika ada musuh dalam jarak berbahaya, maka `bot` memprioritaskan penghindaran.

IF2211	Strategi Algoritma	Tugas Besar	22
--------	--------------------	-------------	----

Sistem juga membandingkan jalur biasa dengan jalur teleport, lalu memilih mana yang lebih cepat. Pendekatan ini membuat pergerakan bot menjadi adaptif dan hemat waktu.

Secara keseluruhan, penggunaan struktur data yang terorganisir dan logika kondisi yang fleksibel memungkinkan bot beradaptasi secara dinamis terhadap perubahan situasi di papan permainan. Walaupun begitu, masih terbuka peluang untuk peningkatan lebih lanjut, seperti penambahan algoritma pathfinding yang lebih cerdas.

IF2211	Strategi Algoritma	Tugas Besar	23
--------	--------------------	-------------	----

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Dalam Tugas Besar ini, telah diciptakan sebuah bot permainan yang memanfaatkan strategi algoritma greedy, mampu membuat keputusan secara efisien dan responsif dalam lingkungan permainan yang berubah-ubah. Bot ini menggunakan pendekatan untuk memilih opsi terbaik dengan mempertimbangkan perbandingan antara nilai diamond dan jarak yang ditempuh, serta otomatis merencanakan kembali ke basis saat inventaris penuh.

Implementasi dari algoritma greedy dilaksanakan dengan cara yang teratur melalui struktur kelas GreedyDiamondLogic, yang menggabungkan pencarian target, penilaian keadaan permainan, dan pengambilan keputusan setiap langkah. Bantuan dari struktur data seperti GameObject dan Board memberikan kemudahan pada bot untuk mengakses informasi penting, sehingga dapat mengoptimalkan pergerakan dan mengumpulkan poin dengan lebih efektif.

Analisis menunjukkan bahwa strategi greedy yang diterapkan cukup efektif dalam hal waktu dan mudah untuk diimplementasikan. Meskipun tidak selalu menjamin hasil optimal secara global, metode ini mampu memberikan hasil yang bersaing, terutama dalam permainan real-time yang membutuhkan kecepatan dan ketepatan dalam membuat keputusan.

Dengan desain yang terorganisir, fleksibel, dan responsif, bot ini telah membuktikan bahwa strategi greedy tetap menjadi salah satu metode yang efektif dalam menangani masalah pencarian solusi optimal lokal di dalam ruang permainan yang rumit.

5.2 Saran

Pada panduan penggunaan game engine diamonds kami memiliki saran untuk menambahkan kesalahan – kesalahan umum yang mungkin terjadi saat melakukan konfigurasi. Hal ini dilakukan untuk meminimalisir terjadinya kesalahan pada saat melakukan instalasi terutama saat melakukan konfigurasi docker.

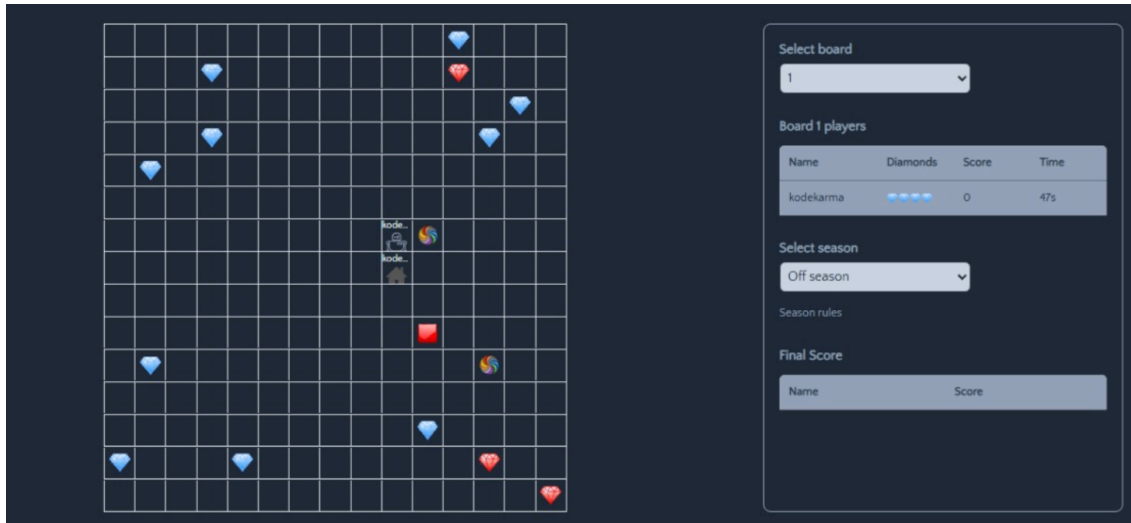
IF2211	Strategi Algoritma	Tugas Besar	24
--------	--------------------	-------------	----

LAMPIRAN

Repository Github

(<https://github.com/03-114-AdityaHotMartuaSihite/KodeKarmaBot.git>)

Bot Kelompok kode karma:



DAFTAR PUSTAKA

Wibowo, A. C., & Wowor, A. D. (2023). Perancangan Distribusi Hasil Produk Textil dengan Rute Terdekat dengan Algoritma Greedy. *J-SAKTI (Jurnal Sains Komputer dan Informatika)*, 7(1), 287-298.

Wang, Y. (2023). Review on greedy algorithm. *Theoretical and Natural Science*, 14, 233-239.

IF2211	Strategi Algoritma	Tugas Besar	26
--------	--------------------	-------------	----