

TUGAS BESAR I STRATEGI ALGORITMA
ALGORITMA GREEDY UNTUK PROGRAM DIAMONDS

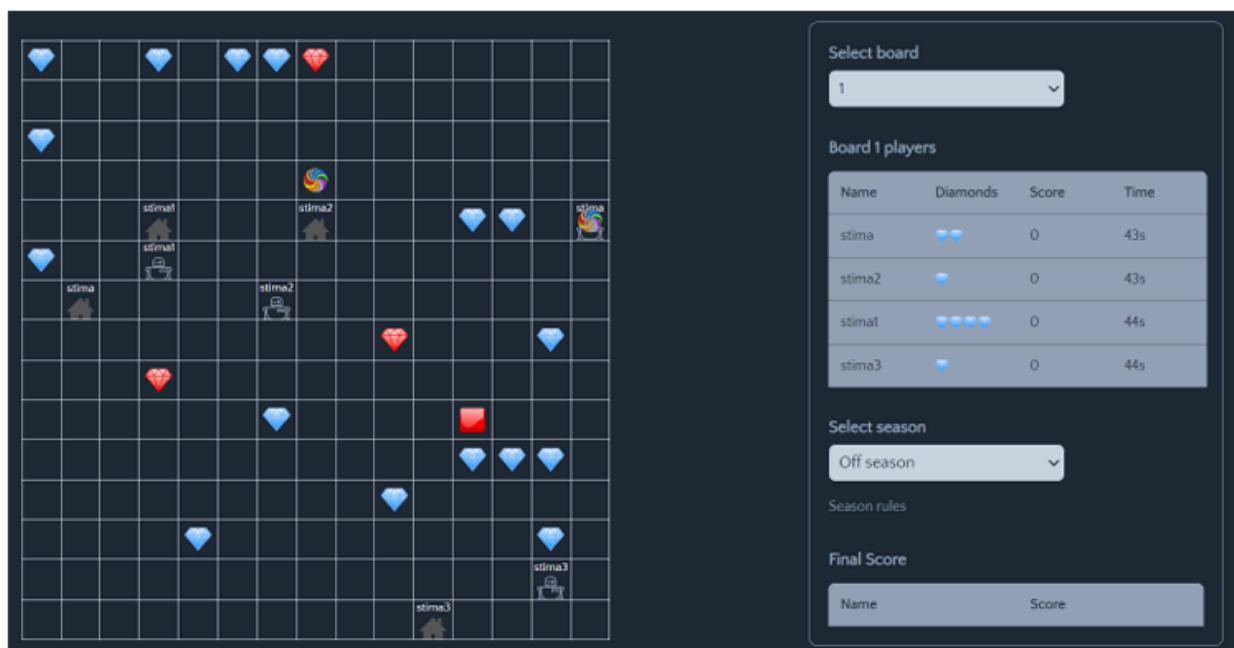


Kelompok 56 :

1. Vincent Hasiholan - 13518108
2. Devinzen - 13522064

1. BAB I : DESKRIPSI TUGAS

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.



Gambar I. Game *Diamonds*

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 :

- a. *Game Engine*, yang secara umum berisi :

- i. Kode *Backend* permainan, yang berisi logic permainan secara keseluruhan serta API yang disediakan untuk berkomunikasi dengan frontend dan program bot
 - ii. Kode *Frontend* permainan, yang berfungsi untuk memvisualisasikan permainan
- b. *Bot Starter Pack*, yang secara umum berisi :
- i. Program untuk memanggil API yang tersedia pada *Backend*
 - ii. Program bot logic (bagian ini yang akan kalian implementasikan dengan algoritma greedy untuk bot kelompok kalian)
 - iii. Program utama (main) dan utilitas lainnya

Komponen-komponen dari permainan Diamonds antara lain :

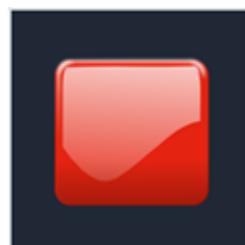
1. Diamonds



Gambar II. Blue Diamond - Red Diamond di dalam program Diamonds

Untuk memenangkan pertandingan, kita harus mengumpulkan diamond ini sebanyak-banyaknya dengan melewati/melangkahinya. 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.

2. Red Button/Diamond Button



Gambar III. Button dalam program Diamonds

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 IV. Teleporter di dalam program Diamonds

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 V. Bots dan Bases dalam program Diamonds

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

5. Inventory

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

Gambar VI. Inventory dalam program Diamonds

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 menimpa 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.

2. BAB II : LANDASAN TEORI

Algoritma Greedy adalah algoritma apa pun yang mengikuti metode heuristik dalam pemecahan masalah untuk membuat pilihan optimal secara lokal di setiap tahap. Dalam banyak permasalahan, strategi greedy tidak menghasilkan solusi optimal, tetapi suatu heuristik greedy dapat menghasilkan solusi optimal lokal yang mendekati solusi optimal global dalam jangka waktu yang wajar. Penyelesaian masalah dengan menggunakan algoritma ini berfokus pada menghasilkan hasil yang maksimal atau minimal. Beberapa contoh permasalahan yang dapat diselesaikan dengan algoritma ini adalah pertukaran uang, pemilihan jadwal dengan waktu seminimal mungkin, dan pemaksimalan banyaknya kegiatan yang dapat dilakukan.

3. BAB III : APLIKASI STRATEGI GREEDY

- a. Fokus ke pengambilan diamond
 - i. Dari koordinat robot berada, robot menganalisa daerah disekitarnya dalam jarak ≤ 5
 - ii. Robot memilih diamond yang nilainya paling tinggi dan jarak paling pendek
 - iii. Robot bergerak ke diamond tersebut
 - iv. Ketika sudah penuh, maka robot akan pergi ke markasnya untuk mengosongkan dari diamond yang sudah dibawa
 - v. Ulangi tahap i - iv hingga waktu permainan sudah habis.

- b. Fokus ke radius pengambilan diamond
 - i. Dari koordinat robot berada, robot menganalisa daerah disekitarnya
 - ii. Robot memilih tempat yang menghasilkan diamond terbanyak dengan memberikan jumlah langkah minimal. Robot perlu memikirkan langkah yang dapat dihasilkan dengan menggunakan teleporter.
 - iii. Ketika sudah tidak ada diamond di dalam jumlah langkah batasan, robot kembali ke markas.
 - iv. Ulangi tahap i - iii hingga waktu permainan sudah habis.
- c. Fokus ke pemain yang memiliki diamond terbanyak
 - i. Dari koordinat robot berada, robot menganalisa daerah disekitarnya
 - ii. Robot menganalisa robot lain yang memiliki diamond terbanyak
 - iii. Tahap i - ii diulang hingga ditemukan robot lain yang memiliki separah - parahnya empat diamond
 - iv. Robot membaca pergerakan dari robot yang ditarget dan mengintersepsi gerakan dari robot tersebut
 - v. Ketika berhasil dimakan robot tersebut, maka robot milik sendiri perkaya markasnya untuk menyimpan diamond di markas.
 - vi. Ulangi tahap i - v hingga waktu permainan selesai

4. BAB IV : IMPLEMENTASI DAN PENGUJIAN

No	Screenshot Kode	Penjelasan
1	<pre>def calc_distance(self, Position1: Position, Position2: Position) -> int: # no teleporters x, y = Position1.x - Position2.x, Position1.y - Position2.y if (x < 0): x *= -1 if (y < 0): y *= -1 return (x + y)</pre>	Fungsi calc_distance menganalisis jarak bot dengan jarak sesuatu objek. (Keterangan : tidak ada teleporter)
2	<pre>def next_move(self, board_bot: GameObject, board: Board): props = board_bot.properties current_position = board_bot.position base = board_bot.properties.base wak = board_bot.properties.wak if ((props.diamonds == 5) or ((1000 * (self.calc_distance(current_position, base) + 3)) > board_bot.properties.milliseconds_left)): # If we're to base self.goal_position = base else: self.goal_position = None diamond_list = board.diamonds # cari diamonds di jarak <= 5 diamond_list_focus = [d for d in diamond_list if (self.calc_distance(current_position, d.position) <= 5)] # kalau tidak ada radiusnya dibesarkan if (len(diamond_list_focus) == 0): diamond_list_focus = [d for d in diamond_list if (self.calc_distance(current_position, d.position) <= 10)] # Target diamond yang paling dekat sejauh dia dapat mengunjungi (teleport) target = None for i in range(len(diamond_list_focus)): if i == 0: target = diamond_list_focus[0] else: target = diamond_list_focus[i] if (target.properties.points <= props.points): target = diamond_list_focus[i] elif ((self.calc_distance(current_position, diamond_list_focus[i].position) < self.calc_distance(current_position, target.position)) and (diamond_list_focus[i].properties.points == target.properties.points)): target = diamond_list_focus[i] if target: self.goal_position = target.position if self.goal_position: # We are aiming for a specific position, calculate delta delta_x, delta_y = get_direction(current_position.x, current_position.y, self.goal_position.x, self.goal_position.y,) if ((delta_x == 0) and (delta_y == 0)): self.goal_position = None if not self.goal_position: # kode dari yang bot starter pack random lollll # kadang masih bisa ke sini kalau waktunya hampir habis (jalan-jalan disekitar base) # Roam around delta = self.directions[self.current_direction] delta_x = delta[0] delta_y = delta[1] if random.random() > 0.6: self.current_direction = (self.current_direction + 1) % len(self.directions) return delta_x, delta_y</pre>	Fungsi next_move adalah fungsi yang menentukan pergerakan dari bot kelompok kami (hasil modifikasi dari fungsi next_move di bot random). Hal pertama yang dilakukan adalah pengecekan apakah bot sudah membawa lima diamonds atau melihat jarak dari bot ke base ketika waktu sudah sedikit. Ketika salah satu syarat dari hal pertama terpenuhi, maka bot pergi ke base. Ketika persyaratan pada hal pertama tidak terpenuhi, maka bot melakukan pencarian terhadap diamond terbanyak. Ketika didapatkan posisi diamond terbanyak dengan batasan area yang dicari adalah lima dari posisi diamond yang ditarget, bot mencatat posisi tersebut dan menghitung jarak dari bot ke posisi tersebut. Bila ditemukan beberapa diamond dengan total poin yang sama, maka dibandingkan diamond mana yang memiliki jarak terdekat dari bot.

Untuk foto pengujian, foto ditaruh pada bagian lampiran.

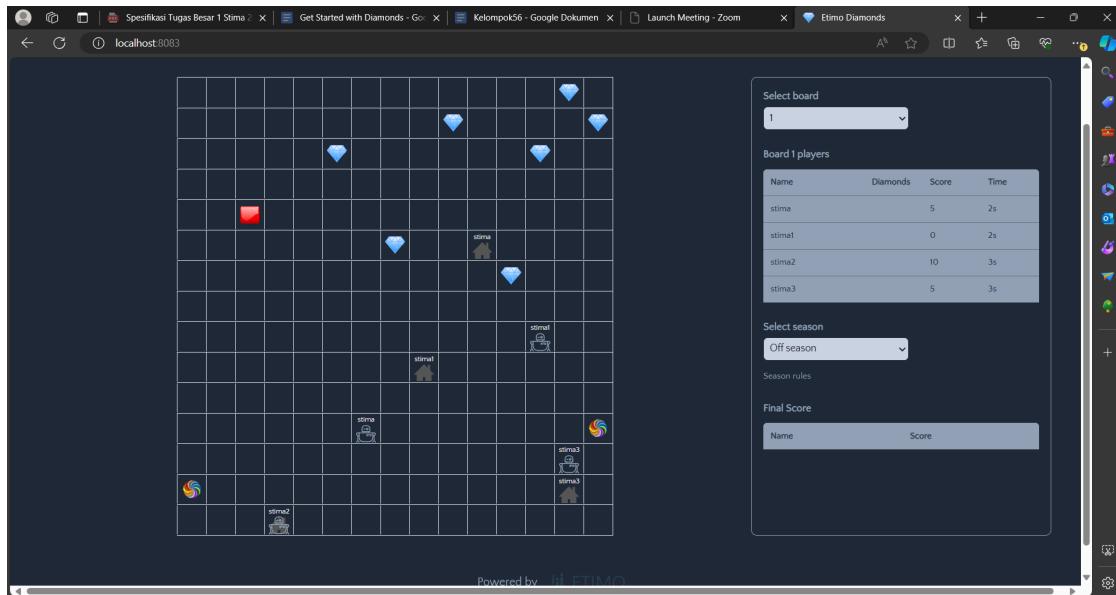
5. BAB V : KESIMPULAN DAN SARAN

Kesimpulan yang dapat kami sampaikan adalah dalam penggunaan algoritma greedy pada program Diamonds, bot yang dapat menganalisa tempat diamond yang memiliki poin terbanyak memiliki kemungkinan untuk menang yang tinggi. Perlu diperhatikan juga untuk waktu apakah masih ada waktu yang tersedia supaya bot dapat memasukkan diamond yang tersisa di dalam bawanya.

Untuk saran, dalam penggerjaan tugas besar ini, diusahakan salah satu komputer/laptop dari anggota dapat digunakan untuk membangun bot yang digunakan. Selain itu, dalam pembangunan bot dalam suatu permainan, disarankan untuk menganalisa area di sekitar bot untuk mengetahui keadaan di sekelilingnya.

6. LAMPIRAN

Repository github : https://github.com/Devinzenzhang/Tubes1_Kelompok56



stima dan stima 1 bot random, stima2 dan stima3 bot greedy

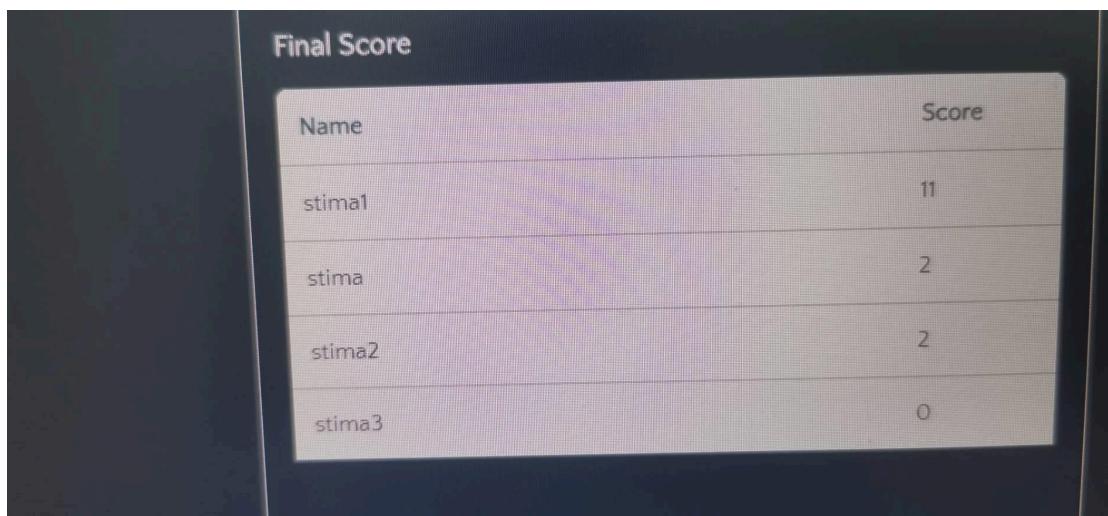


Foto bot stima3 ketika mencoba membuat kode dimana mengincar pemain dengan diamond terbanyak (sisanya bot random)

7. DAFTAR PUSTAKA

1. https://id.wikipedia.org/wiki/Algoritma_greedy
2. [https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2017-2018/Algoritma-Greedy-\(2018\).pdf](https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2017-2018/Algoritma-Greedy-(2018).pdf)