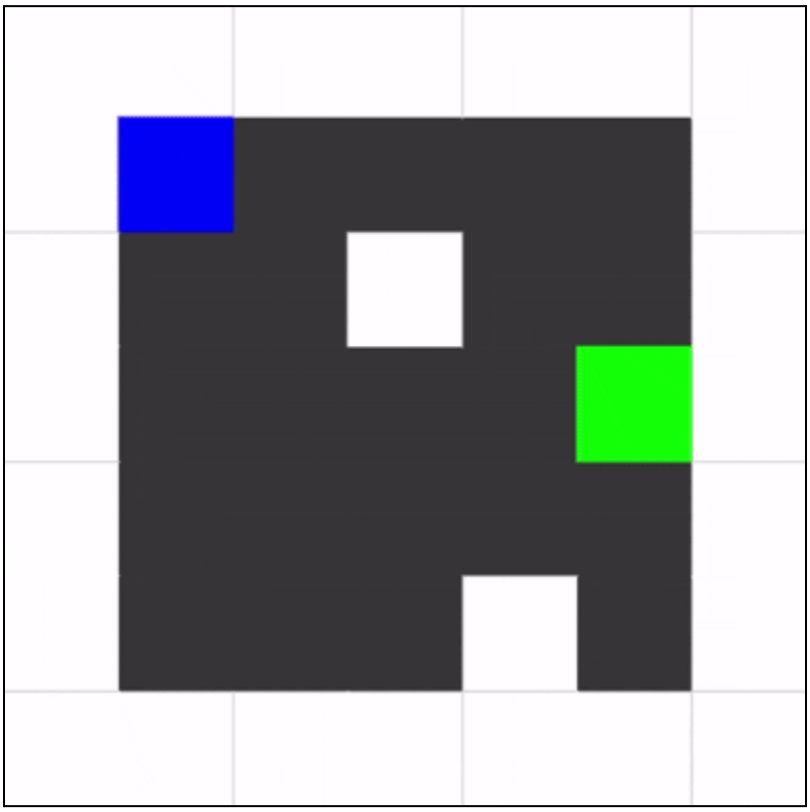
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung

Tugas Kecil 3 IF2211 Strategi Algoritma
Semester II tahun 2025/2026
Ice Sliding Puzzle Solver

Batas pengumpulan : Jumat, 8 Mei 2026, 23.59
Arsip pengumpulan :

- *Source* program yang dapat dijalankan beserta README
- Laporan (*soft copy*)

Deskripsi Tugas



Visualisasi Permainan Ice Sliding Puzzle Solver

Ice Sliding Puzzle adalah permainan logika di mana pemain harus menggerakkan karakter dari titik awal menuju titik keluar di atas permukaan es yang licin. Pin (dalam visualisasi warna biru) hanya dapat bergerak secara horizontal atau vertikal, namun karena permukaan licin, karakter tidak akan berhenti bergerak sampai menabrak dinding atau rintangan (dalam visualisasi warna putih).

Spesifikasi Wajib

- Buatlah sebuah program tersebut dalam bahasa Java, Python, C++, C, atau GO yang mengimplementasikan **algoritma *Pathfinding (UCS, GBFS, dan A*)*** untuk mencari solusi dalam permainan *Ice Sliding Puzzle Solver*
- Tugas dikerjakan **berkelompok** dengan anggota maksimal **2 orang,** boleh lintas kelas maupun kampus. Silahkan isi pendataan kelompok pada link berikut
- **Input:** program akan memberikan pengguna sebuah arahan pada terminal (jika tidak mengerjakan bonus) untuk memilih file test case berekstensi .txt, kemudian program membaca file test case tersebut yang berisi konfigurasi awal dari *map* Permainan yang akan diselesaikan. Perhatikan bahwa input dapat memiliki ukuran papan yang berbeda-beda (tidak harus persegi), serta program harus dapat memvalidasi apakah input yang diberikan merupakan input valid atau bukan

  Berikut adalah contoh file .txt yang akan dijadikan sebagai input.
  Baris pertama memuat 2 angka (N, M) yang menyatakan panjang dan lebar papan
  N baris berikutnya merepresentasikan papan
  N baris berikutnya merepresentasikan cost untuk melewati tile tersebut

```
7 7
XXXXXXX
X0****X
X**X**X
X****OX
X***1LX
XZ**X*X
XXXXXXX
999 999 999 999 999 999 999
999 3   5   2   8   1   999
999 7   4   999 6   9   999
999 2   8   3   5   4   999
999 6   1   7   2   999 999
999 9   3   4   999 8   999
999 999 999 999 999 999 999
```

Dengan keterangan:

* = path yang bisa dilewati

X = Rintangan/Batu. Aktor akan berhenti tepat sebelum batu.

L = Lava. Melewati lava akan mengalami *game over* (Meskipun tidak berhenti tepat di lava).

Z = Aktor/Pengguna

O = Titik tujuan

<i> = Angka. Ini adalah modifikasi yang ada pada spesifikasi tugas kecil berikut. Aktor harus melewati angka ini sesuai dengan urutan angka di papan. Petak dengan angka 1 harus dilewati setelah aktor melewati petak dengan angka 0, Petak dengan angka 2 harus dilewati setelah aktor melewati petak dengan angka 1, dan seterusnya. (Maksimal hanya sampai angka 9). Apabila aturan ini dilanggar,

misalnya petak 1 diinjak sebelum petak 0, maka akan game-over. Setelah petak ini dilewati, maka petak ini akan menjadi petak biasa, dengan kata lain, petak tersebut dapat dilewati lagi tanpa memikirkan constraint ini. Tile ini dianggap sebagai tile normal (bukan batu, artinya, Aktor akan menembus tile ini jika bergerak melaluinya)

Perlu diketahui bahwa, apabila aktor bergerak menuji sisi kanan/atas/kiri/bawah papan, tetapi tidak terdapat rintangan/batu pada sisi tersebut, maka akan mengalami *game over* dan papan ulang dari awal. Pin harus tepat berhenti di titik tujuan dan semua angka telah dilalui sesuai urutan untuk dikatakan sukses.

Setiap tile memiliki cost traversal yang diberikan pada bagian kedua input. Cost ini merepresentasikan biaya yang dikenakan ketika aktor melewati suatu tile. Cost movement dihitung berdasarkan jumlah cost seluruh tile yang dilalui selama satu gerakan sliding, bukan hanya tile tempat aktor berhenti. Tile posisi awal aktor pada awal gerakan tidak dihitung. Namun, tile tujuan akhir gerakan tetap dihitung karena tile tersebut dilalui dan menjadi posisi berhenti aktor.

Contoh:
Jika aktor melakukan gerakan ke kanan dan melewati 3 tile dengan cost 2, 5, dan 4 sebelum berhenti, maka cost gerakan tersebut adalah : 2 + 5 + 4 = 11.

Total cost solusi adalah penjumlahan seluruh cost movement dari setiap gerakan dalam solusi.
Tile X dan L tetap memiliki cost pada input, tetapi:

- Tile X tidak pernah dilewati karena merupakan rintangan/batu.
- Tile L menyebabkan game over jika dilewati, sehingga jalur yang melewati L dianggap tidak valid.

Dengan demikian, cost pada tile X dan L hanya digunakan sebagai bagian dari format input dan tidak dihitung dalam solusi valid

- **Output:**
    1. Menampilkan solusi gerakan yang ditemukan
    2. Menampilkan cost dari solusi
    3. Menampilkan visualisasi dari papan yang di mana oleh *Aktor* bisa mencapai Titik tujuan dengan algoritma *pathfinding* yang digunakan
    4. Menampilkan banyak konfigurasi atau iterasi yang ditinjau oleh algoritma, yang nantinya tiap iterasinya akan dapat disimpan dalam .txt
    5. Menampilkan waktu eksekusi program dalam *milisecond* (cukup waktu eksekusi, tidak termasuk membaca dan menulis file).

6. *Playback* (bukan *Live Update)*: program harus dapat memvisualisasikan proses *pathfinding* yang dilakukan. Visualisasi proses dilakukan setelah proses pencarian selesai dilakukan. Jika membuat bonus GUI, pastikan playback dapat dijalankan dengan kecepatan yang dapat diatur selayaknya video, serta dapat bergerak maju maupun mundur.
   Contoh flow untuk playback pada CLI :
   a. Arrow kiri/kanan untuk maju/mundur step
   b. ESC : User ingin lompat menuju ke step X. Prompt user untuk input suatu angka

Berikut adalah contoh output berdasarkan contoh input diatas

```
>> Masukan file input :
   file/to/input.txt
>> Algoritma apa yang anda pilih? (UCS/GBFS/A*)
   A*
>> Heuristic apa yang anda pilih? (H1/H2/H3)
   H2
Solusi Yang Ditemukan : RULUDRUR
Cost dari Solusi : 87

Initial
XXXXXXX
X0****X
X**X**X
X****OX
X1***LX
XZ**X*X
XXXXXXX

Step 1 : R
XXXXXXX
X0****X
X**X**X
X****OX
X1***LX
X**ZX*X
XXXXXXX

Step 2 : U
XXXXXXX
X0****X
X**X**X
X**Z*OX
X1***LX
X***X*X
XXXXXXX

…
…
…
```

```
Step 8 : R
XXXXXXX
X0****X
X**X**X
X****ZX
X1***LX
X***X*X
XXXXXXX

>> Waktu eksekusi: 10 ms
>> Banyak iterasi yang dilakukan: 9999 iterasi
>> Apakah Anda ingin melakukan playback? (Ya/Tidak) :
   Ya
>> Pada step berapa anda ingin melakukan playback :
   3
>> Apakah Anda ingin menyimpan solusi? (Ya/Tidak) :
   Ya
>> Solusi disimpan pada /path/to/solusi.txt
```

Spesifikasi Bonus

Poin maksimal untuk bonus adalah 10. Silahkan pilih spesifikasi bonus yang akan dikerjakan.

- Membuat GUI (Graphical User Interface) untuk program secara keseluruhan yang dapat memberi *input* dan memberi visualisi (max 6 poin)
- Tambahkan algoritma *pathfinding* selain yang tertera dalam spesifikasi wajib (max 2 poin)
- Tambahkan dua alternatif heuristik selain yang utama (max 2 poin)
- Kecepatan program dalam menemukan solusi (max 5 poin)

Pengumpulan dan Laporan

- Berkas laporan yang dikumpulkan adalah laporan dalam bentuk PDF yang setidaknya berisi:
    1. Algoritma *pathfinding* yang digunakan (termasuk bonus kalau mengerjakan), jelaskan langkah-langkahnya, bukan hanya notasi pseudocode.
    2. Analisis algoritma *pathfinding* yang diimplementasikan, minimal memuat jawaban pertanyaan berikut:
        a. Definisi dari f(n) dan g(n) pada algoritma.
        b. Apakah heuristik yang digunakan pada algoritma A* admissible?
        c. Secara teoritis, perbandingan algoritma yang diimplementasikan pada permasalahan ini.
    3. Source program dalam bahasa pemrograman yang dipilih (pastikan bahwa program telah dapat dijalankan).
    4. Tangkapan layar yang memperlihatkan input dan output, minimal sebanyak 5 buah contoh.
    5. Analisis hasil percobaan algoritma *pathfinding*.

6. Pranala ke repository yang berisi kode program.
7. **Pernyataan tidak melakukan kecurangan yang ditandatangani** dengan format sebagai berikut:

> Tugas ini disusun sepenuhnya tanpa bantuan kecerdasan buatan (*Generative AI*), melainkan hasil pemikiran dan analisis mandiri.
>
> [Tanda tangan mahasiswa]
> [Nama mahasiswa]

- Program disimpan dalam repository yang bernama Tucil3_NIM1_NIM2. Pastikan repository bersifat **public**. Berikut merupakan struktur dari isi repository tersebut:
    1. Folder src berisi source code program.
    2. Folder bin berisi executable file (Sesuaikan dengan bahasa pemrograman yang digunakan).
    3. Folder test berisi solusi jawaban dari data uji yang digunakan dalam laporan.
    4. Folder doc berisi laporan tugas kecil dalam bentuk PDF.
    5. README yang minimal berisi:
        a. Penjelasan singkat program yang dibuat.
        b. Requirement program dan instalasi tertentu bila ada.
        c. Cara mengkompilasi program bila perlu dikompilasi.
        d. Cara menjalankan dan menggunakan program.
        e. Author / identitas pembuat.
- Pendataan kelompok dapat diakses melalui link berikut
- Pertanyaan terkait tugas kecil 3 dapat diakses melalui link QNA berikut
- Pengumpulan tugas kecil 3 dapat dilakukan melalui link berikut

Catatan

- Dilarang keras copy paste program dari internet, AI, repository lain, ataupun program milik teman. Program yang dikerjakan menggunakan agent (Opus, Sonnet, dan lain-lain) akan sangat terlihat dan akan memiliki kemiripan dengan mahasiswa yang menggunakan hal serupa. Segala bentuk kecurangan akan diberikan sanksi berat yaitu nilai tugas menjadi nol. Kami tidak ingin hal itu terjadi dan kami ingin Anda untuk dapat memikirkan solusi algoritma dari hasil pemikiran Anda sendiri.
- Pastikan program dapat dikompilasi setidaknya pada windows dan linux.
- Apabila program tidak dapat dijalankan maka tidak akan dinilai oleh asisten.
- Tugas dikerjakan oleh maksimal dua orang
- Tambahkan tabel berikut yang diisi checklist (✓) pada bagian lampiran laporan Anda.

No	Poin	Ya	Tidak
1	Program berhasil di kompilasi tanpa kesalahan		

2	Program berhasil dijalankan		
3	Solusi yang diberikan program benar dan mematuhi aturan permainan		
4	Program dapat membaca masukan berkas .txt serta menyimpan solusi dalam berkas .txt		
5	Program memiliki Graphical User Interface (GUI)		
6	Ada algoritma *pathfinding* alternatif selain dari spesifikasi wajib (Algoritma X dan Algoritma Y)		
7	Ada tambahkan dua alternatif heuristik selain yang utama (Heuristik X dan Heuristik Y)		