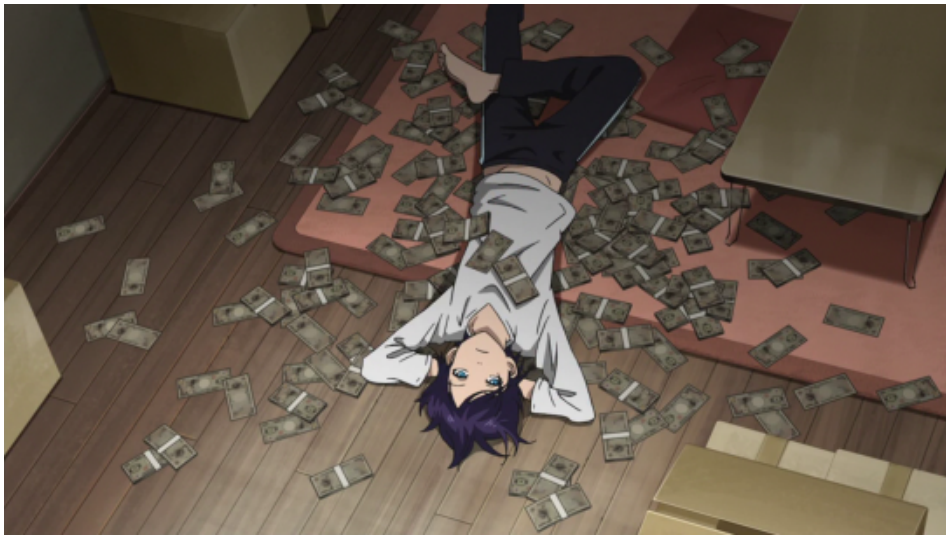
Tugas Besar Logika Komputasional – IF2121
I Quit My Job To Become A Professional Board Game Player
“Here’s The Reason Why I Did That. Number 8 Will Surprise You”

Versi: 8 November 2022
Versi: 15 November 2022

A. Topik

Membuat permainan papan Monopoly untuk seorang *programmer* yang beralih pekerjaan dengan menggunakan bahasa pemrograman deklaratif Prolog (gunakan GNU Prolog).



Gambar 1 Anda dengan uang mainan Monopoly yang bergelimpangan

Implementasi tugas besar harus mengandung materi:

1. Rekurens
2. *List*
3. *Cut*
4. *Fail*
5. *Loop*

B. Jadwal Penugasan/Penyetoran

No	Waktu	Kegiatan
1	Selasa, 8 November 2022	Pembahasan Pra-Praktikum, Rilis Tugas Besar IF2121
2	Selasa, 15 November 2022 (21:21)	Milestone 1: Asistensi Tugas Besar, pengerjaan, dan pengumpulan *draft* kasar tugas besar
3	Selasa, 22 November 2022 (21:21)	Milestone 2: Asistensi Tugas Besar dan pengumpulan *progress* kedua dari tugas besar
4	Senin, 28 November 2022 (21:21)	Milestone 3: Pengumpulan *final deliverables* Tugas Besar Logika Komputasional
5	Rabu, 30 November 2022 s.d. Minggu, 4 Desember 2022	Demo Tugas Besar secara sinkron melalui Google Meet

C. Tujuan

Tujuan dari tugas besar ini adalah mengkombinasikan berbagai keterampilan dan teknik yang telah dipelajari dalam perkuliahan Logika Komputasional IF2121, pra-praktikum, dan eksplorasi mandiri mengenai Logika Komputasional dan Prolog.

D. Domain Permasalahan

Steve adalah seorang *programmer* handal yang sudah mempunyai pengalaman setara 10 tahun dalam bidang Prolog, Assembly, NodeJS, TensorFlow, Flutter, Android Studio, dan (*sebutkan saja semua tech-stack yang kamu ketahui*). Dia sangat menyukai pekerjaannya hingga diangkat menjadi CTO (*Chief Technology Officer*) pada PT Gaib Ltd. Steve merasa semua pekerjaan *programming*-nya membosankan karena sudah terlanjur sangat handal di semua bidangnya sehingga ia mencari tujuan lain dalam hidupnya. Suatu ketika, saat Steve sedang melakukan *scrolling* TokTik, ia disuguhkan suatu permainan yang sangat menarik. Permainan ini merupakan *board game* bernama Monopoly. Terkagum-kagum dengan permainan tersebut, Steve segera membentuk tim untuk membuat pengembangan permainan Monopoly. Kebetulan, Anda mencantumkan *skill* yang sangat cocok untuk kebutuhan ini di platform LinkedOut, yaitu menamatkan *course* filosofi Monopoly. Anda langsung segera direkrut oleh Steve sebagai tim pengembang permainan ini agar Steve dapat menjadi pemain Monopoly profesional.

Tugas Anda adalah mengimplementasikan permainan papan Monopoly dengan fitur yang didefinisikan di bawah ini. Buatlah permainan **sekreatif mungkin**. Kreativitas Anda akan menjadi bagian dari penilaian asisten. Spesifikasi yang diberikan merupakan **batas minimum** yang harus dikerjakan. Oleh karena itu, sangat direkomendasikan untuk membuat permainan Monopoly yang lebih dari spesifikasi. Berikut beberapa spesifikasi yang harus Anda buat dalam program Anda:

1. Papan

Papan permainan berukuran 9 × 9. Setiap *tile* atau kotak diidentifikasikan dengan dua karakter (mutlak 2 karakter, supaya kalian menge-*print*-nya tidak susah). Sisi luar papan merupakan informasi pemilik papan dan jumlah bangunan yang sudah dibangun oleh pemilik. Berikut adalah contoh papannya.

	FP	E1	E2	E3	CC	F1	F2	F3	WT	
V0	D3				→				G1	
V2	D2								G2	V1
VL	D1								G3	
	TX	↑						↓	TX	
	C3								CC	
	C2								H1	W2
	C1				←				H2	W3
	JL	B3	B2	B1	CC	A3	A2	A1	GO	

Keterangan papan:

1. A-H → grup atau kelompok area; angka 1, 2, dan 3 hanya identifier supaya unik
2. V0 → informasi pemilik papan dan jumlah bangunan yang dimilikinya
   Format: [ID player][jumlah bangunan]
   Dalam kasus di atas, artinya D2 dimiliki oleh pemain V, dengan jumlah bangunan = 2.
   NB: jumlah bangunan bisa 0 (hanya memiliki tanah), 1, 2, 3, atau L (*landmark*).
3. GO → tempat start seluruh pemain
4. CC → *chance card*
5. TX → pajak (*tax*)
6. JL → penjara (*jail*)
7. FP → parkir gratis (*free parking*)
8. WT → *world tour*

Contoh Program

```
| ?- map.
                              Z1   ZL
    ---------------------------------------------
    | FP | E1 | E2 | E3 | CC | F1 | F2 | F3 | WT |
 V0 | D3 |----------------------------------| G1 |
 V2 | D2 |                                  | G2 | Y1
 VL | D1 |                                  | G3 |
    | TX |          M O N O P O L Y         | TX |
    | C3 |                                  | CC |
    | C2 |                                  | H1 | Z2
    | C1 |----------------------------------| H2 | Z3
    | JL | B3 | B2 | B1 | CC | A3 | A2 | A1 | GO |
    ---------------------------------------------
           W3   W2

    Posisi pemain:
    W = D1     Z = A1
    V = FP     Y = A1

(43 ms) yes
```

2. Lokasi

Setiap lokasi mempunyai informasinya masing-masing. Terdapat dua jenis lokasi, yakni properti dan non-properti. Deskripsi informasi yang disertakan pada masing-masing jenis adalah sebagai berikut.

- Untuk lokasi properti, informasi yang ditampilkan antara lain **nama**, **deskripsi**, **kepemilikan**, **biaya sewa** untuk yang menghampiri, **biaya akuisisi** (selain landmark), dan **tingkatan properti** saat ini. Jika belum ada yang memiliki properti tersebut, tampilkan bahwa properti belum dimiliki oleh siapapun.
- Untuk lokasi non-properti, informasi yang ditampilkan antara lain **nama** dan **deskripsi** bebas yang intuitif untuk menjelaskan spesifikasi lokasi berkaitan.

Perintah: `checkLocationDetail(x)` (lihat informasi lokasi kode x)

Contoh Program

```
| ?- checkLocationDetail(a1).   /* command dijalankan untuk melihat detail A1 */

Nama Lokasi         :  Jakarta
Deskripsi Lokasi    :  Ibukota Indonesia

Kepemilikan         :  Player V
Biaya Sewa Saat Ini :  10000
Biaya Akuisisi      :  7000
Tingkatan Properti  :  Bangunan 1

(25 ms) yes
```

```
| ?- checkLocationDetail(wt).  /* command dijalankan untuk melihat detail WT */
```

```
Nama Lokasi       :  World Tour
Deskripsi Lokasi  :  Anda dapat berpindah kemana saja, tentunya dengan membayar
uang sebesar X% aset Anda.

(20 ms) yes
```

```
| ?- checkLocationDetail(cc).  /* command dijalankan untuk melihat detail CC */

Nama Lokasi       :  Chance Card
Deskripsi Lokasi  :  Apakah Anda sedang merasa beruntung? Disinilah tempatnya!
Anda berhak mendapatkan salah satu dari kartu berikut :
1.  Deskripsi kartu 1
2.  Deskripsi kartu 2
3.  …


(27 ms) yes
```

```
| ?- checkLocationDetail(p100).  /* contoh masukan command yang tidak tepat */

P100 bukan merupakan lokasi yang valid! Silahkan masukkan lokasi yang tepat.

(10 ms) yes
```

3. Properti

Properti adalah lokasi yang dapat diakuisisi oleh pemain. Properti disimbolkan dengan sebuah karakter yang menunjukkan set warna dan sebuah digit yang menunjukkan urutannya pada map, misalnya properti A1 dan properti C3. Akuisisi semula dari properti berupa tanah kosong, dan properti tersebut dapat kemudian ditingkatkan mulai dari bangunan 1, bangunan 2, bangunan 3, hingga landmark. Peningkatan sebuah properti tidak bergantung pada properti lain (termasuk yang satu set warna). Peningkatan properti ke landmark hanya dapat dilakukan ketika properti telah memiliki bangunan 3 sebelumnya dan pemain telah menyelesaikan putaran pertamanya.

Jika properti belum diakuisisi oleh pemain manapun, pemain yang baru saja mencapai properti tersebut dapat mengakuisisi properti dan menambahkan hingga bangunan 3 di saat yang sama.

Jika properti telah diakuisisi oleh seorang pemain, pemain lain yang bukan merupakan pemilik properti harus membayar biaya sewa sesuai dengan tingkatan properti. Selanjutnya, pemain boleh memilih untuk mengakuisisi properti tersebut dengan membayar biaya akuisisi kepada pemilik semula sejumlah dua kali nilai aset properti saat ini dengan syarat properti belum memiliki landmark, kemudian memilih untuk melakukan peningkatan properti atau tidak.

Jika properti merupakan milik pemain yang mencapai properti tersebut, pemain dapat memilih untuk melakukan peningkatan properti.

Giliran pemain berakhir setelah aksi-aksi di atas dilakukan, kecuali jika pemain mendapatkan *double* pada `throwDice` sebelumnya.

Informasi properti mengandung poin sebagai berikut.
- Nama properti
- Deskripsi properti
- Biaya sewa (tanah, bangunan 1, bangunan 2, bangunan 3, landmark)
  Catatan: biaya sewa tidak bersifat kumulatif
- Harga beli properti: tanah, house, dan landmark

Detail properti dapat ditentukan secara mandiri dengan ketentuan harga beli dan biaya sewa dari tanah hingga landmark meningkat dan tidak absurd. Perhatikan pula nilai properti meningkat dari properti A1 hingga H2.

Pemain dapat melihat informasi properti melalui perintah `checkPropertyDetail(x)` dengan contoh sebagai berikut.

Contoh Program

```
| ?- checkPropertyDetail(a1).

Nama Properti         :  Jakarta
Deskripsi Properti    :  Ibukota Indonesia

Harga Tanah           :  200
Harga Bangunan 1      :  1000
Harga Bangunan 2      :  2000
Harga Bangunan 3      :  3000
Harga Landmark        :  3000

Biaya Sewa Tanah      : 20
Biaya Sewa Bangunan 1 : 120
Biaya Sewa Bangunan 2 : 350
Biaya Sewa Bangunan 3 : 600
Biaya Sewa Landmark   : 1000

(25 ms) yes
```

4. Chance Card

Pemain mendapatkan kartu tertentu secara acak apabila menginjak daerah ini. Tingkat keacakan/probabilitas munculnya masing-masing kartu dibebaskan. Jenis kartu yang **wajib diimplementasikan** adalah sebagai berikut.

- Kartu Tax: Pemain akan langsung pindah ke tempat Tax berikutnya (terdekat) dan langsung dikenai pajak.
- Kartu Hadiah: Pemain langsung mendapatkan uang berdasarkan nilai yang tertera pada kartu tersebut. Nilainya dapat diacak dan nominalnya dibebaskan selama tidak merusak *flow permainan* (*balanced*).

- Kartu Get Out From Jail: Pemain dapat menggunakan kartu ini saat berada di dalam penjara untuk langsung keluar tanpa menunggu tiga kali giliran atau membayar denda.
- Kartu Go To Jail: Pemain langsung ditransportasi ke lokasi Penjara dan dipenjara. Permainan dilanjutkan oleh pemain selanjutnya.

5. Perpajakan

Pemain yang mencapai daerah TX wajib membayar pajak sejumlah 10% dari total aset yang dimilikinya. Aset tersebut termasuk uang yang dimiliki dan semua nilai aset properti yang dimiliki.

6. Penjara

Pemain akan masuk penjara apabila pemain mendapatkan kartu masuk penjara atau mendapatkan *double* 3 kali berturut-turut. Pemain akan langsung dipindahkan ke “jail” dan akan diberi kesempatan untuk bermain dadu selama tiga kali giliran. Terdapat 4 mekanisme sebagai berikut untuk keluar dari penjara:

- Apabila terdapat dadu yang *double* sebelum tiga kali giliran, pemain langsung keluar dari penjara.
- Apabila pemain sudah melempar dadu sebanyak tiga kali, pemain keluar dari penjara.
- Pemain mempunyai kartu untuk lolos dari penjara. Pada giliran berikutnya, pemain dapat memilih untuk mengaktifkannya.
- Pemain dapat membayar pada giliran berikutnya sehingga lolos dari penjara lalu dapat langsung melempar dadu.

Daerah penjara dapat dilewati atau ditempati oleh pemain pada umumnya, tetapi pemain tersebut tidak masuk penjara atau istilahnya “just visiting”.

7. Parkir Gratis

Pemain yang menginjak daerah ini tidak mendapatkan efek apa-apa. Pemain dapat melakukan giliran setelahnya seperti biasa.

8. World Tour

Bila pemain mendarat di kotak “World Tour”, maka pemain akan diberikan kesempatan untuk berpindah ke lokasi manapun di map **kecuali kotak World Tour.** Perpindahan pemain tidak menggunakan konsep *teleport* (langsung berpindah dari kotak World Tour ke kotak tujuan akhir) melainkan berjalan melewati kotak-kotak lainnya. Artinya, jika untuk mencapai kotak tujuan pemain harus melewati titik GO, maka pemain akan mendapatkan uang sesuai dengan nominal yang telah ditentukan. Pemain yang ingin menggunakan World Tour harus membayar sejumlah nominal yang nilainya dibebaskan.

9. Pemain

Total pemain pada permainan Monopoly ini adalah 2 pemain dengan pemain saling berlawanan satu sama lain. Aturan dasar berkaitan dengan pemain adalah sebagai berikut.

- Setiap pemain diberi uang pada awal permainan yang nominalnya dibebaskan.
- Setiap pemain dapat mempunyai aset serta mempunyai informasi total nilai aset yang dimiliki.
- Setiap pemain berhak mendapatkan uang sesuai dengan nominal yang ditentukan saat pemain menginjak atau melewati daerah GO untuk kali kedua, ketiga, dan seterusnya.

Terdapat juga fitur untuk mengetahui kondisi pemain x dimana nantinya akan disertakan informasi-informasi pada saat *command* dijalankan sebagai berikut.

- Lokasi keberadaan pemain x
- Total uang (*cash-on-hand*) yang dimiliki pemain x
- Total nilai properti, yaitu penjumlahan nilai beli semua properti yang telah dimiliki oleh pemain x.
  Sebagai informasi tambahan, misalkan pemain x memiliki **landmark** pada D2, maka nilai properti pada D2 adalah penjumlahan dari harga **tanah** + harga **bangunan 1** + harga **bangunan 2** + harga **bangunan 3** + harga **landmark**.
- Total aset, yaitu penjumlahan dari total uang dengan total nilai properti
- Daftar kepemilikan properti, memberikan keluaran berupa daftar properti yang dimiliki oleh player x. Setiap properti memiliki format `IDlokasi - Tingkatan Bangunan`.
- Daftar kepemilikan *card*, memberikan keluaran berupa daftar *card* yang dimiliki oleh player x

Perintah: `checkPlayerDetail(x)` (dengan x : representasi variabel pemain)

Contoh Program

```
| ?- checkPlayerDetail(v).   /* command untuk melihat detail pemain V */

Informasi Player V

Lokasi                    :  A1     /* lokasi keberadaan pemain V */
Total Uang                :  6000   /* cash-on-hand pemain V */
Total Nilai Properti      :  15000  /* total nilai properti pemain V */
Total Aset                :  21000  /* total nilai properti + cash pemain V */

Daftar Kepemilikan Properti :       /* daftar properti yang dimiliki pemain V */
1. A1 - Bangunan 2
2. D2 - Landmark
3. D3 - Bangunan 1
4. E1 - Tanah

Daftar Kepemilikan Card     :       /* daftar card yang dimiliki pemain V */
1. Angel Card
```

```
2. Get Out From Jail Card


(18 ms) yes
```

10. Dadu

Pada permainan Monopoly, tentunya terdapat dadu untuk melakukan giliran. Dadu di permainan ini direpresentasikan sebagai angka acak **1-6** untuk masing-masing dadu (terdapat **dua** dadu). Pemain akan melangkah sebanyak angka yang ditunjukkan pada dadu. Apabila pemain mendapatkan dua dadu dengan angka yang sama, pemain dapat melakukan gilirannya kembali. Ketika pemain mendapatkan dua angka yang sama 3 kali berturut-turut, pemain akan masuk penjara.

Sebelum melempar dadu, disarankan mengimplementasikan deteksi bangkrut yang dijelaskan lebih lanjut pada spesifikasi nomor 11. Selain itu, dapat ditambahkan juga deskripsi giliran pemain.

Tambahan: untuk membuat dadu menjadi lebih acak, dapat menambah `randomize` setiap kali `throwDice` digunakan.

Contoh Program

```
| ?- throwDice.
Sekarang adalah giliran pemain 1.

Dadu 1: 1.
Dadu 2: 2.
Anda maju sebanyak 3 langkah.

(10 ms) yes
```

```
| ?- throwDice.
Sekarang adalah giliran pemain 1.

Dice 1: 6.
Dice 2: 6.
Double!
Anda maju sebanyak 12 langkah.

(10 ms) yes


| ?- throwDice.
Sekarang adalah giliran pemain 1.

Dice 1: 5.
Dice 2: 5.
Double!
Anda maju sebanyak 10 langkah.

(10 ms) yes
```

```
| ?- throwDice.
Sekarang adalah giliran pemain 1.

Dice 1: 6.
Dice 2: 6.
Double!
Anda masuk ke jail karena mendapatkan Double 3 kali berturut-turut.

(10 ms) yes
```

11. Mekanisme Bangkrut

Pemain dinyatakan kalah apabila pemain tidak lagi mempunyai total kekayaan (uang yang dipegang + total kekayaan properti) yang cukup jika dikenai biaya denda atau biaya sewa.

Jika uang yang dipegang pemain tidak cukup, tetapi pemain memiliki kekayaan properti yang bisa dijual (dan cukup untuk melunasi hutang), pemain dapat memilih untuk menjual propertinya untuk melanjutkan permainan atau menyerah (*surrender*). Hasil penjualan properti adalah 80% dari harga beli. Pemain hanya dapat menjual *tile* atau kotak secara keseluruhan (tidak bisa parsial, misal mengurangi jumlah bangunan).

Kalian disarankan untuk mengimplementasikan "deteksi bangkrut" di dalam `throwDice`. Narasi dan pesan permainan dibebaskan kepada kalian, silakan berkreasi. Yang penting alurnya jelas.

Contoh Program

```
| ?- /* throw dice sudah dipanggil, ternyata terkena biaya dengan rincian sbb:
Uang yang dipegang: 300
Total kekayaan properti: 225
Total kekayaan: 300 + 225 = 525

Biaya sewa: 400
*/

Wah, uangmu kurang! Apakah kamu ingin tetap meneruskan permainan?

(10 ms) yes.

| ?- lanjut.

Daftar propertimu:
1. A2, bangunan 1 : 25
2. H2, bangunan 3 : 150
3. D3, bangunan 1 : 50

Properti mana yang ingin dijual?

(10 ms) yes

| ?- 1.
Uangmu: 325
Total kekayaan properti: 200
```

```
Pembayaran biaya sewa masih kurang!
Daftar propertimu:
1. H2, bangunan 3 : 150
2. D3, bangunan 1 : 50

Properti mana yang ingin dijual?
(10 ms) yes

| ?- 1.
Uangmu: 475
Total kekayaan properti: 50

Hore, sewa sudah bisa dibayar!
Uang setelah membayar sewa: 75

/* Lanjutkan permainan */
```

```
| ?- /* throw dice sudah dipanggil, ternyata terkena biaya dengan rincian sbb:
Uang yang dipegang: 300
Total kekayaan properti: 225
Total kekayaan: 300 + 225 = 525

Biaya sewa: 400
*/

Wah, uangmu kurang! Apakah kamu ingin tetap meneruskan permainan?

(10 ms) yes.

| ?- tidak.

Sayang cepat sekali menyerah, selamat tinggal :)

(10 ms) no.
```

```
| ?- /* throw dice sudah dipanggil, ternyata terkena biaya dengan rincian sbb:
Uang yang dipegang: 300
Total kekayaan properti: 225
Total kekayaan: 300 + 225 = 525

Biaya sewa: 1000
*/

Sayang sekali, uangmu sudah tidak cukup. Selamat tinggal :)

(10 ms) no.
```

12. Bonus

a. “Apalah arti color set kalau tidak ada gunanya.” Implementasikan fungsi color set: seluruh harga rent dari properti pada satu color set yang dimiliki oleh seorang pemain akan menjadi dua kali lipatnya. Efek color set hilang jika pemain kehilangan salah satu properti dalam color set tersebut.

b. Implementasikan World Cup atau Festival seperti di Line Get Rich. Fitur ini memungkinkan sebuah lokasi untuk bernilai dua kali lipat harga rentnya.

Perlu dicatat bahwa ini akan mengakibatkan harga lokasi bisa bernilai empat kali dari harga rent jika bonus color set diimplementasikan. World Cup akan bertahan selama 3 putaran dan dapat diperbarui statusnya jika pemain mendarat kembali di kotak World Cup sebelum efek World Cup habis di sebuah kotak lokasi.

c. Bonus *chance cards*. Implementasi dari bonus ini adalah menambahkan variasi *chance cards* selain 4 kartu yang telah didefinisikan pada spesifikasi wajib. Perlu diperhatikan agar *chance card* tambahan tidak merusak *flow balance* permainan.

d. Buildable Go
   Ketika player berada tepat di kotak GO, player bisa menambah bangunan atau mengubah bangunan menjadi *landmark*. Kotak yang bisa ditambah atau diubah adalah kotak yang dimiliki pemain.

e. Minigame berupa *coin flip*. Implementasi dari bonus ini adalah dengan mengganti salah satu kotak pada papan menjadi kotak *coin flip*. Pemain yang menginjak kotak ini dapat memainkan *mini game* berupa *coin flip* dengan aturan sebagai berikut:
   - Pemain menebak koin yang dilempar (*head/tail*).
   - Jika pemain menebak dengan benar, pemain dapat menebak untuk berikutnya sampai maksimal 3 kali. Jika salah, *coin flip* diakhiri.
   - Di akhir minigame, pemain diberikan uang yang sebanding dengan jumlah tebakan yang benar.

f. Kreativitas tambahan dalam implementasi *game* akan selalu diapresiasi (contoh: alur cerita, animasi dalam *activity mechanism*, fitur-fitur tambahan lainnya).

E. Kelompok

Pembagian kelompok ditentukan sendiri oleh mahasiswa dengan mengisi sheet kelompok berikut ini dengan 1 kelompok terdiri dari 5 mahasiswa yang berasal dari kampus yang sama dan diperbolehkan lintas kelas. Silakan lihat instruksi lebih lanjut di *sheet* kelompok di atas.

F. QnA

Pertanyaan dapat ditanyakan pada link **QnA** berikut (link sama dengan pra-praktikum). Pastikan pertanyaan yang ditanyakan tidak berulang.

G. Aturan

Terdapat beberapa hal yang harus diperhatikan dalam pengerjaan tugas ini, yakni:

1. *Query* dan keluaran tidak harus persis dengan contoh program yang diberikan. Hal-hal yang tidak dispesifikkan dapat dianggap sebagai hal yang bebas diimplementasikan seperti apa, silakan berkreativitas sendiri.
2. Semua perubahan yang mungkin Anda lakukan di dalam implementasi Anda, ataupun fitur-fitur lain yang Anda ubah atau tambahkan, dapat dituliskan di laporan agar dapat dinilai oleh asisten dengan lengkap.
3. Apabila Anda mencari dan mencontoh kode dari Internet, harap cantumkan sumbernya dalam bentuk komentar dalam program.
4. Jika terdapat hal yang tidak dimengerti, silahkan ajukan pertanyaan kepada asisten melalui link QnA yang telah diberikan di atas. Pertanyaan yang diajukan secara personal ke asisten tidak akan dijawab untuk menghindari perbedaan informasi yang didapatkan oleh peserta praktikum.
5. Dilarang melakukan **plagiarisme**. Pelanggaran pada poin ini akan menyebabkan pemberian **nilai E** pada setiap anggota kelompok yang melakukan maupun memberi.

H. Asistensi

Pada setiap deadline pengumpulan Milestone, tim asisten mewajibkan setiap kelompok untuk melakukan asistensi. Asistensi diharapkan menjadi kesempatan bagi kelompok untuk melakukan pengecekan akhir sebelum melakukan submisi di Github. Setiap anggota kelompok diharapkan untuk aktif bertanya dan berinteraksi di asistensi namun pertanyaannya dibatasi ke pertanyaan *technical*. Pertanyaan terkait spesifikasi tugas besar tetap harus ditanyakan melalui sheets QnA.

I. Deliverables

1. Untuk tugas ini Anda diwajibkan menggunakan *version control system* **git** dengan menggunakan sebuah *repository* ***private*** di Github Classroom "**GAIB20**" (gunakan surel *student* agar gratis).

2. Anda diwajibkan untuk membuat *repository* di dalam Github Classroom menggunakan link invitation berikut ini. Berikut adalah alur untuk membuat *repository*:

    a. Ketua kelompok mengisi kolom 'NIM 1' dan 'Username GitHub 1' pada sheet kelompok dan anggota lainnya mengisi kolom 'NIM 2' sampai 'NIM 5' beserta 'Username GitHub 2' sampai 'Username GitHub 5'.

    b. Semua anggota kelompok memilih *identifier* yang sesuai dengan kelas, NIM, dan nama mereka.

    c. Ketua kelompok membuat *team* baru dengan nama kelompok yang sama dengan yang dicantumkan di sheet kelompok.

d. Anggota kelompok melakukan *join* ke *team* yang sudah dibuat ketua kelompok.

3. Meskipun commit tidak dinilai, lakukanlah *commit* yang wajar dan sesuai *best practice* (tidak semua kode satu *commit*). Disarankan juga menggunakan *semantic commits* agar asisten lebih mudah menilai beban pekerjaan masing-masing anggota kelompok.

4. *Repository* memiliki struktur sebagai berikut.

```
nama-repository
├── src
│   ├── file1.pl
│   ├── file2.pl
│   └── main.pl
├── doc
│   └── Laporan_KXX_GYY.pdf
└── README.md
```

5. Pengumpulan dilakukan dengan membuat ***release*** dengan tag **vX.Y** pada repository yang telah kelompok Anda buat sebelum deadline, dengan **X** adalah angka sesuai dengan nomor milestone dan **Y** adalah angka revisi dimulai dari 0 jika tidak ada revisi/versi awal. Pastikan tag sesuai format.

6. Berikut adalah informasi tambahan mengenai Milestone dan *deliverables*-nya:

    a. Pada Milestone 1, Anda diminta untuk paling tidak membuat **detail lengkap rencana fakta dan rule** yang akan digunakan, baik yang menggunakan list, control loop dalam bentuk .txt file. Anda juga diminta untuk melampirkan ~~sebuah .txt file~~ sebuah file yang formatnya dibebaskan (dapat berupa .pdf, .txt, atau .docx) yang berisikan NIM dan nama masing-masing anggota kelompok serta pembagian tugasnya. Anda juga diminta untuk menuliskan progress yang sudah dilakukan secara keseluruhan dalam format:

        i. COMPLETED TASKS

        ii. ONGOING TASKS

        iii. UNSTARTED TASKS

    b. Pada Milestone 2, Anda diminta paling tidak sudah mengimplementasikan fakta-fakta terkait permainan dan.Implementasi rule-rule kendali dasar dalam bentuk *source code*. Anda juga diminta untuk menuliskan progress yang sudah dilakukan secara keseluruhan dalam format yang sama dalam poin a. Cantumkan juga bila ada tambahan pembagian tugas dan fitur.

    c. Pada Milestone 3, Anda diminta untuk mengumpulkan berkas-berkas berikut:

i. Source code program keseluruhan
ii. Laporan hasil kerja dengan format penamaan penamaan **Laporan_KXX_GYY.pdf** dengan XX adalah nomor kelas dan YY adalah nomor kelompok yang terdiri atas:
    1. Halaman *cover* yang memuat judul tugas, kode dan nama mata kuliah, dan identitas anggota kelompok
    2. Penjelasan setiap *command* yang kelompok Anda telah buat, termasuk command bonus (jika ada), penjelasan meliputi:
        a. Kegunaan command tersebut
        b. Skenario-skenario penggunaannya (beserta contoh)
        c. Tidak perlu menjelaskan cara kerja *command*
    3. Hasil eksekusi program berupa jalannya alur permainan (dalam bentuk *screenshot*)
    4. Pembagian dan persentase kerja masing-masing anggota kelompok
iii. README file. Gunakan *template* ini untuk membantu pengerjaan kalian.

7. Tugas Besar IF2121 - Logika Komputasional adalah kegiatan yang bersifat **mandiri berkelompok**. Semua bentuk kecurangan akan ditindaklanjuti sesuai dengan sanksi akademik yang ada.