

IF2121 - Logika Komputasional

Tugas Besar

CAMEL UP!



Disusun Oleh:

13523021	M. Raihan Nazhim Oktana
13523005	Muhammad Alfansya
13523031	Rafen Max Alessandro
13523087	Grace Evelyn Simon

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

2024

Daftar Isi

Daftar Isi.....	2
BAB 1: Deskripsi Masalah.....	3
BAB 2: Penjelasan Command.....	5
2.1. Command startGame.....	5
2.1.1. Kegunaan Command.....	5
2.1.2. Skenario-Skenario Penggunaan.....	5
2.2. Command displayMap.....	5
2.2.1. Kegunaan Command.....	5
2.2.2. Skenario-Skenario Penggunaan.....	6
2.3. Command cekInfo.....	6
2.3.1. Kegunaan Command.....	6
2.3.2. Skenario-Skenario Penggunaan.....	7
2.4. Command investasi.....	8
2.4.1. Kegunaan Command.....	8
2.4.2. Skenario-Skenario Penggunaan.....	8
2.5. Command papanInvestasi.....	9
2.5.1. Kegunaan Command.....	9
2.5.2. Skenario-Skenario Penggunaan.....	9
2.6. Command jalankanUnta.....	9
2.6.1. Kegunaan Command.....	9
2.6.2. Skenario-Skenario Penggunaan.....	10
2.7. Command pasangTrap.....	10
2.7.1. Kegunaan Command.....	10
2.7.2. Skenario-Skenario Penggunaan.....	10
2.8. Command nextTurn.....	12
2.8.1. Kegunaan Command.....	12
2.8.2. Skenario-Skenario Penggunaan.....	12
2.9. Command tukarUnta.....	12
2.9.1. Kegunaan Command.....	12
2.9.2. Skenario-Skenario Penggunaan.....	12
BAB 3: Hasil Eksekusi Program.....	13
Lampiran.....	20
Referensi.....	21

BAB 1: Deskripsi Masalah

Camel POP! merupakan sebuah *board game* yang dibuat dengan menggunakan bahasa pemrograman deklaratif Prolog (GNU Prolog). Camel POP! Berperan sebagai tugas besar pada mata kuliah IF1221 Logika Komputasional yang menjadi wadah bagi mahasiswa jurusan Teknik Informatika dalam mengkombinasikan berbagai keterampilan dan teknik yang telah dipelajari dalam perkuliahan Logika Komputasional IF1221, pra-praktikum, dan eksplorasi mandiri mengenai Logika Komputasional dan Prolog. Implementasi tugas besar ini mengandung materi mengenai Rekurens, List, Cut, Fail, Loop yang menjadi materi pembelajaran pada keberlangsungan pembelajaran mata kuliah.

Alur permainan Camel POP! diinisiasi dengan start game. *Command* ini akan melakukan *setup* berupa pengacakan urutan pemain serta inisiasi kartu, poin, dan *trap* tiap pemain. Permainan Camel POP! dapat dimainkan oleh 2-4 orang pemain. Pada tiap giliran pemain, pemain dapat melakukan maksimal 1 aksi di antara: investasi, jalankan unta, pasang trap (juga sembunyikan unta, restart unta, tukar unta, atau God's Hand jika mengerjakan spesifikasi bonus). *Command* untuk menampilkan peta keberlangsungan permainan (display map) dan layanan informasi karakteristik pemain lain di dalam permainan (cek info) dapat dipanggil secara tidak terbatas, yaitu tidak terpaku terhadap apakah pemain sedang memiliki gilirannya atau tidak. Pemain dapat mengakhiri gilirannya dengan menjalankan *command end turn*.

Permainan akan berakhir ketika terdapat unta yang melewati garis *finish* (angka dadu untuk melewati garis *finish* tidak harus tepat dengan jumlah langkah yang dibutuhkan untuk sampai ke garis *finish*), dan program akan secara otomatis menjalankan *command end game* yang menentukan pemenang permainan berdasarkan poin yang didapatkan. Poin didapatkan berdasarkan urutan investasi pemain terhadap unta dan kemenangan unta. Urutan unta ketika terdapat unta yang telah melewati garis *finish* menjadi dasar faktor penentu besaran poin yang diterima pemain. Besar poin ini kemudian proporsional dengan urutan pemain dalam melakukan investasi terhadap unta tersebut.

Karakteristik utama dari permainan Camel POP! adalah kemampuan unta untuk menunggangi unta lainnya jika unta mendatangi petak yang telah terdapat unta sebelumnya. Unta yang baru datang akan berada di atas unta sebelumnya dan mengikuti pergerakan unta di bawahnya. Lalu, jika ada unta yang melewati garis *finish* secara bersamaan, urutan kemenangan diawali dengan unta yang berada di paling atas.

Secara garis besar, Camel POP! merupakan permainan sederhana dengan mekanisme yang cukup kompleks. Menggunakan GNU Prolog, Camel POP! berjalan sebagai permainan yang mengandalkan logika dan penggunaan sintaks Prolog yang tepat dalam menghasilkan algoritma yang dapat membawa unta ke garis *finish*, baik oleh unta itu sendiri ataupun oleh tunggangan unta lain.

BAB 2: Penjelasan Command

2.1. Command startGame

2.1.1. Kegunaan Command

Command startGame dibuat untuk menginisiasi permainan, diawali dengan memasukkan jumlah pemain (berkisar antara 2-4 pemain), kemudian memasukkan jumlah nama sesuai dengan jumlah pemain yang telah diinput sebelumnya. Program ini menggunakan permutasi untuk *list* segala kemungkinan urutan pemain, kemudian diambil salah satu urutan pemain tersebut menggunakan fungsi bawaan *random* yang disediakan oleh GNU Prolog. *Command* startGame juga menginisialisasi penambahan 4 kartu (warna merah, kuning, hijau, biru), menambahkan 30 poin, serta menambahkan 1 trap untuk masing-masing pemain.

2.1.2. Skenario-Skenario Penggunaan

- *Command* dipanggil untuk menginisiasi permainan

Ketika dynamic predicate *playing/0* bukan merupakan sebuah fakta, maka tidak ada permainan yang sedang berlangsung, sehingga pemanggilan command startGame akan memulai suatu permainan baru.

- *Command* dipanggil ketika terdapat permainan yang sedang berlangsung

Jika *command* dipanggil ketika sebuah permainan sedang berlangsung, yaitu ketika *dynamic predicate playing/0* merupakan sebuah fakta, maka pemanggilan *command* akan memberikan keluaran “Permainan sedang berlangsung! Anda tidak dapat memulai permainan baru”.

2.2. Command displayMap

2.2.1. Kegunaan Command

Command displayMap bertanggung jawab untuk menampilkan peta ketika permainan sedang berlangsung. Ketika pemain menggunakan *command* ini dalam giliran mereka, akan ditampilkan sebuah peta yang menunjukkan

keberlangsungan peta di tengah permainan, menampilkan dinamika setiap petak ketika *command* dipanggil, posisi setiap unta dengan urutannya dalam suatu tumpukan, serta posisi jebakan dan jenisnya (jebakan maju atau jebakan mundur).

2.2.2. Skenario-Skenario Penggunaan

- *Command* dipanggil ketika permainan sedang berlangsung

Sebagai *command* yang menjelaskan dinamika permainan yang sedang berlangsung, *command* displayMap hanya dapat dipanggil dan dijalankan sesuai fungsinya ketika permainan sedang berlangsung. Kondisi apakah permainan sedang berlangsung atau tidak dinyatakan melalui *dynamic predicate playing/0* yang menjadi suatu fakta apabila permainan sedang dijalankan dan sebaliknya

- *Command* dipanggil ketika permainan belum dimulai

Ketika *command* dipanggil dalam kondisi permainan belum dimulai, yaitu ketika *dynamic predicate playing/0* bukan merupakan suatu fakta, maka fungsi tidak akan menampilkan peta, melainkan memberikan keluaran berupa “Permainan belum dilakukan, silakan melakukan perintah startGame terlebih dahulu!”.

2.3. Command cekInfo

2.3.1. Kegunaan Command

Command cekInfo memberikan informasi kepada pemain mengenai karakteristik yang dimiliki oleh salah satu pemain ketika *command* dipanggil. *Command* ini termasuk ke dalam *command* yang hanya dapat dipanggil ketika permainan sedang dijalankan. Selain itu, *command* juga meminta pengguna untuk memasukkan nama pemain yang akan dilihat informasinya. Maka, dilakukan validasi apakah pengguna memasukkan nama pemain yang valid atau tidak, yaitu nama pemain yang sedang menjadi pemain dari permainan yang sedang berlangsung.

2.3.2. Skenario-Skenario Penggunaan

- *Command* dipanggil terhadap salah satu nama pemain

Kondisi yang perlu dipenuhi bagi *command* cekInfo untuk menjalankan fungsinya sebagaimana mestinya adalah berada di dalam permainan dan dipanggil untuk pemain yang sedang bermain. Keberlangsungan permainan dinyatakan menggunakan *dynamic predicate playing/0* yang berperan sebagai fakta ketika permainan sedang berlangsung, dan validasi apakah masukan nama pemain valid disesuaikan dengan *dynamic predicate pemain/1* yang berperan sebagai fakta meliputi nama-nama pemain yang sedang bermain.

Jika kedua kondisi dipenuhi, maka *command* akan menampilkan nama pemain, poin yang dimiliki pemain, warna kartu yang dimiliki oleh pemain, serta kepemilikan pemain terhadap jebakan. Setiap bentuk informasi diambil dari *dynamic predicate* masing-masing yang menjadi representasi kepemilikan pemain terhadap setiap komponen penilaian.

- *Command* dipanggil untuk nama pemain yang tidak valid

Ketika *command* dipanggil untuk nama pemain yang tidak valid, yaitu yang tidak menjadi salah satu pemain dalam permainan yang sedang berlangsung, maka *command* akan menampilkan keluaran

```
format('Tidak terdapat pemain dengan nama ~w!', [Pemain])
```

yang menyatakan bahwa pengguna telah memasukkan masukan nama pemain yang tidak sedang bermain pada permainan yang sedang berlangsung.

- *Command* dipanggil ketika permainan belum dimulai

Ketika *command* dipanggil dalam kondisi permainan belum dimulai, yaitu ketika *dynamic predicate playing/0* bukan merupakan suatu fakta, maka fungsi tidak akan melanjutkan tahapan dengan meminta masukan nama pemain, melainkan memberikan keluaran berupa “Permainan belum dilakukan, silakan melakukan perintah startGame terlebih dahulu!”.

2.4. Command investasi

2.4.1. Kegunaan Command

Command investasi dipanggil ketika pemain ingin melakukan investasi. Pemain dapat memilih unta sesuai dengan kartu warna miliknya yang masih tersisa. Pemain akan diminta memasukan warna unta yang ingin diinvestasi dan kemudian akan menampilkan papan investasi milik unta yang dipilih pemain jika masukan warna valid dan akan menampilkan pesan jika masukan warna tidak valid.

2.4.2. Skenario-Skenario Penggunaan

- *Command* dipanggil oleh pemain yang sedang giliran bermain

Kondisi yang perlu dipenuhi bagi *command* investasi untuk menjalankan fungsinya sebagaimana mestinya adalah berada di dalam permainan dan dipanggil untuk pemain yang sedang bermain. Keberlangsungan permainan dinyatakan menggunakan *dynamic predicate playing/0* yang berperan sebagai fakta ketika permainan sedang berlangsung, dan *dynamic predicate* pemain/1 akan menyimpan nama pemain yang sedang mendapat giliran. Jika kondisi dipenuhi, maka *command* akan menerima input warna pilihan pemain.

- *Command* dipanggil ketika permainan belum dimulai

Ketika *command* dipanggil dalam kondisi permainan belum dimulai, yaitu ketika *dynamic predicate playing/0* bukan merupakan suatu fakta, maka fungsi tidak akan melanjutkan tahapan dengan meminta masukan warna unta, melainkan memberikan keluaran berupa “Permainan belum dilakukan, silakan melakukan perintah startGame terlebih dahulu!”.

- Pemain memasukan warna unta dan memiliki kartu warna tersebut

Jika masukan warna unta valid, maka pemain akan masuk ke daftar pemain yang berinvestasi ke unta tersebut dan *command* akan menampilkan papan investasi milik unta yang dipilih.

- Pemain memasukan warna unta tapi tidak memiliki kartu warna tersebut

Jika masukan warna unta tidak valid, maka *command* akan menampilkan pesan error dan akan meminta inputan ulang.

2.5. Command papanInvestasi

2.5.1. Kegunaan Command

Command dipanggil ketika pemain ingin melihat papan investasi yang berisikan daftar nama pemain yang berinvestasi pada tiap unta. Daftar pemain ditampilkan urut berdasarkan urutan berinvestasi pada unta tersebut. *Command* ini tidak meminta masukan ke pemain.

2.5.2. Skenario-Skenario Penggunaan

- *Command* dipanggil ketika permainan belum dimulai

Ketika *command* dipanggil dalam kondisi permainan belum dimulai, yaitu ketika *dynamic predicate playing/0* bukan merupakan suatu fakta, maka fungsi tidak akan melanjutkan tahapan dengan meminta masukan warna unta, melainkan memberikan keluaran berupa “Permainan belum dilakukan, silakan melakukan perintah startGame terlebih dahulu!”.

- *Command* dipanggil oleh salah satu pemain.

Ketika *command* dipanggil, akan ditampilkan papan investasi untuk masing masing unta dan daftar pemain yang berinvestasi ke unta tersebut.

- *Command* dipanggil ketika belum ada yang berinvestasi.

Ketika *command* dipanggil tapi belum ada pemain yang melakukan investasi. Papan untuk unta yang belum ada yang berinvestasi tersebut akan menampilkan “Belum ada investasi”

2.6. Command jalankanUnta

2.6.1. Kegunaan Command

Command jalankanUnta dipanggil ketika pemain ingin melakukan gilirannya untuk menjalankan unta. Pemain tidak dapat memilih unta yang akan dijalankan, tetapi warna unta yang bergerak akan ditentukan oleh kocokan dadu

yang keluar secara random dengan kemungkinan keluaran yang sama rata. Pemain juga tidak dapat menentukan banyak langkah unta tersebut dapat berjalan, hal tersebut juga ditentukan oleh kocokan dadu yang keluar secara random dengan kemungkinan keluaran yang sama rata. Ketika proses telah dijalankan, program akan mengeluarkan output yang sesuai dan menampilkan map terbaru.

2.6.2. Skenario-Skenario Penggunaan

- *Command* dipanggil oleh pemain yang sedang giliran bermain

Ketika *command* dipanggil dalam kondisi permainan sudah dimulai dan saat giliran bermain, fungsi *kocokDadu* akan terpanggil secara otomatis untuk mengocok dadu warna dan dadu angka secara otomatis. Secara umum, dadu warna akan memilih warna yang tersedia, lalu menghapus pilihan warna tersebut, dan berulang sampai pilihan warna habis dan akan di-*reset*. Langkah unta yang dipilih ditentukan berdasarkan kondisi hasil kocokan dadu tersebut. Jika sudah ada minimal satu unta yang mencapai *finish*, *command* ini tidak dapat dipanggil.

- *Command* dipanggil ketika permainan belum dimulai

Ketika *command* dipanggil dalam kondisi permainan belum dimulai, yaitu ketika *dynamic predicate playing/0* bukan merupakan suatu fakta, maka fungsi tidak akan melanjutkan tahapan dengan meminta masukan nama pemain, melainkan memberikan keluaran berupa “Permainan belum dilakukan, silakan melakukan perintah startGame terlebih dahulu!”.

2.7. Command pasangTrap

2.7.1. Kegunaan Command

Command *pasangTrap* dipanggil ketika pemain ingin melakukan gilirannya untuk memasang *trap*. Pemain dapat menentukan pada petak mana *trap* ingin dipasang serta jenis *trap* yang ingin dipasang (maju / mundur).

2.7.2. Skenario-Skenario Penggunaan

- *Command* dipanggil saat permainan sudah dimulai dan *trap* berhasil dipasang

Ketika *command* dipanggil dalam kondisi permainan sudah dimulai, pemilihan petak valid (terdefinisi), tidak ada unta apapun pada petak tersebut, serta petak sebelum dan setelahnya tidak ada *trap* yang terpasang, dan pemain yang sedang bermain masih memiliki *trap* untuk dipasang (*dynamic predicate* jebakan/1 untuk pemain tersebut masih ada), maka pemasangan *trap* dianggap berhasil. Ketika proses ini berhasil, maka akan menghapus *dynamic predicate* jebakan/1 untuk pemain tersebut. Ketika proses telah dijalankan, program akan mengeluarkan output yang sesuai.

- *Command* dipanggil saat permainan sudah dimulai tetapi *trap* gagal terpasang

Ketika *command* dipanggil dalam kondisi permainan sudah dimulai, tetapi pemilihan petak tidak valid (tidak terdefinisi), atau ada unta apapun pada petak tersebut, atau petak sebelum ataupun setelahnya sudah ada *trap* yang terpasang, atau pemain yang sedang bermain sudah tidak memiliki *trap* untuk dipasang (*dynamic predicate* jebakan/1 untuk pemain tersebut sudah tidak ada), maka pemasangan *trap* dianggap gagal. Ketika proses ini terjadi, status *dynamic predicate* jebakan/1 untuk pemain tersebut tidak akan berubah. Ketika proses telah dijalankan, program akan mengeluarkan output yang sesuai dengan kesalahan yang terjadi.

- *Command* dipanggil ketika permainan belum dimulai

Ketika *command* dipanggil dalam kondisi permainan belum dimulai, yaitu ketika *dynamic predicate playing/0* bukan merupakan suatu fakta, maka fungsi tidak akan melanjutkan tahapan dengan meminta masukan nama pemain, melainkan memberikan keluaran berupa “Permainan belum dilakukan, silakan melakukan perintah startGame terlebih dahulu!”.

2.8. Command nextTurn

2.8.1. Kegunaan Command

Command nextTurn dipanggil ketika pemain ingin meneruskan gilirannya ke pemain lain. Hal ini karena aksi-aksi seperti investasi, jalankanUnta, dan pasangTrap hanya dapat dipanggil satu kali dalam satu *turn*, dan untuk melanjutkan permainan diperlukan aksi berikutnya.

2.8.2. Skenario-Skenario Penggunaan

- *Command* dipanggil saat permainan sudah dimulai

Command nextTurn digunakan untuk melanjutkan giliran ke pemain berikutnya sesuai urutan permainan yang diinisialisasi pada awal game.

2.9. Command tukarUnta

2.9.1. Kegunaan Command

Command tukarUnta dipanggil ketika pemain ingin menukar posisi dua unta dengan warna acak dengan unta-unta yang setumpuk dalam petak yang sama dapat ditukar posisi tumpukannya.

2.9.2. Skenario-Skenario Penggunaan

- *Command* dipanggil saat permainan sudah dimulai

Command tukarUnta akan melakukan kocokan dadu sebanyak 2 kali, apabila hasil warnanya sama, maka pengocokan dadu diulang

- *Command* dipanggil saat permainan sudah dimulai

Command tukarUnta akan melakukan kocokan dadu sebanyak 2 kali, apabila hasil warnanya berbeda, maka unta warna pertama ditukar posisinya dengan unta warna kedua

BAB 3: Hasil Eksekusi Program

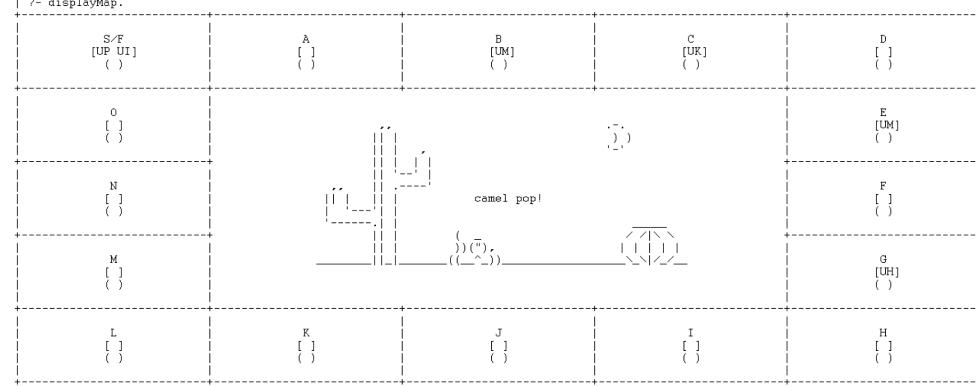
Nama Command	Hasil Eksekusi
startGame	<pre> ?- startGame. Masukan jumlah pemain: 4. Masukkan nama pemain 1: raihan. Nama pemain yang dimasukkan: raihan Masukkan nama pemain 2: alfansya. Nama pemain yang dimasukkan: alfansya Masukkan nama pemain 3: rafen. Nama pemain yang dimasukkan: rafen Masukkan nama pemain 4: grace. Nama pemain yang dimasukkan: grace Urutan pemain: grace - alfansya - rafen - raihan Setiap pemain mendapatkan 4 kartu, 30 poin, dan 1 trap. Kartu grace: merah, kuning, hijau, biru. Kartu alfansya: merah, kuning, hijau, biru. Kartu rafen: merah, kuning, hijau, biru. Kartu raihan: merah, kuning, hijau, biru. Poin grace: 30 Poin alfansya: 30 Poin rafen: 30 Poin raihan: 30 Trap grace: 1 Trap alfansya: 1 Trap rafen: 1 Trap raihan: 1 Mau melakukan apa, nih? (investasi, jalankanUnta, pasangTrap, cekInfo, displayMap)</pre> <p>Untuk memulai permainan, pemain dapat menuliskan <i>command</i> startGame yang menginisialisasi jumlah pemain, nama-nama pemain, urutan pemain, serta banyak kartu, poin, dan trap yang dimiliki oleh masing-masing pemain.</p>
displayMap	<pre> ?- displayMap.</pre> <p>(63 ms) yes</p> <p>Selanjutnya, pemain dapat menuliskan <i>command</i> displayMap untuk melihat peta permainan saat ini. Pada awal permainan, seluruh unta diinisialisasi pada kotak S/F. <i>Command</i> displayMap dapat dijalankan tanpa batas.</p>

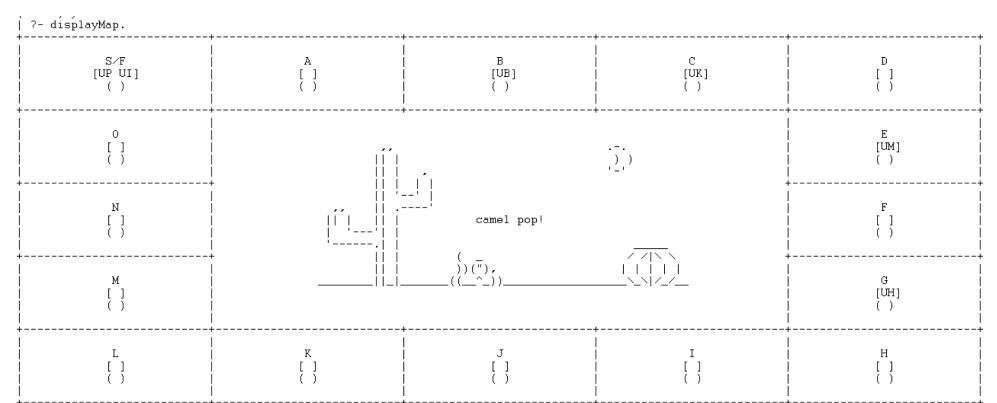
cekInfo	<pre> ?- cekInfo. Masukkan nama pemain: raihan. Nama pemain: raihan Poin: 30 Kartu: [merah,kuning,hijau,biru] Jebakan: 1 yes ?- cekInfo. Masukkan nama pemain: alfansya. Nama pemain: alfansya Poin: 30 Kartu: [merah,kuning,hijau,biru] Jebakan: 1 yes ?- cekInfo. Masukkan nama pemain: rafen. Nama pemain: rafen Poin: 30 Kartu: [merah,kuning,hijau,biru] Jebakan: 1 (16 ms) yes ?- cekInfo. Masukkan nama pemain: grace. Nama pemain: grace Poin: 30 Kartu: [merah,kuning,hijau,biru] Jebakan: 1 yes . </pre> <p>Pemain juga dapat menuliskan <i>command</i> cekInfo untuk mendapatkan informasi mengenai diri sendiri maupun pemain lain terkait poin, banyak kartu, dan banyak jebakan yang dimiliki pemain terkait. Seperti <i>command</i> displayMap, <i>command</i> cekInfo juga dapat dijalankan tanpa batas.</p>
investasi	<pre> ?- investasi. Melakukan investasi... Giliran grace Pilih warna unta untuk investasi (merah, kuning, hijau, biru): merah. Investasi berhasil! grace telah berinvestasi pada unta merah. grace masuk ke daftar investasi unta merah. Papan investasi pada unta merah saat ini: +-----+ INVESTASI UNTA +-----+ 1. grace true ? (16 ms) yes ?- investasi. Anda sudah tidak bisa lagi melakukan aksi tersebut pada giliran ini! (16 ms) yes </pre> <p>Untuk melakukan investasi pada warna unta tertentu, pemain dapat menuliskan <i>command</i> investasi. Kemudian, pemain akan memasukkan <i>input</i> berupa warna unta yang hendak diinvestasikan. Program kemudian akan menampilkan papan investasi warna unta terkait. Dalam satu kali <i>turn</i>, <i>command</i> investasi hanya dapat dijalankan satu kali.</p>

papanInvestasi	<pre> ?- papanInvestasi.</pre> <p>Papan investasi pada unta merah saat ini:</p> <pre>+-----+ INVESTASI UNTA +-----+</pre> <p>Belum ada investasi</p> <p>Papan investasi pada unta kuning saat ini:</p> <pre>+-----+ INVESTASI UNTA +-----+</pre> <p>Belum ada investasi</p> <p>Papan investasi pada unta biru saat ini:</p> <pre>+-----+ INVESTASI UNTA +-----+</pre> <p>Belum ada investasi</p> <p>Papan investasi pada unta hijau saat ini:</p> <pre>+-----+ INVESTASI UNTA +-----+</pre> <p>Belum ada investasi</p> <p>yes</p> <p>Untuk melihat papan investasi seluruh unta, pemain dapat menuliskan <i>command</i> papanInvestasi. Program kemudian akan menampilkan papan investasi dari keempat unta yang dapat diinvestasi, <i>Command</i> papanInvestasi dapat dijalankan tanpa batas.</p>																									
jalankanUnta	<pre>yes ?- aksi_jalankanUnta.</pre> <p>Kondisi peta saat ini:</p> <tr> <td>S/F [UM UK UH UB UP UI] ()</td> <td>A [] ()</td> <td>B [] (Trap)</td> <td>C [] ()</td> <td>D [] ()</td> </tr> <tr> <td>0 [] ()</td> <td></td> <td></td> <td></td> <td>E [] ()</td> </tr> <tr> <td>N [] ()</td> <td></td> <td>camel up!</td> <td></td> <td>F [] ()</td> </tr> <tr> <td>M [] ()</td> <td></td> <td></td> <td></td> <td>G [] (Trap)</td> </tr> <tr> <td>L [] (Trap)</td> <td>K [] ()</td> <td>J [] ()</td> <td>I [] ()</td> <td>H [] ()</td> </tr>	S/F [UM UK UH UB UP UI] ()	A [] ()	B [] (Trap)	C [] ()	D [] ()	0 [] ()				E [] ()	N [] ()		camel up!		F [] ()	M [] ()				G [] (Trap)	L [] (Trap)	K [] ()	J [] ()	I [] ()	H [] ()
S/F [UM UK UH UB UP UI] ()	A [] ()	B [] (Trap)	C [] ()	D [] ()																						
0 [] ()				E [] ()																						
N [] ()		camel up!		F [] ()																						
M [] ()				G [] (Trap)																						
L [] (Trap)	K [] ()	J [] ()	I [] ()	H [] ()																						

	<p>Dadu dikocok.. Dadu Angka : 5 Dadu Warna : kuning</p> <p>Unta kuning bergerak maju sebanyak 5 langkah...</p> <p>Kondisi peta saat ini:</p>
	<p>Tidak ada unta yang mengenai Trap.</p> <p>Perintah jalankan Unta mengocok dadu dari 5 pilihan warna secara acak (merah, kuning, hijau, biru, dan abu-abu) dan 6 langkah yang dapat diambil. Setelah didapat kombinasi jumlah langkah dan warna unta, unta dengan warna tersebut akan melangkah maju sejauh jumlah langkah, kecuali unta putih dan hitam yang bergerak mundur berdasarkan paritas dari dadu berwarna abu-abu.</p> <p>Unta hitam bergerak mundur sebanyak 3 langkah...</p> <p>Kondisi peta saat ini:</p>
	<p>Ada unta mengenai Trap!</p> <p>Kondisi peta saat ini:</p>

	<p>Perintah jalankanUnta juga melakukan perubahan pada unta yang terkena <i>trap</i> setelah bergerak. Unta yang terkena <i>trap</i> akan berpindah petak sesuai dengan arah <i>trap</i> dan <i>trap</i> pada petak tersebut akan hilang.</p>
pasangTrap	<pre> ?- pasangTrap. Memasang trap... Kondisi peta saat ini:</pre> <pre>Masukkan kode petak : 'A'. Masukkan jenis trap (maju/mundur) : maju. Trap berhasil dipasang. Unta yang mendarat pada petak tersebut akan maju 1 petak.grace akan mendapatkan 10 poin ketika ada unta yang mendarat di petak tersebut. Sisa trap grace : 0.</pre> <pre> ?- pasangTrap. Memasang trap... Kondisi peta saat ini:</pre> <pre>Masukkan kode petak : 'B'. Masukkan jenis trap (maju/mundur) : maju. Trap gagal dipasang. Pastikan tidak ada 2 trap di petak yang bertetangga. Sisa trap raihan : 1.</pre>

	Jebakan tidak dapat ditaruh bersebelahan dengan jebakan yang sudah ada, sehingga apabila pemain berusaha untuk meletakkan jebakan di sebelah jebakan lain, akan dikeluarkan <i>output</i> “Trap gagal dipasang”.
nextTurn	<pre> ?- nextTurn. Sekarang giliran pemain b. Mau melakukan apa, nih? (investasi, jalankanUnta, pasangTrap, cekInfo, displayMap)</pre> <p>Kemudian, untuk berpindah ke pemain berikutnya, pemain dapat menuliskan <i>command</i> nextTurn.</p>
tukarUnta	 <p>Tampilan awal unta pada peta sebelum dilakukan tukarUnta, UM berada pada petak B dan UB berada pada petak G.</p> <pre> ?- tukarUnta. Menukar unta... Kocok dadu... Warna unta 1: merah Warna unta 2: biru Posisi sebelum unta merah: B Posisi sebelum unta biru: G Unta merah dan biru bertukar posisi... Posisi sesudah unta merah: G Posisi sesudah unta biru: B (16 ms) yes</pre> <p><i>Command</i> tukarUnta memanggil fungsi kocok dadu untuk mendapatkan dua warna unta. Apabila warnanya sama, dilakukan pemanggilan kembali fungsi kocok dadu. Apabila warnanya berbeda, dilakukan pertukaran posisi unta warna pertama dengan unta kedua pada peta.</p>



Tampilan awal unta pada peta setelah dilakukan tukarUnta, UM berada pada petak G dan UB berada pada petak B (keduanya bertukar posisi).

Lampiran

NIM	Persentase Kerja
13523021	25%
13523005	25%
13523031	25%
13523087	25%

Referensi

Tim Asisten Lab Intelektual Buatan 2022. (2024). Spesifikasi Tugas Besar IF1221 Logika Komputasional Tahun 2024/2025 Camel POP! https://docs.google.com/document/d/1LMeviChuAEhfG-VOgZT-GbGPqEnb8mmYrlIB5zLzj3c/edit?tab=t_0

Informatics Engineering Study Program School of Electrical Engineering and Informatics ITB, *Introduction to Prolog.* PowerPoint Presentation, 2024. https://cdn-edunex.itb.ac.id/storages/files/1698024074255_MateriMinggu10_23_Logika_Introduction_to_Prolog.pdf

Informatics Engineering Study Program School of Electrical Engineering and Informatics ITB, *Prolog (Rekurens, Cut, Fail, List).* PowerPoint Presentation, 2024. https://cdn-edunex.itb.ac.id/storages/files/1698024096437_MateriMinggu10_23_Logika_Prolog_Rekurens_Cut_Fail.pdf

Informatics Engineering Study Program School of Electrical Engineering and Informatics ITB, *Prolog (List, File Eksternal, Loop).* PowerPoint Presentation, 2024. https://cdn-edunex.itb.ac.id/54386-Computational-Logic-Parallel-Class/175916-Minggu-11/77094-Prolog-List-File-Loop/1698387519098_MateriMinggu11_23_Prolog_List_File_Eksternal_Loop.pdf