IF2121 LOGIKA KOMPUTASIONAL

GLOBAL CONQUEST: BATTLE FOR SUPREMACY

Laporan Tugas Besar 1

Disusun untuk memenuhi tugas mata kuliah Logika Komputasional
pada Semester 1 (satu) Tahun Akademik 2023/2024
Tugas Besar IF2121 Logika Komputasional

Oleh

Shabrina Maharani	**13522134**
Auralea Alvinia Syaikha	**13522148**
Muhammad Roihan	**13522152**
Muhammad Rasheed Qais Tandjung	**13522158**

Kelompok santaidulugasih:-\+hMin2.

PROGRAM STUDI TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
BANDUNG
2023

DAFTAR ISI

DAFTAR ISI **2**
BAB 1 DESKRIPSI MASALAH **4**
BAB 2 PROGRAM **5**
2.1 Command Program 5
2.1.1 List (list.pl) 5
1. min 5
2. max 5
3. range 5
4. count 5
5. sum 6
6. getIndex 6
7. getElement 6
8. swap 6
9. slice 6
10. sortList 7
11. shift 7
12. remove 7
13. subset 7
14. writeList 8
15. randomList 8
2.1.2 Inisialisasi (init.pl) 8
1. start 8
2. getPlayerCount 8
3. getPlayerList 8
4. rollPlayerOrder 8
5. writePlayerOrder 9
6. initPlayerTroops 9
7. takeLocation 9
8. placeTroops 9
9. placeAutomatic 9
10. checkFinish 9
2.1.3 Wilayah (wilayah.pl) 9
1. checkLocationDetail 9
2. getNeighborNames 10
2.1.4 Player (player.pl) 10
2.1.5 Turn 11
2.1.5.1 endturn (turn.pl) 11
1. endTurn 11
2. checkDominatedTerritory 12
2.1.5.2 attack.pl (attack.pl) 12
1. attack 12
2. getNeighbors 13

3. removeNonEnemyNeighbor 13
4. rollDiceAttack 13
5. writeListAttack 13
2.1.5.3 risk (risk.pl) 14
1. risk 14
2. rebel 14
3. riskC 15
4. runRisk 15
2.1.5.4 move (move.pl) 16
1. move 16
2.1.5.5 draft (draft.pl) 16
1. draft 16
2.1.6 Map (map.pl) 16
1. slowFormatMap 16
2. displayMap 17
3. printPlayerTerritories 17
2.1.7 Bonus 17
2.1.7.1 reinforcement (reinforcement.pl) 17
1. rollReinforcement 17
2. updateReinforcementCount 18
3. ambilReinforcement 18
2.1.7.2 win_probabilities (win_probabilities.pl) 18
1. factorial 18
2. permute 18
3. choose 18
4. exp 18
5. sigma dan sigma3 18
6. winProbability 19
2.1.7.3 world_leaders (world_leaders.pl) 19
1. isDominating 19
2. checkLose 19
2.1.7.4 cheat (risk.pl) 19
1. riskC 19
BAB 3 HASIL EKSEKUSI **20**
BAB 4 LAMPIRAN **29**
4.1 Pembagian dan Persentase Kerja 29
4.2 Link Repository Github 29
REFERENSI **30**

BAB 1 DESKRIPSI MASALAH

Dalam Tugas Besar Logika Komputasional, semua mahasiswa yang terdaftar di IF2121 diminta untuk membuat game strategi sebagai seorang programmer yang ingin menguasai dunia menggunakan bahasa pemrograman deklaratif Prolog (GNU Prolog). Tugas besar ini mengharuskan implementasi konsep rekurens, list, cut, fail, dan loop yang telah dipelajari dalam perkuliahan Logika Komputasional IF2121, pra-praktikum, dan eksplorasi mandiri mengenai Logika Komputasional dan Prolog.

Alur permainan dimulai dengan inisiasi jumlah pemain (2-4 pemain) dan penginputan nama pemain. Dunia permainan terdiri dari 24 wilayah di 6 benua, di mana setiap pemain memulai dengan jumlah tentara tertentu. Pemain dapat menduduki wilayah dengan mendistribusikan tentara secara manual atau otomatis.

Setiap giliran, pemain mendapatkan tentara tambahan berdasarkan jumlah wilayah yang dimilikinya. Jika pemain menguasai satu benua, jumlah tentara tambahan bertambah. Pemain dapat melakukan perintah seperti Draft (menempatkan tentara tambahan), Move (memindahkan tentara), Risk (mencoba mendapatkan risk card), Attack (menyerang wilayah tetangga), dan EndTurn (mengakhiri giliran).

Kriteria kemenangan adalah menguasai semua wilayah di papan dan mengeliminasi semua lawan. Ini menciptakan dinamika persaingan yang memerlukan strategi, keberanian, dan kecerdikan pemain untuk mencapai tujuan dominasi dunia.

BAB 2 PROGRAM

2.1 Command Program

Berikut adalah daftar command program yang telah kami buat disertai penjelasan serta skenario mengenai penggunaan masing-masing command.

2.1.1 List (list.pl)

Dalam file list.pl terdapat beberapa command sebagai berikut.

1. **min**

   Command “min” digunakan untuk menemukan nilai minimum dalam sebuah list. Fungsi min menerima dua argumen, yaitu list yang ingin diinspeksi dan variabel untuk menyimpan nilai minimum yang ditemukan. Pada dasarnya, predikat ini melakukan rekursi melalui elemen-elemen list dan memperbarui nilai minimum ketika menemukan elemen yang lebih kecil. Contoh penggunaannya adalah min([3, 8, 5, 2, 10], Min).

2. **max**

   Command “max” digunakan untuk mencari nilai maksimum (max) dari sebuah daftar (list). Fungsi max menerima dua argumen, yaitu sebuah daftar (List) dan variabel Max yang akan menyimpan nilai maksimum dari daftar tersebut. Dalam skenario penggunaan, pengguna dapat memanggil fungsi max dengan memberikan daftar angka sebagai argumen. Contoh penggunaannya adalah max([3, 8, 5, 2, 10], Max).

3. **range**

   Command “range” berfungsi untuk menghitung jangkauan (range) dari sebuah daftar (list) bilangan. Dalam aturan ini, digunakan dua fungsi bantuan, yaitu max dan min, untuk mencari nilai maksimum dan minimum dari daftar tersebut. Selanjutnya, nilai range dihitung dengan mengurangkan nilai minimum dari nilai maksimum. Contoh penggunaannya adalah range([10, 5, 8, 15, 3], Range).

4. **count**

   Command “count” digunakan untuk menghitung jumlah elemen dalam sebuah daftar (list). Aturan ini terdiri dari dua klausa. Klausa pertama menyatakan bahwa jumlah elemen dalam daftar kosong adalah 0. Klausa kedua menggunakan rekursi untuk menghitung jumlah elemen dalam daftar yang tidak kosong. Saat merespon permintaan penghitungan, Prolog akan memanggil fungsi count dengan melewatkan daftar sebagai argumen. Dalam setiap langkah rekursif, elemen pertama daftar diabaikan dan proses diulang pada sisa daftar hingga mencapai kondisi dasar. Contoh penggunaannya adalah count([5, 2, 8, 7, 3], Count).

5. **sum**

   Command “sum” digunakan untuk menghitung jumlah (sum) dari semua elemen dalam sebuah daftar (list). Aturan ini terdiri dari dua klausa. Klausa pertama menyatakan bahwa jumlah elemen dalam daftar kosong adalah 0. Klausa kedua menggunakan rekursi untuk menghitung jumlah elemen dalam daftar yang tidak kosong. Saat merespon permintaan penghitungan, Prolog akan memanggil fungsi sum dengan melewatkan daftar sebagai argumen. Dalam setiap langkah rekursif, elemen pertama daftar diambil dan ditambahkan dengan jumlah dari sisa daftar hingga mencapai kondisi dasar. Contoh penggunaannya adalah sum([3, 7, 1, 4, 2], Total).

6. **getIndex**

   Command “getIndex” digunakan untuk mencari indeks dari elemen pertama dalam sebuah daftar (list). Aturan ini terdiri dari dua klausa. Klausa pertama menyatakan bahwa jika elemen yang dicari adalah elemen pertama dari daftar, maka indeksnya adalah 1. Klausa kedua menggunakan rekursi untuk mencari indeks elemen dalam sisa daftar yang tidak mengandung elemen tersebut. Contoh penggunaannya adalah getIndex([5, 8, 2, 7, 3], 2, Index).

7. **getElement**

   Command “getElement” digunakan untuk mendapatkan elemen pada indeks tertentu dalam sebuah daftar (list). Aturan ini terdiri dari dua klausa. Klausa pertama menyatakan bahwa jika indeks yang dicari adalah 1, maka elemen yang dihasilkan adalah elemen pertama dari daftar. Klausa kedua menggunakan rekursi untuk mencari elemen pada indeks tertentu dalam sisa daftar. Contoh penggunaannya adalah getElement([5, 8, 2, 7, 3], 3, E).

8. **swap**

   Command “swap” digunakan untuk menukar dua elemen dalam sebuah daftar (list) berdasarkan indeks yang diberikan. Aturan ini terdiri dari beberapa klausa. Pertama, klausa appendList digunakan untuk menggabungkan dua daftar. Klausa swap utama menggunakan fungsi getElement untuk mendapatkan nilai elemen pada indeks yang diberikan. Fungsi rekursif swapR kemudian berperan dalam melakukan pertukaran elemen dan membentuk daftar baru dengan elemen yang sudah ditukar. Contoh penggunaannya adalah swap([5, 8, 2, 7, 3], 2, 4, Result).

9. **slice**

   Command “slice” digunakan untuk mendapatkan potongan (slice) dari sebuah daftar (list) berdasarkan dua indeks yang diberikan, yaitu indeks awal (I1) dan indeks akhir (I2). Aturan ini terdiri dari dua klausa.

Klausa indexInRange digunakan untuk memeriksa apakah indeks berada dalam rentang yang ditentukan. Klausa utama slice menggunakan fungsi rekursif sliceR untuk memproses setiap elemen dalam daftar dan membangun daftar hasil (slice) sesuai dengan rentang indeks yang diberikan. Contoh penggunaannya adalah slice([1, 2, 3, 4, 5, 6], 2, 5, Result).

10. **sortList**

    Command "sortList" digunakan untuk mengurutkan sebuah daftar (list) menggunakan metode Selection Sort. Aturan ini terdiri dari dua klausa. Klausa pertama menyatakan bahwa mengurutkan daftar kosong akan menghasilkan daftar kosong. Klausa kedua merupakan implementasi algoritma Selection Sort menggunakan fungsi-fungsi bantuan seperti min, getIndex, dan swap untuk mencari nilai minimum, indeks minimum, dan melakukan pertukaran elemen. Contoh penggunaannya adalah sortList([5, 2, 8, 1, 6], Result).

11. **shift**

    Command "shift" digunakan untuk melakukan operasi pergeseran sirkular pada sebuah daftar (list). Operasi ini menggeser elemen-elemen daftar sehingga elemen yang sebelumnya berada di posisi akhir akan menjadi elemen pertama setelah pergeseran. Aturan ini memanfaatkan fungsi slice, count, dan appendList untuk memisahkan dan menggabungkan bagian-bagian daftar yang terlibat dalam pergeseran. Contoh penggunaannya adalah shift([1, 2, 3, 4, 5], 2, Result).

12. **remove**

    Command "remove" digunakan untuk menghapus elemen dengan nilai tertentu dari sebuah daftar (list). Aturan ini menggunakan fungsi-fungsi bantuan seperti getIndex, slice, count, dan appendList untuk memisahkan elemen sebelum dan sesudah elemen yang akan dihapus, lalu menggabungkan kembali sisa-sisa daftar tersebut. Contoh penggunaannya adalah remove([1, 3, 5, 2, 4], 5, Result).

13. **subset**

    Command "subset" digunakan untuk memeriksa apakah suatu daftar (list) merupakan subdaftar (subset) dari daftar lainnya. Aturan ini terdiri dari dua klausa. Klausa pertama menyatakan bahwa suatu daftar selalu merupakan subdaftar dari daftar lainnya jika daftar tersebut kosong. Klausa kedua menggunakan rekursi untuk memeriksa setiap elemen dalam subdaftar, yaitu apakah setiap elemen tersebut terdapat dalam daftar utama. Contoh penggunaannya adalah subset([1, 2, 3, 4, 5], [2, 4]).

14. **writeList**

    Command “writeList” digunakan untuk menuliskan konten dari sebuah daftar (list) ke konsol. Aturan ini terdiri dari dua klausa. Klausa pertama menyatakan bahwa menuliskan daftar kosong tidak menghasilkan output apapun. Klausa kedua, yang merupakan implementasi rekursif writeListR, digunakan untuk menuliskan setiap elemen daftar beserta nomor urutnya ke konsol dengan menggunakan format tertentu. Contoh penggunaannya adalah writeList([apple, banana, orange]).

15. **randomList**

    Command randomList digunakan untuk mendapatkan elemen acak dari sebuah daftar (list). Aturan ini terdiri dari dua klausa. Klausa pertama menghitung jumlah elemen dalam daftar menggunakan fungsi count. Klausa kedua menggunakan fungsi bawaan random untuk menghasilkan nomor acak antara 1 dan jumlah elemen daftar, lalu menggunakan fungsi getElement untuk mendapatkan elemen yang bersesuaian dengan nomor acak tersebut. Contoh penggunaanya adalah randomList([10, 20, 30, 40, 50], Result).

2.1.2 Inisialisasi (init.pl)

Dalam file init.pl terdapat beberapa command sebagai berikut.

1. **start**

   Setelah melakukan *console* prolog pada “main.pl”, pengguna diharuskan untuk mengetik command “start.” agar program permainan dapat berjalan. Contoh penggunaannya adalah start.

2. **getPlayerCount**

   Command “getPlayerCount” digunakan untuk meminta input jumlah pemain yang hanya dapat dimainkan oleh 2 pemain hingga 4 pemain. Command ini harus dilakukan setelah program mulai berjalan. Contoh penggunaannya adalah getPlayerCount(N),

3. **getPlayerList**

   Command “getPlayerList” digunakan untuk meminta input nama pemain yang hanya dapat dimainkan oleh 2 pemain hingga 4 pemain secara bergiliran. Command ini dapat dilakukan setelah jumlah pemain diinisialisasi. Contoh penggunaannya adalah getPlayerList(N, PlayerList).

4. **rollPlayerOrder**

   Command “rollPlayerOrder” digunakan untuk menggambarkan proses pelemparan dadu untuk menentukan urutan pemain dalam permainan. Command ini dapat dilakukan setelah jumlah dan nama pemain diinisialisasi. Contoh penggunaannya adalah rollPlayerOrder(PlayerList, [], Order).

5. **writePlayerOrder**

   Command “writePlayerOrder” digunakan untuk menampilkan list pemain yang sesuai dengan urutan pemain setelah dilakukan pelemparan dadu menggunakan command writePlayerOrder dalam permainan. Contoh penggunaannya adalah writePlayerOrder(TailNPL).

6. **initPlayerTroops**

   Command “initPlayerTroops” digunakan untuk menginisialisasi data tentara ke database program. Command ini wajib dilakukan sebelum adanya pembagian tentara untuk masing-masing wilayah. Contoh penggunaannya adalah initPlayerTroops(PList, NTroops).

7. **takeLocation**

   Command “takeLocation” digunakan untuk pemain mengambil wilayah sesuai dengan urutan yang telah dilakukan. Command ini harus dilakukan dengan kondisi urutan pemain sudah ditetapkan. Contoh penggunaannya adalah takeLocation(af3).

8. **placeTroops**

   Command “placeTroops” digunakan untuk menempatkan tentara tambahan pada suatu wilayah. Di dalam program, command ini memiliki parameter *Terr* yang merupakan wilayah tempat tentara akan ditempatkan. Contoh penggunaannya adalah placeTroops(af3,2).

9. **placeAutomatic**

   Command “placeAutomatic” digunakan untuk mendistribusikan tentara secara otomatis ke wilayah-wilayah pemain yang belum mendapat tentara pada awal permainan. Fungsi ini bekerja berdasarkan aturan tertentu dan melibatkan beberapa proses secara rekursif. Contoh penggunaannya adalah placeAutomatic.

10. **checkFinish**

    Command “checkFinish” digunakan untuk memeriksa apakah distribusi tentara pada papan permainan telah selesai atau belum. Kode ini mengambil informasi tentang pemain saat ini (currentPlayer(I)) dan total jumlah pemain (playerCount(N)), lalu memeriksa apakah pemain saat ini sudah mencapai pemain terakhir. Jika pemain saat ini sudah mencapai pemain terakhir, maka distribusi tentara dianggap selesai. Contoh penggunaanya adalah checkFInish.

2.1.3 Wilayah (wilayah.pl)

Dalam file wilayah.pl terdapat beberapa command sebagai berikut.

1. **checkLocationDetail**

   Command checkLocationDetail digunakan untuk memeriksa dan menampilkan detail dari suatu wilayah dalam konteks permainan strategi. Fungsi ini mengambil sebuah lokasi (wilayah) sebagai argumen dan

berusaha mendapatkan informasi tentang wilayah tersebut. Jika wilayah tersebut ditemukan dalam basis pengetahuan (fakta-fakta wilayah), maka detail wilayah seperti kode, nama, pemilik, jumlah tentara, dan tetangga-tetangganya akan ditampilkan secara berurutan dengan menggunakan fungsi slowFormat untuk memberikan efek tampilan bertahap. Contoh penggunaannya adalah checkLocationDetail(na1).

2. **getNeighborNames**

   Command getNeighborNames digunakan untuk mendapatkan nama-nama tetangga dari sebuah daftar wilayah (regions). Aturan ini mengambil daftar wilayah sebagai argumen dan menggunakan rekursi dalam implementasi getNeighborNamesR untuk mendapatkan nama-nama tetangga dari setiap wilayah dalam daftar. Contoh getNeighborNames(‘au1’,Result).

2.1.4 Player (player.pl)

Dalam file player.pl terdapat beberapa command sebagai berikut.

1. checkPlayerDetail

   Command "checkPlayerDetail" digunakan untuk menampilkan detail informasi pemain, termasuk nama pemain, benua yang dimilikinya, total wilayah yang dimilikinya, total tentara aktif, dan total tentara tambahan. Contoh penggunaannya adalah checkPlayerDetail(player1). alam contoh ini, checkPlayerDetail akan menampilkan detail informasi dari player1, termasuk nama pemain, benua yang dimilikinya, total wilayah yang dimilikinya, total tentara aktif, dan total tentara tambahan.

2. checkPlayerTerritories

   Command "checkPlayerTerritories" digunakan untuk mengecek dan menampilkan detail wilayah yang dikuasai oleh seorang pemain, termasuk informasi tentang nama wilayah, jumlah tentara di setiap wilayah, dan benua tempat wilayah tersebut berada. Fungsionalitas ini memberikan gambaran yang lebih jelas tentang keadaan pemain dalam permainan Risiko, dengan fokus pada wilayah yang dikuasainya. Contoh penggunaannya adalah checkPlayerTerritories(player1). Dalam contoh ini, checkPlayerTerritories akan menampilkan informasi detail tentang wilayah yang dikuasai oleh player1, termasuk nama wilayah, jumlah tentara di setiap wilayah, dan benua tempat wilayah tersebut berada. Informasi ini dapat membantu pemain dalam merencanakan strategi dan mengambil keputusan selama permainan Risiko.

3. checkIncomingTroops

   Command "checkIncomingTroops" digunakan untuk mengecek jumlah tentara tambahan yang akan diterima oleh seorang pemain pada giliran selanjutnya. Predikat ini memberikan gambaran tentang berapa

banyak tentara tambahan yang akan dimiliki oleh pemain, termasuk jumlah tentara tambahan dari wilayah-wilayah yang dimilikinya, bonus benua yang dimilikinya, dan tentara tambahan lainnya yang mungkin diperoleh. Contoh penggunaannya adalah checkIncomingTroops(player1). Dalam contoh ini, checkIncomingTroops akan menampilkan informasi tentang jumlah tentara tambahan yang akan diterima oleh player1 pada giliran selanjutnya. Informasi ini dapat membantu pemain dalam merencanakan strategi dan pengambilan keputusan untuk giliran selanjutnya dalam permainan Risiko. Selain itu, predikat displayHighlightedMap akan menyoroti (highlight) wilayah-wilayah yang dimiliki oleh player1 pada peta permainan.

2.1.5 Turn

2.1.5.1 endturn (turn.pl)

1. **endTurn**

   Command "endTurn" berfungsi untuk mengakhiri giliran seorang pemain dalam permainan. Fungsi endTurn dimulai dengan memastikan bahwa permainan masih berlangsung (isPlaying). Kemudian, ia mengidentifikasi pemain yang sedang aktif (player(P)) dan menampilkan pesan yang menyatakan bahwa pemain tersebut mengakhiri gilirannya. Setelah itu, fungsi endTurnCont dipanggil. Fungsi endTurnCont bertanggung jawab untuk melakukan beberapa tugas terkait pergantian giliran. Pertama, ia melakukan perpindahan ke pemain berikutnya dengan memanggil nextPlayer. Jika pemain yang terakhir mengakhiri gilirannya, fungsi ini memberikan informasi bahwa satu ronde permainan sudah berakhir dan ronde berikutnya dimulai. Selanjutnya, fungsi rollReinforcement dipanggil untuk menentukan jumlah tentara tambahan yang akan diberikan kepada pemain pada awal gilirannya. Selanjutnya, fungsi ini mengupdate beberapa variabel boolean seperti riskAvailable dan attackAvailable. Selain itu, efek RISK tertentu seperti ceasefire, superSoldier, dan diseaseOutbreak dihapus dengan menggunakan retractall. Fungsi juga menangani efek RISK khusus, yaitu supplyChainIssue. Jika pemain terkena efek ini, maka pemain tidak mendapatkan tentara tambahan pada giliran tersebut. Informasi efek ini ditampilkan, dan pemain diberikan opsi untuk melihat peta sebelum mengakhiri giliran. Selanjutnya, fungsi menghitung jumlah tentara tambahan yang akan diberikan kepada pemain. Jumlah ini terdiri dari tambahan tentara berdasarkan wilayah yang dimiliki pemain (NewTroops) dan tambahan berdasarkan wilayah yang dikuasai oleh pemain di setiap benua (BonusTroops). Total tambahan tentara kemudian dihitung dan ditampilkan. Fungsi endTurn juga menangani efek RISK auxiliary, yang

menggandakan jumlah tentara tambahan yang diberikan kepada pemain pada giliran tersebut. Akhirnya, peta permainan diperbarui dengan menampilkan wilayah-wilayah yang dimiliki oleh pemain, dan status permainan (isPlaying) diperbarui untuk menunjukkan bahwa permainan masih berlangsung. Sebagai hasilnya, pemain berikutnya dapat melanjutkan gilirannya. Contoh penggunaannya adalah endTurn.

2. **checkDominatedTerritory**

   Command “checkDominatedTerritory” dalam implementasi tersebut digunakan untuk menghitung bonus tentara tambahan yang akan diberikan kepada seorang pemain berdasarkan wilayah yang telah dikuasainya. Fungsi ini menerima dua daftar, yaitu daftar wilayah yang mewakili enam benua (North America, South America, Europe, Africa, Asia, Australia) dan daftar bonus tentara yang sesuai dengan setiap benua. Fungsi kemudian memeriksa wilayah-wilayah yang dimiliki oleh seorang pemain, dan jika pemain berhasil mendominasi seluruh wilayah di suatu benua, bonus tentara dari benua tersebut akan ditambahkan ke total bonus. Contoh penggunaannya adalah checkDominatedTerritory(BonusTroops).

2.1.5.2 attack.pl (attack.pl)

1. **attack**

   Command "attack" digunakan untuk memulai serangan antar wilayah dalam permainan. Ketika pemain memanggil attack, pertama-tama akan diperiksa apakah pemain dapat menyerang pada giliran tersebut melalui aturan attackAvailable. Jika pemain sudah menyerang pada giliran tersebut, pesan akan ditampilkan bahwa pemain sudah menyerang dan diminta untuk menunggu giliran berikutnya. Jika pemain belum menyerang pada giliran tersebut, pemain dapat memulai serangan. Selanjutnya, pemain diminta untuk memilih wilayah yang akan digunakan sebagai basis penyerangan. Pemilihan wilayah ini melibatkan validasi untuk memastikan bahwa pemain memilih wilayah yang dimilikinya. Setelah pemilihan wilayah, pemain diminta untuk memasukkan jumlah tentara yang akan digunakan dalam serangan, dengan validasi agar jumlah tentara yang dimasukkan tidak melebihi jumlah tentara yang dimiliki di wilayah tersebut. Kemudian, pemain diberikan daftar wilayah tetangga yang dapat diserang, dan pemain diminta untuk memilih wilayah yang akan diserang. Setelah pemilihan wilayah, pertempuran dimulai dengan lemparan dadu yang dihasilkan secara acak. Hasil pertempuran akan mempengaruhi kepemilikan wilayah. Jika pemain menang dalam pertempuran, wilayah yang diserang akan dikuasai oleh pemain, dan pemain dapat mengirimkan sebagian dari tentaranya untuk mengisi

wilayah tersebut. Jika pemain kalah, tentaranya akan berkurang, dan tidak ada perubahan kepemilikan wilayah. Contoh penggunaanya adalah attack.

2. **getNeighbors**

   Command "getNeighbors" memiliki tujuan untuk mendapatkan daftar wilayah tetangga dari suatu wilayah tertentu dalam permainan. Fungsi ini membutuhkan dua parameter, yaitu Terr (wilayah yang ingin dicari tetangganya) dan Res (hasil berupa daftar wilayah tetangga). Contoh penggunaannya adalah getNeighbors('na1', Result).

3. **removeNonEnemyNeighbor**

   Command "removeNonEnemyNeighbor" memiliki tujuan untuk menghapus wilayah tetangga yang dimiliki oleh pemain saat ini dari daftar wilayah tetangga. Fungsi ini membutuhkan tiga parameter: daftar wilayah tetangga awal ([Head | Tail]), daftar sementara (CurrRes), dan hasil akhir (Res).Hasil Res adalah daftar wilayah tetangga yang tidak dimiliki oleh pemain saat ini atau tidak sedang dalam kondisi gencatan senjata. Fungsi ini dapat digunakan, misalnya, ketika pemain hendak menentukan wilayah mana yang dapat diserang selama giliran permainan. Maka, pemain dapat menghapus wilayah yang dimilikinya sendiri atau sedang dalam kondisi gencatan senjata dari pilihan serangan. Contoh penggunaannya adalah removeNonEnemyNeighbor([Head | Tail], CurrRes, Res).

4. **rollDiceAttack**

   Command "rollDiceAttack" digunakan untuk melempar dadu selama fase penyerangan dalam permainan. Fungsi ini membutuhkan tiga parameter: jumlah dadu yang akan dilempar (N), hasil akhir dari jumlah yang keluar pada dadu-dadu tersebut (Res), dan pemain yang sedang menjalankan serangan (Player). Fungsi ini memanggil predikat rollDiceAttackR yang melakukan rekursi sebanyak N kali untuk melakukan pelemparan dadu sebanyak jumlah yang diinginkan. Pada setiap lemparan, hasil dadu ditambahkan ke dalam CurrSum, yang kemudian diakumulasi hingga selesai melakukan semua lemparan dadu. Hasil akhir dari lemparan dadu dicetak ke layar, memberikan pemain informasi tentang nilai setiap dadu yang dilempar. Fungsi ini berguna untuk menentukan hasil dari serangan, di mana jumlah dadu yang keluar akan mempengaruhi hasil pertempuran antara wilayah yang menyerang dan wilayah yang diserang. Contoh penggunaannya adalah rollDiceAttack(N, Res, Player).

5. **writeListAttack**

   Command "writeListAttack" digunakan untuk menampilkan daftar wilayah yang dapat diserang, serta probabilitas kemenangan pada setiap serangan. Fungsi ini mengambil dua parameter: List, yang merupakan

daftar wilayah yang dapat diserang, dan NTroops, yang merupakan jumlah tentara yang akan digunakan dalam serangan. Fungsi ini memanggil predikat writeListAttackR yang melakukan rekursi untuk setiap wilayah dalam List. Pada setiap iterasi, fungsi ini mendapatkan informasi tentang jumlah tentara di wilayah yang akan diserang (TroopsAttacked) dan menghitung probabilitas kemenangan menggunakan predikat winProbability. Hasilnya kemudian ditampilkan ke layar dengan format yang mencakup nomor wilayah, nama wilayah, dan persentase kemenangan. Skenario penggunaan dapat ditemui ketika pemain memutuskan untuk menyerang wilayah musuh dan ingin melihat daftar wilayah yang dapat diserang berserta probabilitas kemenangan pada setiap serangan. Informasi ini membantu pemain dalam mengambil keputusan strategis dalam memilih wilayah yang akan diserang. Contoh penggunaannya adalah writeListAttack(List, NTroops).

2.1.5.3 risk (risk.pl)

1. **risk**

   Command “risk” digunakan untuk memainkan kartu RISK pada permainan. RISK adalah mekanisme dalam permainan yang memberikan efek atau keuntungan tertentu kepada pemain yang memainkannya. Pada setiap pemanggilan risk, sistem secara acak memilih jenis kartu RISK yang akan dimainkan antara Ceasefire Order (CFO), Super Soldier Serum (SSO), Auxiliary Troops (AUX), Rebellion (REB), Disease Outbreak (DSOB), atau Supply Chain Issue (SCI). Setelah jenis kartu RISK dipilih, fungsi runRisk dijalankan untuk memberikan efek atau keuntungan sesuai dengan jenis kartu yang dipilih. Efek tersebut dapat mencakup gencatan senjata (ceasefire), peningkatan kekuatan tentara (super soldier), tentara tambahan (auxiliary), pemberontakan wilayah (rebellion), wabah penyakit (disease outbreak), atau masalah rantai pasokan (supply chain issue). Skenario penggunaan dapat terjadi ketika pemain ingin memanfaatkan kartu RISK yang dimilikinya pada giliran tertentu. Kartu RISK memberikan dimensi strategis tambahan dalam permainan, memungkinkan pemain untuk mengubah dinamika permainan sesuai dengan efek yang dimiliki oleh kartu RISK tersebut. Contoh penggunaannya adalah risk.

2. **rebel**

   Command “rebel” digunakan untuk melaksanakan pemberontakan (rebellion) di dalam permainan. Pemberontakan ini terjadi secara acak, memilih pemain lain secara random dan mengambil alih salah satu wilayah yang dimiliki oleh pemain yang memanggil perintah ini. Dalam

skenario penggunaan, pemanggilan rebel menciptakan elemen kejutan dan ketidakpastian dalam permainan, karena wilayah yang diambil alih oleh pemain lain dipilih secara acak. Proses rebel dimulai dengan pemilihan pemain lain secara acak, dan kemudian wilayah yang dimiliki oleh pemanggil perintah rebel. Tentara di wilayah tersebut kemudian diserahkan kepada pemain yang terpilih secara acak. Efek dari pemberontakan ini dapat merubah keseimbangan kekuatan antar pemain dan memicu strategi baru dalam permainan. Contoh skenario penggunaan rebel adalah ketika seorang pemain ingin menciptakan ketidakpastian dalam permainan dengan merampas wilayah lawan secara acak. Pemanggilan rebel secara efektif dapat mengubah tata letak kekuatan dan menghadirkan tantangan baru bagi pemain yang kehilangan wilayahnya. Contoh penggunaannya adalah rebel.

3. **riskC**

   Command “riskC” digunakan untuk memainkan kartu RISK dengan memilih jenis kartu RISK yang diinginkan. Kartu RISK memberikan efek khusus yang dapat mempengaruhi dinamika permainan. Pada skenario penggunaannya, pemain memilih jenis kartu RISK yang ingin mereka dapatkan dari sebuah daftar pilihan yang tersedia. Pada implementasinya, setelah memastikan bahwa pemain masih dalam kondisi bermain dan kartu RISK tersedia, pemanggilan riskC memberikan opsi pemilihan jenis kartu RISK kepada pemain. Pemilihan dilakukan dengan memasukkan nomor yang sesuai dengan jenis kartu yang diinginkan. Setelah pemain memilih, efek dari kartu RISK yang dipilih tersebut akan dijalankan dalam permainan. Contoh penggunaannya adalah riskC.

4. **runRisk**

   Command “runRisk” digunakan untuk menjalankan efek dari kartu RISK yang diperoleh oleh seorang pemain. Ketika seorang pemain memperoleh kartu RISK, runRisk dipanggil untuk memberikan efek yang sesuai dengan jenis kartu RISK yang diperoleh. Pada implementasinya, runRisk menerima tiga argumen: Card sebagai functor dari jenis kartu RISK, CardName sebagai nama kartu RISK, dan Message sebagai pesan yang akan ditampilkan kepada pemain terkait efek kartu RISK tersebut. Selanjutnya, command ini memasukkan fakta bahwa pemain tersebut memiliki kartu RISK ke dalam basis pengetahuan permainan. Setelah itu, pesan efek dari kartu RISK ditampilkan kepada pemain. runRisk(Card, CardName, Message).

2.1.5.4 move (move.pl)

1. **move**

   Command "move" digunakan untuk memindahkan tentara dari satu wilayah ke wilayah lain dalam permainan Risiko. Predikat ini memiliki tiga argumen: WilayahA, WilayahB, dan Totaltroops. WilayahA adalah wilayah asal, WilayahB adalah wilayah tujuan, dan Totaltroops adalah jumlah tentara yang akan dipindahkan. Pemain dapat menggunakan move ketika pemain ingin memindahkan tentara dari satu wilayah ke wilayah lain yang dimilikinya. Misalnya, jika pemain ingin memperkuat wilayah perbatasan yang rentan, mereka dapat menggunakan move untuk memindahkan sebagian dari tentara mereka dari wilayah yang lebih aman. Contoh penggunaanya adalah move(au1, au2, 3).

2.1.5.5 draft (draft.pl)

1. **draft**

   Command "draft" digunakan untuk meletakkan tentara tambahan pada sebuah wilayah dalam permainan Risiko. Predikat ini memiliki dua argumen: Wilayah dan Totaltroops. Wilayah adalah wilayah tempat meletakkan tentara tambahan, dan Totaltroops adalah jumlah tentara tambahan yang akan ditempatkan.Pemain dapat menggunakan draft untuk menempatkan tentara tambahan pada wilayah yang ingin mereka perkuat sebelum melakukan serangan atau sebagai persiapan pertahanan. Misalnya, jika pemain telah mendapatkan tentara tambahan melalui kartu RISK atau sebagai tambahan setiap giliran, mereka dapat menggunakan draft untuk menempatkan tentara-tentara ini pada wilayah yang dianggap strategis. Contoh penggunaannya adalah draft(wilayah1, 5). Penggunaan tersebut berarti meletakkan 5 tentara tambahan pada wilayah1

2.1.6 Map (map.pl)

1. **slowFormatMap**

   Command "slowFormatMap" digunakan untuk menghasilkan efek cetakan yang berbeda dari sebuah peta permainan berdasarkan input teks dan daftar daerah yang sesuai. Fungsi ini memperlambat proses cetakan karakter dan mencetak karakter-karakter peta satu per satu untuk menciptakan efek animasi yang lebih menyenangkan dan menarik bagi pemain.Fungsi ini mengkonversi teks (dalam bentuk atom) menjadi daftar kode karakter ASCII (CodeList). Kemudian, slowFormatMapR digunakan untuk mengelola proses cetakannya. Selama proses cetak, karakter yang berupa angka dianggap sebagai kode daerah dan dicetak satu per satu. Beberapa karakter, seperti spasi dan tanda minus , dapat diabaikan untuk memberikan efek visual yang lebih baik. Saat pemain bergerak atau terjadi peristiwa tertentu, pemanggilan fungsi ini memberikan efek cetakan peta yang

lebih menarik dan memudahkan pemain untuk melihat dan memahami kondisi peta permainan. Contoh penggunaannya adalah slowFormatMap(‘ok’,List1,0.02).

2. **displayMap**

   Command displayMap digunakan untuk melihat kondisi peta saat ini. Informasi yang ditampilkan berupa tampilan peta, nama lokasi, dan juga jumlah tentara pada lokasi tersebut. Command ini juga menampilkan list berupa wilayah yang dikuasai oleh setiap pemain. Untuk menampilkan list tersebut program akan memanggil fungsi printPlayerTerritories. Contoh penggunaanya adalah displayMap.

3. **printPlayerTerritories**

   Command "printPlayerTerritories" digunakan untuk mencetak daftar wilayah yang dimiliki oleh setiap pemain. Fungsi ini juga memiliki fungsi tambahan, yaitu untuk memeriksa apakah seorang pemain telah gugur dari permainan berdasarkan kondisi tertentu.Fungsi ini memeriksa apakah daftar wilayah milik setiap pemain sudah habis. Jika sudah habis, maka pemain tersebut dianggap gugur dari permainan. Proses ini diimplementasikan dengan memanggil fungsi getPlayerTerritory untuk mendapatkan daftar wilayah yang dimiliki oleh setiap pemain.Penggunaan printPlayerTerritories adalah ketika ingin menampilkan daftar wilayah yang dimiliki oleh setiap pemain dalam permainan. Fungsi ini biasanya dipanggil setelah giliran seorang pemain berakhir. Jika suatu pemain kehilangan semua wilayahnya atau hanya memiliki satu wilayah dan tidak mendapatkan tambahan tentara pada giliran tersebut, pemain tersebut dianggap gugur dan dihapus dari daftar pemain yang aktif. Selanjutnya, pemain yang tersisa akan melanjutkan permainan. Contoh penggunaannya adalah printPlayerTerritories(Result).

2.1.7 Bonus

2.1.7.1 reinforcement (reinforcement.pl)

1. **rollReinforcement**

   Command rollReinforcement digunakan untuk mengundi apakah pemain mendapatkan wilayah reinforcement atau tidak dengan probabilitas yang sudah ditentukan. Untuk mekanisme penundiannya, program akan mengambil angka random, jika angka tersebut adalah 1 yang artinya pemain beruntung program akan menjalankan fungsi ambilReinforcement untuk menginisiasi bahwasannya sebuah wilayah reinforcement dimiliki oleh pemain tersebut. Setelah itu program akan mengupdate nilai reinforcementCount menggunakan fungsi updateReinforcementCount. Command ini akan dipanggil setelah seluruh pemain meletakkan seluruh tentara. Dalam setiap game wilayah reinforcement hanya dibatasi sebanyak 2 wilayah.

2. **updateReinforcementCount**

   Command ini digunakan untuk mengupdate nilai reinforcementCount. Nilai ini digunakan pada command rollReinforcement untuk memberikan batas bahwasannya untuk setiap game hanya tersedia 2 wilayah reinforcement.

3. **ambilReinforcement**

   Command ini digunakan untuk menginisiasi sebuah wilayah reinforcement, menandakan bahwa wilayah tersebut sekarang dimiliki oleh pemain. Command ini digunakan pada rollReinforcement.

2.1.7.2 win_probabilities (win_probabilities.pl)

1. **factorial**

   Command “factorial” digunakan untuk menghitung faktorial dari suatu bilangan. Pertama, terdapat aturan basis untuk N=0, di mana faktorialnya adalah 1. Selanjutnya, aturan rekursif digunakan untuk menghitung faktorial dari bilangan N. Proses rekursif ini terus berlanjut dengan mengurangkan N satu per satu hingga mencapai nilai 0. Contoh penggunaannya adalah factorial(5, Result).

2. **permute**

   Command “permute” digunakan untuk menghitung jumlah permutasi dari N elemen yang diambil R elemen pada setiap pengambilan. Permutasi ini menunjukkan berapa banyak cara berbeda yang dapat kita susun R elemen dari N elemen yang berbeda. Contoh penggunaannya adalah permute(5, 2, Result).

3. **choose**

   Command “choose” digunakan untuk menghitung kombinasi, atau disebut juga sebagai "n choose r" atau C(N,R). Kombinasi ini menggambarkan berapa banyak cara berbeda kita dapat memilih R elemen dari N elemen yang berbeda, tanpa memperhatikan urutan. Contoh penggunaanya adalah choose(5, 2, Result).

4. **exp**

   Command “exp” digunakan untuk menghitung eksponensial di mana A adalah basis dan X adalah eksponen. Operasi eksponensial menghasilkan hasil dari perkalian berulang basis A sebanyak X kali. Contoh penggunaannya adalah exp(2, 3, Result). yang akan menghasilkan operasi dua pangkat tiga.

5. **sigma dan sigma3**

   Command "sigma" dan "sigma3" mewakili fungsi sigma dalam logika pemrograman. Fungsi sigma, sering disebut juga sebagai notasi sigma digunakan untuk menjumlahkan serangkaian nilai sesuai dengan

suatu pola atau fungsi tertentu. Contoh penggunaannya adalah sigma3(sumProduct, 1, 3, 2, 3, Result).

6. **winProbability**

   Command "winProbability" digunakan untuk menghitung probabilitas bahwa hasil pelemparan N dadu memiliki jumlah yang lebih besar daripada hasil pelemparan M dadu. Predikat ini menggunakan fungsi sigma (wpInner) untuk melakukan perhitungan yang melibatkan probabilitas untuk jumlah hasil dadu tertentu.

2.1.7.3 world_leaders (world_leaders.pl)

1. **isDominating**

   Command "isDominating" digunakan untuk mengecek apakah suatu benua (continent) merupakan subset dari wilayah (territory) yang dimiliki oleh pemain saat ini. Pada dasarnya, predikat ini menginformasikan apakah pemain tersebut mendominasi suatu benua.Misalnya, dalam permainan menggunakan isDominating untuk menentukan apakah seorang pemain memiliki semua wilayah di suatu benua, sehingga dapat memperoleh bonus tentara sesuai dengan aturan permainan. Contoh penggunaan command nya adalah isDominating(northAmerica). Hal tersebut berarti apakah pemain saat ini mendominasi benua North America

2. **checkLose**

   Command "checkLose" digunakan untuk memeriksa apakah seorang pemain telah kehilangan semua wilayahnya dan oleh karena itu telah kalah dalam permainan. Contoh penggunaannya adalah checkLose(player1). Penggunaan tersebut berarti apakah player1 telah dinyatakan kalah atau belum.

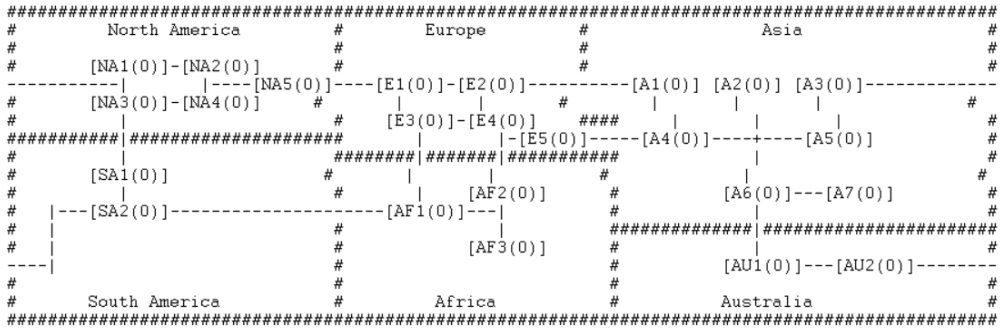
2.1.7.4 cheat (risk.pl)

1. **riskC**

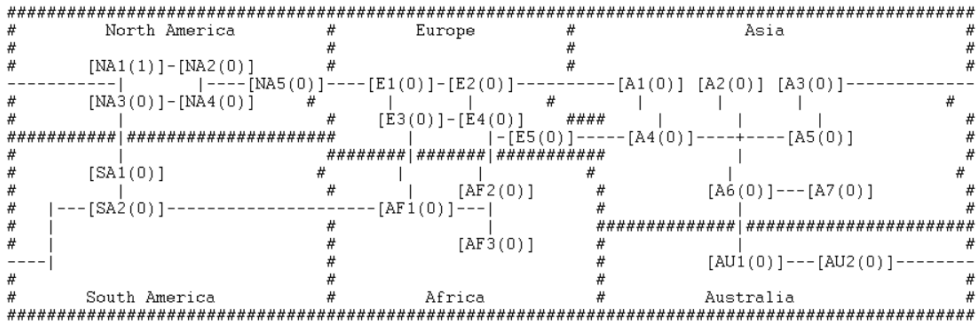
   Command “riskC” digunakan untuk memainkan kartu RISK dengan memilih jenis kartu RISK yang diinginkan. Kartu RISK memberikan efek khusus yang dapat mempengaruhi dinamika permainan. Pada skenario penggunaannya, pemain memilih jenis kartu RISK yang ingin mereka dapatkan dari sebuah daftar pilihan yang tersedia. Pada implementasinya, setelah memastikan bahwa pemain masih dalam kondisi bermain dan kartu RISK tersedia, pemanggilan riskC memberikan opsi pemilihan jenis kartu RISK kepada pemain. Pemilihan dilakukan dengan memasukkan nomor yang sesuai dengan jenis kartu yang diinginkan. Setelah pemain memilih, efek dari kartu RISK yang dipilih tersebut akan dijalankan dalam permainan. Contoh penggunaannya adalah riskC.

BAB 3 HASIL EKSEKUSI PROGRAM

```
| ?- start.
Masukkan jumlah pemain: 2.
Jumlah pemain: 2

Masukkan nama pemain 1: asep.
Masukkan nama pemain 2: budi.


asep melempar dadu dan mendapatkan 2
budi melempar dadu dan mendapatkan 8

Urutan pemain: budi - asep
budi dapat mulai terlebih dahulu.
Setiap pemain mendapatkan 24 tentara.
```

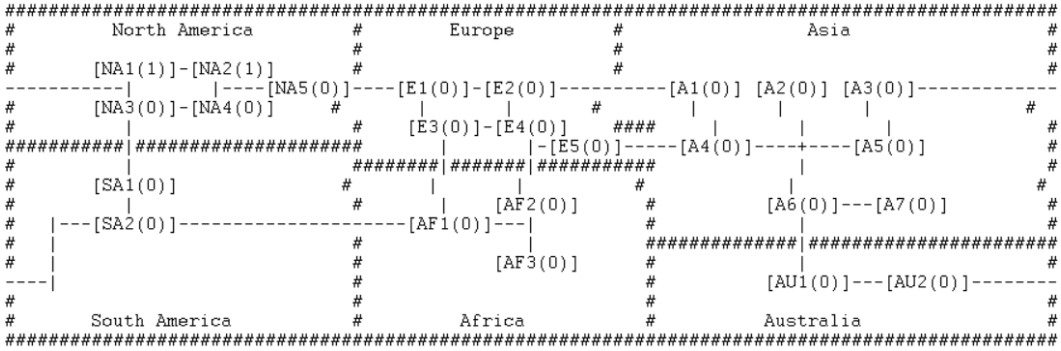



```
Wilayah yang dikuasai:

budi: []
asep: []

Sekarang giliran pemain: budi

Giliran budi untuk memilih wilayahnya.

Giliran budi untuk memilih wilayahnya.

(16 ms) yes
| ?- takeLocation(na1).
budi mengambil wilayah na1
```

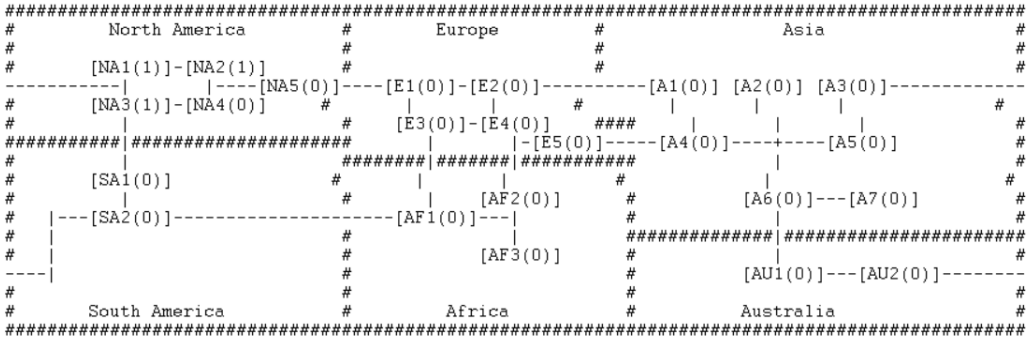



```
Wilayah yang dikuasai:

budi: [na1]
asep: []

Sekarang giliran pemain: asep

Giliran asep untuk memilih wilayahnya.

yes
| ?- takeLocation(na2).
asep mengambil wilayah na2
```

```
Giliran asep untuk memilih wilayahnya.

yes
| ?- takeLocation(na2).
asep mengambil wilayah na2
```

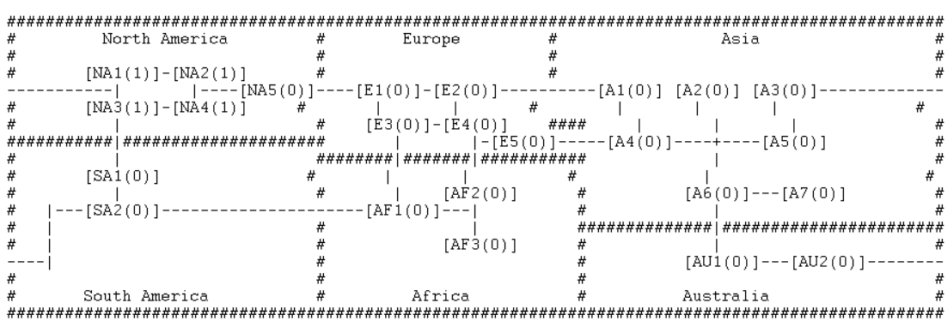



```
Wilayah yang dikuasai:

budi: [na1]
asep: [na2]

Sekarang giliran pemain: budi

Giliran budi untuk memilih wilayahnya.


Giliran budi untuk memilih wilayahnya.

yes
| ?- takeLocation(na3).
budi mengambil wilayah na3
```

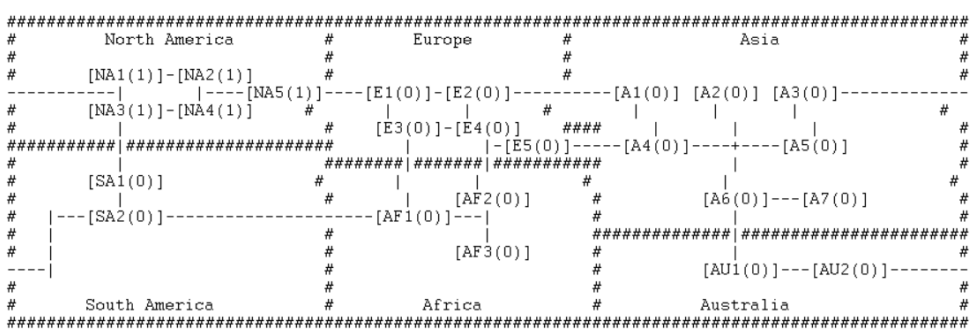



```
Wilayah yang dikuasai:

budi: [na1, na3]
asep: [na2]

Sekarang giliran pemain: asep

Giliran asep untuk memilih wilayahnya.
```

```
Sekarang giliran pemain: asep

Giliran asep untuk memilih wilayahnya.

yes
| ?- takeLocation(na4).
asep mengambil wilayah na4
```

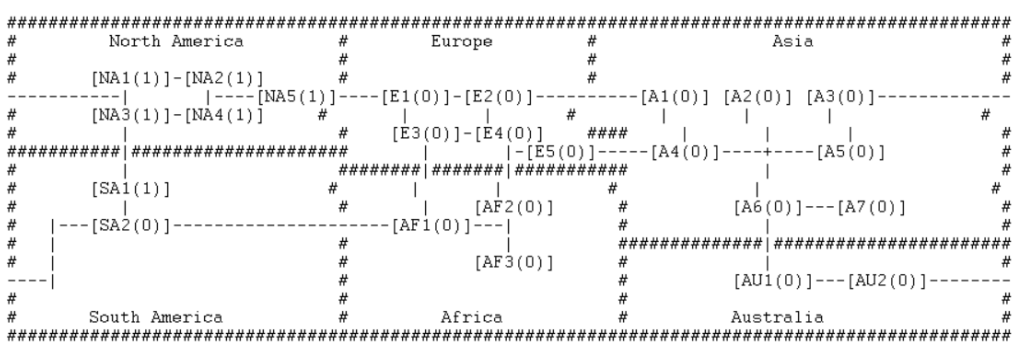



```
Wilayah yang dikuasai:

budi: [na1, na3]
asep: [na2, na4]

Sekarang giliran pemain: budi

Giliran budi untuk memilih wilayahnya.


Sekarang giliran pemain: budi

Giliran budi untuk memilih wilayahnya.

(16 ms) yes
| ?- takeLocation(na5).
budi mengambil wilayah na5
```

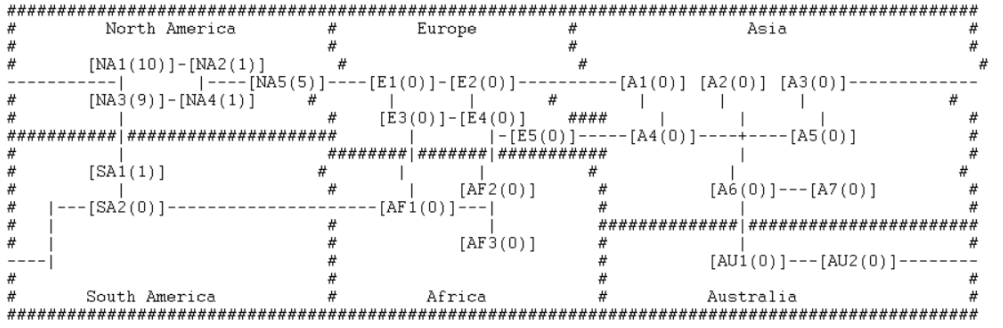



```
Wilayah yang dikuasai:

budi: [na1, na3, na5]
asep: [na2, na4]

Sekarang giliran pemain: asep

Giliran asep untuk memilih wilayahnya.
```

```
Sekarang giliran pemain: asep

Giliran asep untuk memilih wilayahnya.

(15 ms) yes
| ?- takeLocation(sa1).
asep mengambil wilayah sa1
```



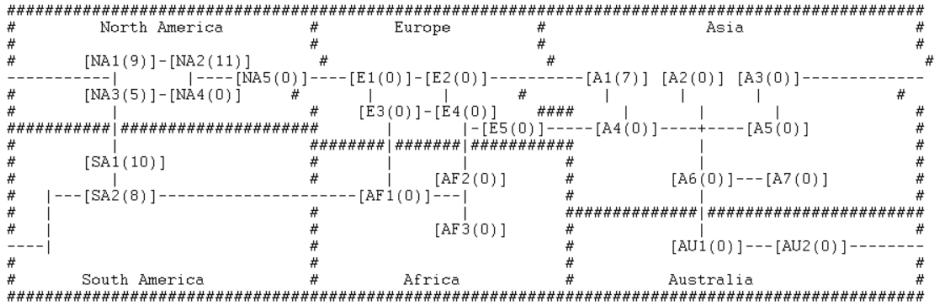

```
Wilayah yang dikuasai:

budi: [na1, na3, na5]
asep: [na2, na4, sa1]

Sekarang giliran pemain: budi

Seluruh wilayah telah diambil pemain.
Memulai pembagian sisa tentara.
Giliran budi untuk meletakkan tentaranya.

Sekarang giliran pemain: budi

Seluruh wilayah telah diambil pemain.
Memulai pembagian sisa tentara.
Giliran budi untuk meletakkan tentaranya.

(16 ms) yes
| ?- placeAutomatic.

budi meletakkan 5 tentara di wilayah na1
budi meletakkan 3 tentara di wilayah na3
budi meletakkan 5 tentara di wilayah na3
budi meletakkan 4 tentara di wilayah na5
budi meletakkan 4 tentara di wilayah na1
```



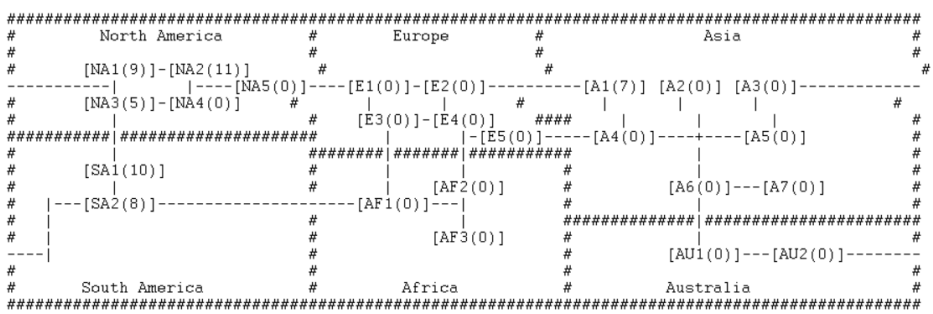

```
Wilayah yang dikuasai:

budi: [na1, na3, na5]
asep: [na2, na4, sa1]

Sekarang giliran pemain: asep

Giliran asep untuk meletakkan tentaranya.
```

```
Sekarang giliran pemain: asep

Giliran asep untuk meletakkan tentaranya.

yes
| ?- placeAutomatic.

asep meletakkan 3 tentara di wilayah sa1
asep meletakkan 1 tentara di wilayah na2
asep meletakkan 2 tentara di wilayah na2
asep meletakkan 1 tentara di wilayah na2
asep meletakkan 3 tentara di wilayah sa1
asep meletakkan 1 tentara di wilayah na2
asep meletakkan 2 tentara di wilayah na2
asep meletakkan 3 tentara di wilayah na4
asep meletakkan 4 tentara di wilayah na4
asep meletakkan 1 tentara di wilayah na2
Seluruh pemain telah meletakkan sisa tentara.
Permainan dimulai!


Satu ronde sudah berakhir.
Ronde berikutnya dimulai.
Reinforcement roll for player budi: 1
Pemain budi berhasil mendapatkan wilayah reinforcement!
Pemain budi mendapatkan wilayah r1.

Sekarang giliran player budi!
Tambahan tentara wilayah: Pemain mendapatkan 2 tentara tambahan.
Benua dikuasai: Pemain mendapatkan 0 tentara bonus.
Player budi mendapatkan 2 tentara tambahan.

Sekarang giliran player budi!
Tambahan tentara wilayah: Pemain mendapatkan 2 tentara tambahan.
Benua dikuasai: Pemain mendapatkan 0 tentara bonus.
Player budi mendapatkan 2 tentara tambahan.

##########################################################################################
#         North America          #        Europe         #                 Asia                 #
#                                #                       #                                      #
#       [NA1(9)]-[NA2(9)]        #                       #                                      #
-----------|       |----[NA5(0)]----[E1(0)]-[E2(0)]----------[A1(7)] [A2(0)] [A3(0)]-------------
#       [NA3(5)]-[NA4(0)]     #       |       |       #        |       |       |              #
#          |                   #    [E3(0)]-[E4(0)]    ####     |       |       |              #
###########|#####################       |       |-[E5(0)]-----[A4(0)]----+----[A5(0)]           #
#          |                   ########|#######|###########              |                      #
#       [SA1(10)]              #       |       |           #             |                      #
#          |                   #       |    [AF2(0)]       #          [A6(0)]---[A7(0)]         #
#   |---[SA2(8)]----------------------[AF1(0)]---|          #             |                      #
#   |                           #               |          ##############|#######################
#   |                           #            [AF3(0)]      #             |                      #
----|                           #                          #          [AU1(0)]---[AU2(0)]--------
#                               #                          #                                      #
#       South America           #         Africa           #          Australia                   #
##########################################################################################

Wilayah yang dikuasai:

budi: [sa1, na2, na3, r1]
asep: [na1, sa2, a1]

Sekarang giliran pemain: budi

(47 ms) yes
| ?- draft(sa1,2).

Player budi meletakkan 2 tentara tambahan di sa1
Tentara total di sa1: 12
Jumlah Pasukan Tambahan Player budi: 0

yes
| ?- move(sa1,na2,2).

budi memindahkan 2 tentara dari sa1 ke na2
Jumlah tentara di sa1: 10
Jumlah tentara di na2: 11
```

```
yes
| ?- risk.

Player budi mendapatkan risk card CEASEFIRE ORDER
Hingga giliran berikutnya, wilayah pemain tidak dapat diserang oleh lawan.

(16 ms) yes
| ?- attack.

Sekarang giliran player budi menyerang.
```

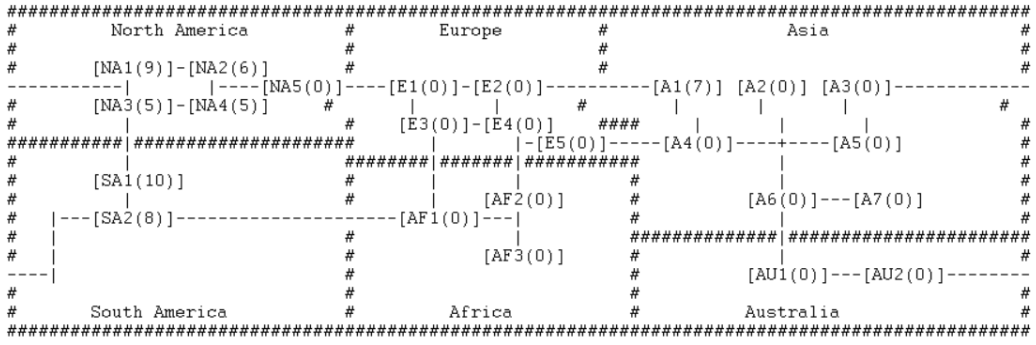



```
Wilayah yang dikuasai:

budi: [sa1, na2, na3, r1]
asep: [na1, sa2, a1]

Sekarang giliran pemain: budi

Pilihlah daerah yang ingin anda kirim untuk bertempur: na2.

Player budi ingin memulai penyerangan dari daerah na2
Dalam daerah na2, anda memiliki sebanyak 11 tentara.

Masukkan banyak tentara yang akan bertempur: 10.
Player budi mengirim sebanyak 10 tentara untuk bertempur.
```




```
Wilayah yang dikuasai:

budi: [sa1, na2, na3, r1]
asep: [na1, sa2, a1]

Sekarang giliran pemain: budi

Daerah tetangga yang dapat diserang:

1. na1 (66 win chance)
2. na4 (100 win chance)

Pilih wilayah yang ingin diserang (angka): 2.
```

```
Wilayah yang ingin diserang berhasil dipilih.
Perang dimulai!
Player budi
Dadu 1: 4
Dadu 2: 5
Dadu 3: 5
Dadu 4: 4
Dadu 5: 2
Dadu 6: 6
Dadu 7: 5
Dadu 8: 6
Dadu 9: 6
Dadu 10: 4
Total: 47

Player x
Total: 0


Player budi menang! Wilayah na4 sekarang dikuasai oleh player budi.
Masukkan banyak tentara yang ingin dikirim ke wilayah tersebut: 5.

Tentara di wilayah na2: 6
Tentara di wilayah na4: 5
```




```
Wilayah yang dikuasai:

budi: [sa1, na2, na3, r1]
asep: [na1, sa2, a1]

Sekarang giliran pemain: budi

(31 ms) yes
| ?- endTurn.

Player budi mengakhiri giliran.
Sekarang giliran player asep!
Tambahan tentara wilayah: Pemain mendapatkan 1 tentara tambahan.
Benua dikuasai: Pemain mendapatkan 5 tentara bonus.
Player asep mendapatkan 6 tentara tambahan.


| ?- riskC.

Silahkan pilih kartu RISK yang ingin didapatkan:

1. CEASEFIRE ORDER
2. SUPER SOLDIER SERUM
3. AUXILIARY TROOPS
4. REBELLION
5. DISEASE OUTBREAK
6. SUPPLY CHAIN ISSUE

Pilih: 1.

Player asep mendapatkan risk card CEASEFIRE ORDER
Hingga giliran berikutnya, wilayah pemain tidak dapat diserang oleh lawan.

yes
```

```
| ?- checkPlayerDetail(p1).

Nama                    : budi
Benua                   : []
Total Wilayah           : 4
Total Tentara Aktif     : 21
Total Tentara Tambahan  : 0

yes
(15 ms) yes
| ?- checkLocationDetail(na3).

Kode              : NA3
Nama              : Grenada
Pemilik           : budi
Total Tentara     : 5
Tetangga          : [Panama,Brazil,Nikaragua]

(15 ms) yes
| ?- checkIncomingTroops(p1).


Nama                    : budi
Total Wilayah           : 4
Tentara Tambahan        : 2
Bonus - Amerika Utara   : 0
Bonus - Amerika Selatan : 0
Bonus - Eropa           : 0
Bonus - Afrika          : 0
Bonus - Asia            : 0
Bonus - Australia       : 0
Total tentara tambahan  : 2
```

```
(32 ms) yes
| ?- checkPlayerTerritories(p1).

Nama            : budi

Benua Amerika Utara (2/5)
NA1
Nama                    : Greenland
Jumlah Tentara          : 6

NA3
Nama                    : Grenada
Jumlah Tentara          : 5


Benua Amerika Selatan (1/2)
SA1
Nama                    : Brazil
Jumlah Tentara          : 10
```

BAB 4 LAMPIRAN

4.1 Pembagian dan Persentase Kerja

Nomor	Nama	NIM	Pembagian Kerja
1	Shabrina Maharani	13522134	Player, Wilayah, Laporan
2	Auralea Alvinia Syaikha	13522148	Move, Draft, Readme
3	Muhammad Roihan	13522152	Reinforcement, Laporan,Readme
4	Muhammad Rasheed Qais Tandjung	13522158	Inisialisasi, Attack, Endturn, Map, World Leaders, Win Probabilities, Cheat, Risk

4.2 Link Repository Github

1. **Link Repository Github :**
   https://github.com/GAIB21/tugas-besar-if2121-logika-komputasional-2023-santaidulugasih-hmin2

REFERENSI

Bahan Kuliah Logika Komputasional 2023/2024