Spesifikasi Tugas Besar 1
IF2110 Algoritma dan Struktur Data

Revisi

ver. 17 Oktober 2019

Deskripsi Game

Saat avatar Aang belum muncul, dunia menjadi kacau dan terjadi perang dunia. Perang ini diikuti oleh 4 negara - Api, Air, Tanah, dan Udara. Pada mulanya, keempat negara berada di keempat penjuru dunia. Mereka memiliki pasukan masing-masing dan jumlahnya terus bertambah. Tiap negara dapat bergerak dan menduduki sebuah kota. Untuk menduduki sebuah kota, sebuah negara harus menyerang dengan pasukan lebih banyak dari penghuni kota tersebut. Ketika sebuah kota diduduki, kota tersebut akan menghasilkan pasukan untuk negara yang berhasil mendudukinya. Pemenangnya adalah negara yang berhasil menduduki seluruh kota.

Game Avatar World War adalah game *turn-based strategy* yang mensimulasikan perang dunia yang sudah diceritakan di atas. Game ini dimainkan dengan cara memasukkan perintah melalui *command line interface* dengan *command* yang akan dijelaskan pada masing-masing fitur game. Game ini dibuat dengan bahasa C.

Game Mechanics

1. **Main Menu**
   a. Pada awal game, pemain memilih untuk memulai permainan baru.
   b. **(Bonus)** Selain memulai permainan baru, pemain dapat *load* game yang sudah disimpan.
   c. Jumlah pemain tepat 2. Setiap pemain harus memiliki identitas **warna** unik untuk penggambaran peta pada terminal. Modul penulisan berwarna akan disediakan oleh asisten (baca bab Penggunaan Modul Berwarna)

2. **Peta**
   a. Pada permainan ini, terdapat peta Dunia Avatar dengan ukuran NxM. Ukuran peta minimal 10 x 10 dan maksimal 20 x 30 (ditentukan oleh konfigurasi permainan). Titik (i, j) merupakan sebuah **petak** pada baris ke-i dan kolom ke-j.
   b. Berikut adalah ilustrasi peta 10 x 15 yang ditampilkan kepada pemain

```
******************
*C         V    T C*
*  C              *
*T        V      C *
*     F           *
*              F   *
*  T              *
*           T      *
* C    V          T*
*               C  *
*C T              C*
******************
```

Petak diberi boundary karakter *. Karakter pada petak melambangkan bangunan yang ada di petak tersebut, sedangkan warna font sama dengan warna pemain. Petak tanpa bangunan cukup dituliskan sebagai spasi saja.
Pada ilustrasi di atas, pemain 1 (biru) memiliki 2 castle dan 1 village, pemain 2 (merah) memiliki 2 castle dan 1 tower.

3. Bangunan

a. Sebuah bangunan memiliki properti sebagai berikut:

i. Kepemilikan
Mengindikasikan pemain yang memiliki bangunan yang bersangkutan. Sebuah bangunan mungkin tidak dimiliki siapapun.

ii. Jumlah pasukan
Merupakan banyaknya pasukan yang sedang menempati bangunan. Dapat diasumsikan banyaknya pasukan tidak melebihi 1 juta.

iii. Level
Merupakan bilangan bulat 1 - 4 menyatakan level dari bangunan

iv. Nilai penambahan pasukan (A)
Pada setiap awal giliran, jumlah pasukan pada bangunan ini bertambah sebanyak A.

v. Maksimum penambahan pasukan (M)
Jika bangunan sudah menampung pasukan lebih dari sama dengan M, maka penambahan pasukan pada poin (ii) tidak dilakukan.

vi. Pertahanan (P)
Berupa nilai ya / tidak yang menyatakan apakah bangunan memiliki pertahanan. Jika ya, saat bangunan diserang oleh N pasukan, jumlah pasukan di bangunan ini hanya berkurang sebesar ¾ N (dibulatkan ke bawah)

vii. Pasukan awal (U)
Merupakan jumlah pasukan awal yang harus dikalahkan untuk menjadikan bangunan ini milik pemain.

b. Bangunan dapat menaikkan level. Saat menaikkan level, harus ada pasukan sebanyak M / 2 di bangunan tersebut. Setelah level naik, jumlah pasukan pada bangunan akan berkurang sebanyak M / 2.
c. Terdapat beberapa jenis bangunan, dengan detail nilai properti adalah sebagai berikut:

i. Castle (C)

Level	A	M	P	U
1	10	40	no	40
2	15	60	no	-
3	20	80	no	-
4	25	100	no	-

ii. Tower (T)

Level	A	M	P	U
1	5	20	yes	30
2	10	30	yes	-
3	20	40	yes	-
4	30	50	yes	-

iii. Fort (F)

Level	A	M	P	U
1	10	20	no	80
2	20	40	no	-
3	30	60	yes	-
4	40	80	yes	-

iv. Village (V)

Level	A	M	P	U
1	5	20	no	20
2	10	30	no	-
3	15	40	no	-
4	20	50	no	-

4. Skill

a. Dalam permainan, terdapat beberapa skill yang dapat digunakan pemain pada gilirannya. Tiap pemain memiliki daftar skill yang bentuknya menyerupai *queue*.
b. Pada awal permainan, tiap pemain mendapat beberapa skill dengan urutan tertentu yang dapat digunakan. Setelah pemain menggunakan sebuah skill, skill hangus dan digantikan skill selanjutnya yang ada pada daftar. Pada saat permainan berlangsung, mungkin terjadi penambahan skill dan masuk pada daftar skill.

c. Daftar skill beserta efek dan penyebab pemain mendapat skill tersebut:
   i. Instant Upgrade
      Seluruh bangunan yang dimiliki pemain akan naik 1 level.
      Pemain tidak akan mendapat skill ini selain dari daftar skill awal.
   ii. Shield (**Bonus**)
      Seluruh bangunan yang dimiliki oleh pemain akan memiliki pertahanan selama 2 turn. Apabila skill ini digunakan 2 kali berturut-turut, durasi tidak akan bertambah, namun menjadi nilai maksimum.
      Pemain mendapat skill ini jika setelah sebuah lawan menyerang, bangunan pemain berkurang 1 menjadi sisa 2.
   iii. Extra Turn
      Setelah giliran pengaktifan skill ini berakhir, pemain selanjutnya tetap pemain yang sama.
      Pemain mendapat skill ini jika Fort pemain tersebut direbut lawan.
   iv. Attack Up (**Bonus**)
      Pada giliran ini, setelah skill ini diaktifkan, pertahanan bangunan musuh tidak akan mempengaruhi penyerangan.
      Pemain mendapat skill ini jika pemain baru saja menyerang Tower lawan dan jumlah towernya menjadi 3.
   v. Critical Hit (**Bonus**)
      Pada giliran ini, setelah skill diaktifkan, jumlah pasukan pada bangunan yang melakukan serangan tepat selanjutnya hanya berkurang ½ dari jumlah seharusnya.
      Pemain mendapat skill ini jika lawan baru saja mengaktifkan skill Extra Turn.
   vi. Instant Reinforcement
      Seluruh bangunan mendapatkan tambahan 5 pasukan.
      Pemain mendapat skill ini di akhir gilirannya bila semua bangunan yang ia miliki memiliki level 4.
   vii. Barrage
      Jumlah pasukan pada seluruh bangunan musuh akan berkurang sebanyak 10 pasukan.
      Pemain mendapat skill ini jika lawan baru saja bertambah bangunannya menjadi 10 bangunan.

5. Kondisi Awal Permainan

a. Pada saat permainan dimulai, game akan membaca konfigurasi permainan dari file eksternal (dijelaskan pada bab selanjutnya).
b. *Queue* skill setiap pemain berisi 1 buah skill, yaitu Instant Upgrade
c. Setelah itu, permainan akan berjalan seperti pada poin 6

6. Mekanisme Permainan

a. Permainan dimainkan secara bergilir, dimulai dari pemain 1, pemain 2, dan seterusnya hingga game selesai.

b. Pada saat giliran pemain X dimulai, setiap petak yang bangunannya dimiliki oleh pemain X akan bertambah jumlah pasukannya sesuai jenis bangunan. Tergantung jumlah pasukan yang sudah ada, mungkin saja jumlah pasukan tidak akan bertambah (lihat poin 3).
c. Pada saat bermain, peta akan selalu ditampilkan beserta kondisi pemain sekarang, setidak-tidaknya:
   i. Nomor pemain
   ii. Daftar bangunan yang dimiliki, masing-masing meliputi jenis bangunan, lokasi bangunan, jumlah pasukan pada petak yang bersangkutan, dan level dari bangunan.
   iii. Skill terdepan di *queue* yang dapat dipakai, jika *queue* tidak kosong.
d. Kemudian, pemain dapat memasukkan *command* yang dijelaskan pada poin 8.
e. Giliran akan berakhir apabila pemain sudah memasukkan *command* END_TURN.

7. Mekanisme *Attack*

a. Setiap bangunan pada satu giliran hanya bisa menyerang sekali saja.
b. Bangunan hanya bisa menyerang bangunan lain (selain milik pemain sendiri) yang saling terhubung.
c. Pemain dapat menuliskan berapa banyak pasukan yang digunakan untuk menyerang, misalkan X pasukan menyerang bangunan yang memiliki Y pasukan.
   i. Jika X < Y, maka bangunan yang diserang tetap menjadi milik lawan, namun berkurang jumlah pasukannya di bangunan itu menjadi Y - X. Jumlah pasukan di bangunan yang menyerang berkurang sebanyak X.
   ii. Jika X >= Y, maka bangunan yang diserang menjadi milik penyerang, dan memiliki jumlah pasukan (X - Y). Jumlah pasukan di bangunan yang menyerang berkurang sebanyak X.
d. Dalam kasus dimana bangunan yang diserang memiliki pertahanan, nilai serang akan menjadi kurang dari X (lihat bagian 3.a.iv)
e. Bila bangunan berpindah kepemilikan, level akan kembali menjadi 1.

8. Commands

Setelah state awal permainan diinisiasi, maka game akan selalu menunggu masukan dari pemain. Berikut adalah masukan *command* yang dapat diberikan oleh pengguna.

a. ATTACK
   Digunakan untuk melakukan serangan ke bangunan lain.

   Contoh skenario 1:

```
ENTER COMMAND: ATTACK
Daftar bangunan:
1. Castle (1,15) 20 lv. 3
2. Tower (1,13) 50 lv. 1
```

```
3. Castle (3,14) 30 lv. 2
Bangunan yang digunakan untuk menyerang: 2
Daftar bangunan yang dapat diserang:
1. Village (1,9) 20 lv. 1
2. Tower (5,12) 30 lv. 2
Bangunan yang diserang: 2
Jumlah pasukan: 44
Bangunan menjadi milikmu!
```

Setelah skenario di atas, bangunan Tower (5, 12) menjadi milik penyerang dengan level 1 dan jumlah pasukan 4. Tower (3, 13) akan tetap memiliki level 1 dengan pasukan menjadi hanya 6.

Contoh skenario 2:

```
ENTER COMMAND: ATTACK
Daftar bangunan:
1. Castle (1,15) 20 lv. 3
2. Tower (1,13) 50 lv. 1
3. Castle (3,14) 30 lv. 2
Bangunan yang digunakan untuk menyerang: 2
Daftar bangunan yang dapat diserang:
1. Village (1,9) 20 lv. 1
2. Tower (5,12) 30 lv. 2
Bangunan yang diserang: 2
Jumlah pasukan: 20
Bangunan gagal direbut.
```

Setelah skenario di atas, bangunan Tower (5, 12) tidak berganti kepemilikan dengan level 2 dan jumlah pasukan 15. Tower (3,13) akan berkurang pasukannya menjadi tersisa 30.

Contoh skenario 3:

```
ENTER COMMAND: ATTACK
Daftar bangunan:
1. Castle (1,15) 20 lv. 3
2. Tower (1,13) 50 lv. 1
3. Castle (3,14) 30 lv. 2
Bangunan yang digunakan untuk menyerang: 1
Tidak ada bangunan yang dapat diserang
```

Skenario di atas mungkin terjadi bila bangunan Castle (1,15) hanya terhubung ke bangunan yang dimiliki oleh pemain itu sendiri.

b. LEVEL_UP

Digunakan untuk menaikkan level dari suatu bangunan. Dalam satu giliran, level up

dapat dilakukan berkali-kali.

Contoh skenario 1:

```
ENTER COMMAND: LEVEL_UP
Daftar bangunan:
1. Castle (1,15) 20 lv. 3
2. Tower (1,13) 50 lv. 1
3. Castle (3,14) 30 lv. 2
Bangunan yang akan di level up: 2
Level Tower-mu meningkat menjadi 2!
```

Pada skenario di atas, Tower menjadi level 2 dengan pasukan tersisa 40 (berkurang 10, karena M = 20 dan pasukan berkurang M / 2).

Contoh skenario 2:

```
ENTER COMMAND: LEVEL_UP
Daftar bangunan:
1. Castle (1,15) 20 lv. 3
2. Tower (1,13) 50 lv. 1
3. Castle (3,14) 30 lv. 2
Bangunan yang akan di level up: 1
Jumlah pasukan Castle kurang untuk level up
```

Pada skenario di atas, Castle butuh 40 pasukan untuk level up, namun jumlah pasukan Castle masih kurang.

c. SKILL
Digunakan untuk memakai *skill* yang sedang dimiliki oleh pemain.

```
ENTER COMMAND: SKILL
```

d. UNDO
Digunakan untuk membatalkan perintah terakhir.

```
ENTER COMMAND: UNDO
```

e. END_TURN
Digunakan untuk mengakhiri giliran dari pemain.

```
ENTER COMMAND: END_TURN
```

f. SAVE (Bonus)
Digunakan untuk menyimpan status permainan sekarang ke dalam file eksternal.

```
ENTER COMMAND: SAVE
Lokasi save file: C:\User\Documents\save_file.dat
```

```
Game berhasil di save!
```

g. MOVE (**Bonus**)
Digunakan untuk memindahkan pasukan dari suatu bangunan ke bangunan lain milik pemain yang terhubung dengan bangunan tersebut.

Contoh :

```
ENTER COMMAND: MOVE
Daftar bangunan:
1. Castle (1,15) 20 lv. 3
2. Tower (1,13) 50 lv. 1
3. Castle (3,14) 30 lv. 2
Pilih bangunan: 2
Daftar bangunan terdekat:
1. Village (1,9) 20 lv. 1
2. Tower (5,12) 30 lv. 2
Bangunan yang akan menerima: 1
Jumlah pasukan: 10
10 pasukan dari Tower (1,13) telah berpindah ke Village (1,9)
```

h. EXIT
Digunakan untuk keluar dari permainan/program.

```
ENTER COMMAND: EXIT
```

9. Kondisi Akhir

a. Permainan berakhir apabila seorang pemain tidak punya bangunan lagi.

Contoh Tampilan Game

Berikut adalah contoh state dari permainan yang akan digunakan untuk beberapa command yang penting. Misalkan player 1 adalah player dengan warna merah, dan player 2 adalah player warna biru. Perhatikan bahwa Anda tidak perlu mengimplementasikan output sama persis dengan contoh.

```
******************
*C        V    T C*
*  C              *
*T       V      C *
*     F           *
*             F    *
*  T              *
*           T      *
* C    V          T*
*              C   *
*C T             C*
******************
Player 2
1. Castle (1,15) 20 lv. 3
2. Tower (1,13) 50 lv. 1
3. Castle (3,14) 30 lv. 2
Skill Available: IU
ENTER COMMAND:
```

Penjelasan: Untuk skill, digunakan akronim dari skill, penggunaan akronim bebas, tapi disarankan untuk menggunakan huruf pertama dari setiap kata pada nama skill.

Konfigurasi Permainan

Pada awal permainan, konfigurasi permainan dibaca dari file eksternal dengan format berikut:

Baris pertama berisi 2 bilangan, N dan M menyatakan tinggi dan lebar peta.

Baris kedua berisi 1 bilangan, B (4 < B < 30) menyatakan banyaknya bangunan. Bangunan dinomori dari 1 hingga B. Bangunan 1 adalah bangunan mula-mula pemain 1 dan bangunan 2 adalah bangunan mula-mula pemain 2.

B baris berikutnya menjelaskan bangunan yang ada di peta, dengan baris ke-i berisi bangunan nomor i. Tiap baris terdiri dari satu karakter representasi jenis bangunan (C/T/F/V), indeks baris dan kolom posisi bangunan, dipisahkan oleh spasi.

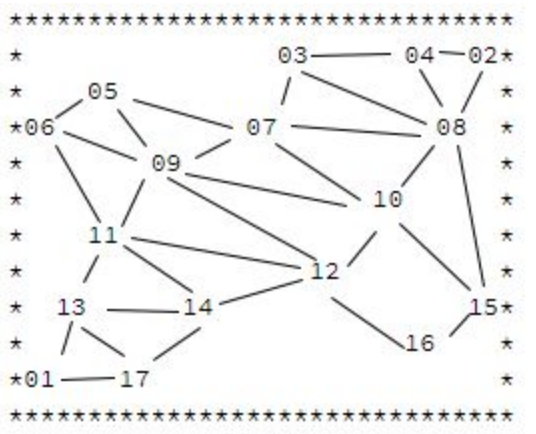
B baris berikutnya terdiri dari B bilangan yang merepresentasikan graf keterhubungan tiap pasang bangunan. Tiap bilangan adalah 0 (tidak terhubung) atau 1 (terhubung). Keterhubungan bersifat dua arah, dan sebuah bangunan tidak terhubung ke dirinya sendiri. Jika bilangan pada baris ke-i dan kolom ke-j bernilai 1, berarti bangunan i dan j terhubung dua arah.

Contoh file konfigurasi:

```
10 10
17
C 10 1
C 1 15
V 1 9
T 1 13
C 2 3
T 3 1
V 3 8
C 3 14
F 4 5
F 5 12
T 6 3
T 7 10
C 8 2
V 8 6
T 8 15
C 9 13
T 10 3
0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1
0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0
0 0 0 1 0 0 1 1 0 0 0 0 0 0 0 0 0
0 1 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0
0 0 0 0 0 1 1 0 1 0 0 0 0 0 0 0 0
0 0 0 0 1 0 0 0 1 0 1 0 0 0 0 0 0
0 0 1 0 1 0 0 1 1 1 0 0 0 0 0 0 0
0 1 1 1 0 0 1 0 0 1 0 0 0 0 1 0 0
0 0 0 0 1 1 1 0 0 1 1 1 0 0 0 0 0
0 0 0 0 0 0 1 1 1 0 0 1 0 0 1 0 0
0 0 0 0 0 1 0 0 1 0 0 1 1 1 0 0 0
0 0 0 0 0 0 0 0 1 1 1 0 0 1 0 1 0
1 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 1
0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0 1
0 0 0 0 0 0 0 1 0 1 0 0 0 0 0 1 0
0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0
1 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0
```

File konfigurasi di atas bersesuaian dengan tampilan peta berikut:

```
******************
*          V    T C*
*  C               *
*T        V       C *
*     F            *
*              F    *
*  T                *
*           T       *
* C    V           T*
*               C   *
*C T                *
******************
```



Penggunaan Modul Print Berwarna

Karena setiap pemain diharuskan menggunakan warna berbeda, asisten akan menyediakan sebuah modul untuk melakukan print berwarna pada terminal. Modul ini dapat digunakan pada Windows Powershell dan terminal UNIX. Silahkan unduh **pcolor.c** dan **pcolor.h** pada Olympia, serta lihatlah contoh pada pcolor.h untuk penggunaannya. Anda tentu dapat memodifikasi modul tersebut jika diperlukan.

Daftar ADT yang Digunakan

Anda diwajibkan menggunakan ADT di bawah ini. Selain itu, Anda dapat pula menggunakan ADT lain, namun cantumkan analisis alasan kenapa menggunakan ADT tersebut pada laporan.

1. **ADT Point**
   ADT ini digunakan untuk menunjukkan posisi bangunan
2. **ADT Array Implisit**
   ADT ini digunakan untuk *menyimpan daftar bangunan yang tersedia pada peta*
3. **ADT Matrix**
   ADT ini digunakan sebagai representasi peta
4. **ADT Mesin Karakter dan Mesin Kata**
   ADT ini digunakan untuk:
   a. Membaca informasi peta dari file eksternal,
   b. Membaca *command* dari interaksi user terhadap program, dan
   c. Membaca *state* dari *game* yang sudah pernah disimpan

5. **ADT Queue**
   ADT ini digunakan untuk penyimpanan daftar skill masing-masing pemain.

6. **ADT Stack**
   ADT ini digunakan untuk mendukung mekanisme command UNDO.

7. **ADT List**
   ADT ini digunakan untuk menyimpan daftar indeks bangunan yang dimiliki oleh seorang pemain.

8. **ADT Graph (variasi multilist)**
   ADT ini digunakan untuk menggambarkan bangunan pada peta

9. **ADT Lain**
   ADT ini dibuat dan digunakan untuk abstraksi beberapa hal lain. Mahasiswa dipersilakan untuk mendefinisikan sendiri.