


LAPORAN TUGAS BESAR IF2110/ALGORITMA DAN STRUKTUR DATA

AVATAR WORLD WAR GAME

Dipersiapkan oleh:
Kelompok 08 / K-03

Muhammad Hasan	13518012
Anna Elvira Hartoyo	13518045
Daniel Riyanto	13518075
Faris M. Kautsar	13518105
Gregorius Jovan K.	13518135

**Sekolah Teknik Elektro dan Informatika - Institut Teknologi Bandung
Jl. Ganesha 10, Bandung 40132**

	Sekolah Teknik Elektro dan Informatika ITB	Nomor Dokumen		Halaman
		<i>IF2110-TB-08-03</i>		<i>24</i>
		<i>Revisi</i>	<i>1</i>	<i>25 November 2019</i>

Daftar Isi

1.	Ringkasan.....	4
2.	Penjelasan Tambahan Spesifikasi Tugas.....	4
2.1.	Spesifikasi Skill : Shield.....	4
2.2.	Spesifikasi Skill : Attack Up.....	4
2.3.	Spesifikasi Skill : Critical Hit.....	4
2.4.	Spesifikasi Save dan Load.....	4
3.	Struktur Data	
3.1.	ADT Point.....	5
3.2.	ADT Array Dinamis.....	5
3.3.	ADT Matrix.....	6
3.4.	ADT Mesin Karakter.....	6
3.5.	ADT Mesin Kata.....	6
3.6.	ADT Queue.....	7
3.7.	ADT Stack.....	7
3.8.	ADT List.....	7
3.9.	ADT Graph (Variasi Multilist).....	7
3.10.	ADT Player.....	8
3.11.	ADT Command.....	8
3.12.	ADT Color.....	8
3.13.	ADT Save.....	9
3.14.	ADT Load.....	9
4.	Program Utama.....	9
5.	Algoritma-Algoritma Menarik.....	10
5.1.	Algoritma Undo.....	10
6.	Data Test.....	11
6.1.	Data Test 1.....	11
6.2.	Data Test 2.....	11
6.3.	Data Test 3.....	12
6.4.	Data Test 4.....	12
6.5.	Data Test 5.....	12
6.6.	Data Test 6.....	13
6.7.	Data Test 7.....	14
6.8.	Data Test 8.....	14
6.9.	Data Test 9.....	15
6.10.	Data Test 10.....	15

6.11.	Data Test 11.....	16
6.12.	Data Test 12.....	17
6.13.	Data Test 13.....	18
7.	Test Script.....	18
8.	Pembagian Kerja dalam Kelompok.....	20
9.	Lampiran.....	20
9.1.	Deskripsi Tugas Besar.....	20
9.2.	Notulen Rapat.....	21
9.3.	Log Activity Anggota Kelompok.....	23

1. Ringkasan

Avatar World War Game merupakan game *turn-based strategy* yang mensimulasikan sebuah perang dunia antar 4 negara, yaitu Api, Air, Tanah, dan Udara. Mulanya setiap negara memiliki jumlah tentara tertentu yang terus bertambah. Setiap negara memiliki tujuan untuk berperang dengan negara lain dan menduduki sebuah kota dengan cara menyerang dengan pasukan yang jumlahnya lebih banyak dari penghuni kota. Saat sebuah kota telah diduduki, kota akan menghasilkan pasukan bagi negara yang berhasil mendudukinya. Pemenang adalah negara yang berhasil menduduki seluruh kota yang ada.

Berdasarkan deskripsi game tersebut, kami membuat program game dengan game mechanics yang terdiri dari 9 komponen, yaitu main menu, peta, bangunan, skill, kondisi awal permainan, mekanisme permainan, mekanisme attack, command, dan kondisi akhir.

Laporan mencakup deskripsi umum persoalan game, penjelasan tambahan spesifikasi tugas, penjelasan struktur data yang digunakan, penjelasan program utama, data-data dan script yang digunakan untuk menguji keberjalanan game, dan pembagian tugas serta notulensi rapat dan lampiran lainnya.

Program dibuat menggunakan bahasa C dengan memanfaatkan ADT yang sudah dipelajari pada mata kuliah IF2110 - Algoritma dan Struktur Data. Game ini dimulai dengan menampilkan main menu yang memberi pilihan kepada pemain untuk melakukan load game, start game, atau quit. Setelah permainan dimulai, karena game merupakan game *turn-based strategy*, kedua pemain akan melakukan aksi (command) secara bergiliran. Command yang valid pada game ini adalah *attack*, *skill*, *move*, *level_up*, *undo*, *save*, *end_turn*, dan *exit*. Permainan akan terus berlangsung bergiliran antara pemain 1 dan pemain 2 hingga salah satu pemain kalah atau pemain memutuskan untuk melakukan *exit game*. Hasil kelompok kami, program sudah dapat berjalan sesuai dengan spesifikasi. Bonus yang berhasil dibuat oleh kelompok adalah *skill* : *shield*, *critical hit*, dan *attack up*, *save*, dan *load*.

2. Penjelasan Tambahan Spesifikasi Tugas

2.1. Spesifikasi Skill : Shield

Ketika *skill Shield* (SH) diaktifkan, seluruh bangunan pemain yang mengaktifkan *skill* itu akan memiliki pertahanan selama 2 *turn* lawan. Pertahanan berarti saat bangunan diserang oleh N pasukan, jumlah pasukan di bangunan ini hanya berkurang sebesar $\frac{3}{4} N$ (dibulatkan ke bawah). Jika *skill* digunakan 2 kali berturut-turut, durasi *skill* menjadi nilai maksimumnya. Misalnya pemain 1 mengaktifkan *skill* SH pada *turn* pertamanya yang berarti akan memberi efek pada *turn* ke-1 dan ke-2 lawan, lalu pemain 1 kembali mengaktifkan *skill* SH pada *turn* keduanya yang berarti akan memberi efek pada *turn* ke-2 dan *turn* ke-3 lawan, maka *skill* ini hanya akan bertahan sampai *turn* ke-3 lawan. Skill SH ini didapatkan dengan ketika lawan menyerang bangunan pemain dan jumlah bangunan pemain berkurang 1 menjadi sisa 2.

2.2. Spesifikasi Skill : Attack Up

Ketika *skill Attack Up* (AU) diaktifkan, pertahanan pada bangunan lawan (termasuk karena efek *skill Shield*) tidak akan berpengaruh pada saat pemain melakukan penyerangan, sehingga jumlah pasukan pada bangunan lawan ketika diserang tetap akan berkurang sesuai

STEI- ITB	IF2110-TB-08-03	Halaman 4 dari 24 halaman
Template dokumen ini dan informasi yang dimilikinya adalah milik Sekolah Teknik Elektro dan Informatika ITB dan bersifat rahasia. Dilarang me-reproduksi dokumen ini tanpa diketahui oleh Sekolah Teknik Elektro dan Informatika ITB.		

aturan biasa. *Skill* hanya memberi efek pada 1 kali *turn*, yaitu *turn* saat pemain mengaktifkan *skill*-nya. Pemain mendapatkan *skill* ini jika pemain baru saja menyerang *tower* lawan, dan jumlah *tower* lawan menjadi 3.

2.3. Spesifikasi *Skill* : *Critical Hit*

Ketika *skill Critical Hit* (CH) diaktifkan, pada *turn* tersebut, bangunan yang digunakan tepat selanjutnya untuk menyerang akan efektif 2 kali lipat jumlah pasukan. Misalnya pemain memasukkan jumlah pasukan yang digunakan untuk menyerang sebanyak 20, dengan *skill* ini jumlah pasukan untuk menyerang menjadi 40. Penyerangan dengan jumlah 2 kali pasukan ini hanya berlaku 1 kali. *Skill* CH juga menonaktifkan *skill Shield* dan pertahanan bangunan lawan. Pemain mendapatkan *skill* ini jika lawan baru saja mengaktifkan *skill Extra Turn*.

2.4. Spesifikasi *Save* dan *Load*

Save dilakukan dengan command “SAVE” pada saat *game* berlangsung atau saat pemain hendak keluar dari *game*. Extension *file* eksternal hasil *save* berupa .txt atau .dat. Pemain dapat memasukkan nama *file* hasil *save* yang diinginkan tanpa extension. *Load* dilakukan di awal permainan dan pemain harus memasukkan nama *file* eksternal .txt atau .dat yang ingin di-load tanpa extension.

3. Struktur Data (ADT)

3.1. ADT *Point*

- Sketsa struktur data : struktur POINT terdiri atas nilai X dan Y yang bertipe *float*. *Prototype*-nya terdiri atas konstruktor pembentuk *point* untuk membentuk suatu *point* dan fungsi untuk menuliskan *point* ke layar.
- Persoalan yang diselesaikan : menentukan posisi setiap bangunan pada peta.
- Alasan pemilihan : karena posisi bangunan pada peta berada pada sebuah matriks ukuran tertentu yang tiap elemennya lebih mudah dinyatakan dalam bentuk *point* dengan komponen X (absis) dan Y (ordinat).
- Diimplementasikan sebagai ADT *point* dengan nama *file header* “point.h”.

3.2. ADT *Array Dinamis*

- Sketsa struktur data : terdapat struktur *info_bangunan* yang terdiri atas *name* (*char*), posisi (POINT), *level* (*integer*), dan *tentara* (*integer*). Sedangkan struktur bangunan terdiri atas BI (*info_bangunan*), *MaxEl* (*integer*), dan *Neff* (*integer*). Struktur bangunan ini merupakan representasi *array of info_bangunan* dengan *array* dinamis eksplisit. Primitif yang ada terdiri atas konstruktor pembentuk *array* kosong, destruktur untuk dealokasi memori *array*, selektor indeks, *info* penuh/kosong, baca/tulis dengan *input/output*, dan primitif lain yang lebih spesifik untuk penambahan/pengurangan jumlah *tentara* dan *level-up* bangunan.
- Persoalan yang diselesaikan : informasi seluruh bangunan dapat disimpan dalam *array of info_bangunan* ini dan setiap aksi pengurangan/penambahan *tentara* dan *level-up* bangunan dapat terselesaikan.

STEI- ITB	IF2110-TB-08-03	Halaman 5 dari 24 halaman
Template dokumen ini dan informasi yang dimilikinya adalah milik Sekolah Teknik Elektro dan Informatika ITB dan bersifat rahasia. Dilarang me-reproduksi dokumen ini tanpa diketahui oleh Sekolah Teknik Elektro dan Informatika ITB.		

- Alasan pemilihan : Jumlah bangunan yang ada dalam permainan dapat berubah tergantung *file Configure*, sehingga pemilihan ADT *array* dinamis ini tepat untuk menampung bangunan.
- Diimplementasikan sebagai ADT *array* dengan nama *file header* “array.h”.

3.3. ADT *Matrix*

- Sketsa struktur data : struktur MATRIKS terdiri atas Mem (Eltype, yaitu *integer*), MaxBrs (*integer*), dan MaxKol (*integer*). *Prototype* yang didefinisikan terdiri atas konstruktor pembentuk matriks kosong, destruktur untuk dealokasi memori, selektor, baca/tulis matriks, dan operasi lain seperti untuk menghitung jumlah elemen pada matriks.
- Persoalan yang diselesaikan : representasi tampilan peta permainan pada layar beserta bangunan-bangunan yang ada.
- Alasan pemilihan : peta direpresentasikan memiliki jumlah baris dan kolom tertentu, setiap bangunan juga memiliki titik koordinat yang dapat direpresentasikan sebagai posisi pada baris dan kolom di matriks.
- Diimplementasikan sebagai ADT matriks dengan nama *file header* “matriks.h”.

3.4. ADT Mesin Karakter

- Sketsa struktur data : terdapat sebuah *variable extern* bertipe *char* dengan nama CC. *Prototype* yang ada dalam ADT ini yaitu START() dan ADV().
- Persoalan yang diselesaikan : ADT mesin kata.
- Alasan pemilihan : mesin kata membutuhkan mesin karakter dalam fungsi-fungsi *prototype*-nya.
- Diimplementasikan sebagai ADT mesin karakter dengan nama *file header* “mesinkar_config.h”

3.5. ADT Mesin Kata

- Sketsa struktur data : struktur Kata terdiri atas TabKata dengan tipe data *array of char* dan Length (*integer*). BLANK yang didefinisikan dalam ADT ini ada 2, yaitu ‘ ’ (spasi) dan ‘\n’ (*newline*). *Prototype* yang terdapat pada ADT ini terdiri dari prosedur-prosedur dasa, seperti prosedur untuk mengabaikan *blank*, memulai pembacaan kata, mengakuisisi kata, menyalin kata, menulis kata ke layar, hingga fungsi-fungsi yang spesifik untuk membaca *file configure*, yaitu fungsi untuk konversi kata ke *integer*, kata ke *char*, membaca kata menjadi info_bangunan, membaca kata menjadi *array* bangunan, membaca kata menjadi matriks, dan membaca kata menjadi *graph*.
- Persoalan yang diselesaikan : membaca informasi peta dari *file external configure*, membaca *command* dari interaksi *user*, membaca *state* dari *game* yang sudah disimpan.
- Alasan pemilihan : pembacaan informasi dari *file external* memerlukan akuisisi kata satu per satu dan menerjemahkannya ke dalam bentuk data bangunan, peta matriks, dan relasi. Pembacaan *input command* dari *user* dan *state* dari *game* yang disimpan juga berupa string sehingga perlu digunakan mesin kata.
- Diimplementasikan sebagai ADT mesin kata dengan nama *file header* “mesinkata_config.h”.

STEI- ITB	IF2110-TB-08-03	Halaman 6 dari 24 halaman
Template dokumen ini dan informasi yang dimilikinya adalah milik Sekolah Teknik Elektro dan Informatika ITB dan bersifat rahasia. Dilarang me-reproduksi dokumen ini tanpa diketahui oleh Sekolah Teknik Elektro dan Informatika ITB.		

3.6. ADT *Queue*

- Sketsa struktur data : struktur *QUEUE* terdiri atas *T* sebagai tabel penyimpanan elemen (*char* dengan panjang 10 karakter), *HEAD* (Qaddress bertipe *integer*), *TAIL* (Qaddress bertipe *integer*), dan *MaxQEI* (*integer*). *Prototype* yang terdapat pada ADT ini terdiri dari konstruktor untuk membuat suatu *queue* kosong, destruktorkan untuk dealokasi memori, primitif untuk menambahkan (*add*) dan menghapus (*del*) elemen *queue*, penulisan elemen *head* ke layar, dan pengecekan *queue* kosong/penuh. *Queue* yang dipilih adalah dengan *circular buffer* (versi III).
- Persoalan yang diselesaikan : penyimpanan daftar *skill* masing-masing pemain.
- Alasan pemilihan : *Skill* yang dimiliki pemain direpresentasikan sebagai *queue* yang harus digunakan secara berurutan (*first in first out* / *FIFO*).
- Diimplementasikan sebagai ADT *queue* dengan nama *file header* “*queue.h*”.

3.7. ADT *Stack*

- Sketsa struktur data : *infotype* dari elemen *stack* diberi nama *Sinfotype*. Struktur dari *Sinfotype* terdiri dari *pemain1* (*Player*), *pemain2* (*Player*), *turn* (*integer*), dan *databuild* (Bangunan). Sementara struktur *stack* terdiri atas *T* sebagai tabel penyimpanan *Sinfotype*, dan *TOP* (*Saddress*). *Prototype* dari ADT ini memuat fungsi dan prosedur untuk membuat *stack* kosong dan mengatur kondisi awal permainan, menghapus semua isi *stack*, pengecekan keadaan *stack*, prosedur *Push* (memasukkan *Sinfotype* sebagai *TOP* dari *stack*), prosedur *Pop* (elemen *TOP* dari *stack* dikeluarkan), fungsi dan prosedur untuk *turn handling*.
- Persoalan yang diselesaikan : mekanisme *Undo* dengan menyimpan *game states*.
- Alasan pemilihan : ADT *stack* dapat melakukan *push* setiap kali ada perubahan *game states* dan *pop* ketika dilakukan *Undo*.
- Diimplementasikan sebagai ADT *stack* dengan nama *file header* “*stackt.h*”.

3.8. ADT *List*

- Sketsa struktur data : struktur *list* terdiri atas *address First*, sementara untuk struktur elemennya (*tElmtList*) terdiri atas *info* (dengan tipe data urutan (*integer*)) dan *next* yang bertipe data *address*. *Prototype* dalam ADT ini meliputi kreator *list* kosong, manajemen memori (alokasi dan dealokasi), pengecekan *list* kosong, pencarian elemen dalam *list*, penambahan elemen, dan prosedur khusus berkaitan dengan penambahan tentara.
- Persoalan yang diselesaikan : penyimpanan daftar indeks bangunan yang dimiliki masing-masing pemain.
- Alasan pemilihan : pemain hanya perlu mengetahui indeks bangunan yang dimilikinya, sehingga indeks tersebut disimpan dalam sebuah *list linier*. Selain itu, penggunaan *list linier* juga akan memudahkan saat ada bangunan yang direbut lawan (penambahan dan pengurangan bangunan).
- Diimplementasikan sebagai ADT *list linier* dengan nama *file header* “*listlinier.h*”.

3.9. ADT *Graph* (Variasi *multilist*)

STEI- ITB	IF2110-TB-08-03	Halaman 7 dari 24 halaman
Template dokumen ini dan informasi yang dimilikinya adalah milik Sekolah Teknik Elektro dan Informatika ITB dan bersifat rahasia. Dilarang me-reproduksi dokumen ini tanpa diketahui oleh Sekolah Teknik Elektro dan Informatika ITB.		

- Sketsa struktur data : struktur ElmtGraph terdiri atas infoG (*integer*), NextParent (addrGraph), dan FirstChild (addrGraph2), struktur ElmtGraph2 terdiri atas infoG2 (*integer*) dan NextChild (addrGraph2), dan struktur Graph terdiri atas FirstG (addrGraph). *Prototype* dan primitif pada ADT ini memuat selektor, konstruktor untuk membentuk graph kosong, manajemen memori (alokasi dan dealokasi), pengecekan *graph* kosong dan penuh, prosedur untuk baca dan tulis *graph*, penambahan elemen pada *graph*, proses element *graph*, pencarian, pengecekan relasi, dan fungsi khusus yang berkaitan dengan *command attack* dan *move*.
- Persoalan yang diselesaikan : representasi keterhubungan bangunan pada peta.
- Alasan pemilihan : *graph* adalah ADT yang menggambarkan keterhubungan antar *vertex*-nya sehingga cocok untuk merepresentasikan keterhubungan bangunan.
- Diimplementasikan sebagai ADT *graph* dengan nama *file header* “graph.h”.

3.10. ADT *Player*

- Sketsa struktur data : terdapat 3 struktur dalam ADT ini, yang pertama ShieldFx yang terdiri atas *duration* (*integer*) dan activeSH (*boolean*). Struktur ini digunakan untuk *skill Shield*. Selanjutnya terdapat struktur StatusEffect yang terdiri dari *attackup* (*boolean*), *criticalhit* (*boolean*), ShieldFx (*shield*), dan *extraturn* (*boolean*). Struktur ini berperan untuk memberi keterangan efek apa saja yang sedang berpengaruh pada seorang *player*. Selanjutnya terdapat struktur Player yang terdiri atas *queueskill* (*Queue*), listBangunan (*List*), *color* (Warna), dan FX (StatusEffects). *Prototype* terdiri atas selektor, *conditional check* untuk memeriksa player sudah kalah atau belum, kreator untuk membentuk *player*, fungsi-fungsi untuk FX *shield*, *color handling* untuk mengatur warna pemain, prosedur khusus, seperti prosedur *capture* yang digunakan ketika suatu bangunan lawan berhasil direbut, dan seluruh prosedur dan detektor *skill*.
- Persoalan yang diselesaikan : informasi *queue skill* pemain, *list* bangunan yang dimiliki pemain, warna pemain pada tampilan peta,
- Alasan pemilihan : ADT *player* diperlukan untuk menyimpan segala komponen informasi yang berkaitan dengan pemain tersebut.
- Diimplementasikan sebagai ADT *player* dengan nama *file header* “player.h”.

3.11. ADT *Command*

- Sketsa struktur data : struktur *command* terdiri dari prosedur-prosedur yang berisi perintah *attack*, *move*, *level up*, *end turn*, *save*, dan *exit*.
- Persoalan yang diselesaikan : *command* yang akan di-input oleh *user* selama *game* berlangsung dan dampak jika *command* itu dijalankan.
- Alasan Pemilihan : kode menjadi lebih terstruktur dan ringkas.
- Diimplementasikan sebagai ADT *command* dengan nama *file header* “command.h”.

3.12. ADT *Color*

- Sketsa struktur data : terdapat struktur dengan nama TabColor sebagai *array* untuk menampung warna. Struktur ini terdiri atas *array of* Warna sebagai memori untuk menyimpan

STEI- ITB	IF2110-TB-08-03	Halaman 8 dari 24 halaman
Template dokumen ini dan informasi yang dimilikinya adalah milik Sekolah Teknik Elektro dan Informatika ITB dan bersifat rahasia. Dilarang me-reproduksi dokumen ini tanpa diketahui oleh Sekolah Teknik Elektro dan Informatika ITB.		

elemen dan ColNeff (*integer*) sebagai banyaknya elemen efektif. Terdapat total 6 warna yang didefinisikan, yaitu *red*, *green*, *yellow*, *blue*, *magenta*, dan *cyan* juga sebuah warna normal. Prototype pada ADT ini meliputi prosedur untuk inialisasi palet warna, print palet warna yang tersedia, dan print karakter dengan warna.

- Persoalan yang diselesaikan : tampilan bangunan pada peta berwarna untuk masing-masing pemain.
- Alasan pemilihan : diperlukan warna sebagai tanda kepemilikan gedung untuk kedua pemain
- Diimplementasikan sebagai ADT *color* dengan nama *file header pcolor.h*

3.13. ADT Save

- Sketsa struktur data : Hanya ada satu prosedur untuk save bernama *saveData* yang menerima tiga parameter yakni Sinfotype *state*, Graph relasi, dan Matriks map.
- Persoalan yang diselesaikan : save state
- Alasan pemilihan : diperlukan file luar berupa txt/dat untuk menyimpan semua informasi mengenai state game yang sedang berlangsung
- Diimplementasikan sebagai ADT *save* dengan nama *file header save.h*

3.14. ADT Load

- Sketsa struktur data : Hanya ada satu prosedur untuk load bernama *loadFile* yang menerima tiga parameter yakni Sinfotype *state*, Graph relasi, dan Matriks map.
- Persoalan yang diselesaikan : load file txt dari save state yang sudah ada
- Alasan pemilihan : diperlukan load untuk file luar berupa txt/dat yang menyimpan semua informasi mengenai state game yang sedang berlangsung
- Diimplementasikan sebagai ADT *load* dengan nama *file header load.h*

4. Program Utama

Program utama dengan nama file *MainProgram.c* akan meng-include semua *file header* dari ADT. Program utama pertama akan melakukan *looping* yang pertama untuk menampilkan main menu berupa judul *game*. *Looping* ini akan selesai jika pemain memilih keluar dari permainan. Selanjutnya program akan melakukan *looping* yang kedua, yaitu *looping* untuk menampilkan main menu yang dapat dipilih oleh pemain (*play*, *tutorial*, *credits*, dan *exit*) dan akan meminta *input* masukkan menu dari *user*. Pembacaan *input* dilakukan menggunakan mesin kata. Jika pemain memasukkan *input* "TUTORIAL", akan ditampilkan *tutorial* cara memainkan game. Sementara jika pemain memasukkan *input* "CREDITS", program utama akan memanggil prosedur *Credits()* dari file *assets.c*. Setelah pemain selesai membaca Jika pemain memasukkan input "QUIT", prosedur *Quit()* dari file *assets.c* akan dipanggil dan *looping* pertama dan kedua akan berhenti.

Jika pemain memasukkan input "PLAY", hal pertama yang dilakukan adalah program akan menanyakan terlebih dahulu kepada pemain apakah akan melakukan load game atau tidak? (Y/N). Input ini divalidasi terlebih dahulu. Jika pemain menjawab "Y" yang berarti yes, program utama akan memanggil fungsi *LoadFile(GameState)* untuk memanggil states pada game yang disimpan. Jika pemain menjawab "N" yang berarti no, pemain harus memulai permainan dari awal, yaitu memilih

STEI- ITB	IF2110-TB-08-03	Halaman 9 dari 24 halaman
Template dokumen ini dan informasi yang dimilikinya adalah milik Sekolah Teknik Elektro dan Informatika ITB dan bersifat rahasia. Dilarang me-reproduksi dokumen ini tanpa diketahui oleh Sekolah Teknik Elektro dan Informatika ITB.		

warna untuk masing-masing pemain. Lalu program utama akan memulai untuk membaca file eksternal configure dengan memanfaatkan fungsi dan prosedur dari ADT mesin kata. Setelah selesai, program akan memasukkan satu buah bangunan ke list masing-masing pemain dan melakukan set turn sebagai turn player 1. Lalu program akan memanggil fungsi dari ADT stack untuk membuat sebuah stack kosong dan memulai turn.

Memasuki ke bagian utama permainan, terdapat sebuah *loop* yang akan terus berulang selama belum ada pemain yang kalah atau pemain belum memutuskan untuk keluar (exit) dari permainan. Di awal *loop* ini, terdapat inisialisasi turn yang akan mengatur variabel-variabel pada stack. Setelah itu, terdapat lagi sebuah *loop* untuk sebuah turn yang akan terus berulang hingga pemain memasukkan command “END_TURN” atau “EXIT”. Di awal *loop* ini, dilakukan penambahan jumlah tentara untuk setiap bangunan yang sudah dimiliki pemain. Selanjutnya program akan memanggil fungsi untuk menampilkan peta dengan warna bangunan sesuai warna pemain dan juga menampilkan kondisi *game state*. Lalu program menampilkan daftar *command* yang *valid* melalui prosedur Command() dan meminta *input valid* dari *user* (jika input tidak *valid*, akan pembacaan *input* akan terus diulangi). Pembacaan input dari user ini menggunakan mesin kata. Hal-hal ini akan dilakukan setiap awal turn dan setelah pemain memasukkan suatu *command* yang *valid*.

Jika pemain memasukan command “ATTACK”, program utama akan mengatur stack dan memanggil fungsi ATTACK dari ADT command. Melalui fungsi ini mekanisme attack dijalankan. Jika pemain memasukkan command “LEVEL_UP”, program utama akan mengatur stack lalu memanggil fungsi LEVEL_UP dari ADT command. Jika pemain memasukkan command “SKILL”, program utama akan memanggil fungsi SKILL dari file ADT command. Melalui fungsi ini akan dieksekusi skill yang ada pada head dari queue. Jika pemain memasukkan command “MOVE”, program utama akan mengatur stack dan memanggil fungsi MOVE dari ADT command. Dengan fungsi ini, mekanisme move dijalankan. Jika pemain memasukkan command “UNDO”, program utama akan memanggil fungsi UNDO dari ADT command. Fungsi ini akan melakukan pop stack. Jika pemain memasukkan command “END_TURN”, program utama akan memanggil fungsi END_TURN dari ADT command dan giliran pemain saat itu akan berakhir dan berganti ke giliran lawan. Jika pemain memasukkan command “SAVE”, program utama akan memanggil fungsi SAVE dari ADT command dan save game dilakukan. Jika pemain memasukkan command “EXIT”, program utama akan memanggil fungsi EXIT dari ADT command dan fungsi ini akan menanyakan terlebih dahulu pemain akan melakukan save game atau tidak lalu keluar dari game.

5. Algoritma-Algoritma Menarik

5.1. Algoritma Undo

Algoritma undo didefinisikan sebagai prosedur UNDO pada ADT command

```
void UNDO(Stack *gamestate);
```

Algoritma ini digunakan pada saat program utama dijalankan dan pemain memasukkan command “UNDO”. Alasan algoritma ini disebut menarik karena algoritma ini menggunakan implementasi ADT Stack yang termodifikasi sehingga Stack tidak akan dibiarkan kosong, dalam arti saat permainan baru bermula atau giliran baru saja berganti, maka stack akan selalu berisi 1 buah elemen (tipe Sinfotype) yaitu InfoTop yang menyimpan kondisi permainan yang sedang berlangsung. Saat stack hanya berisi 1 elemen, maka UNDO tidak mengubah stack.

STEI- ITB	IF2110-TB-08-03	Halaman 10 dari 24 halaman
Template dokumen ini dan informasi yang dimilikinya adalah milik Sekolah Teknik Elektro dan Informatika ITB dan bersifat rahasia. Dilarang me-reproduksi dokumen ini tanpa diketahui oleh Sekolah Teknik Elektro dan Informatika ITB.		

6. Data Test

6.1. Data Test 1

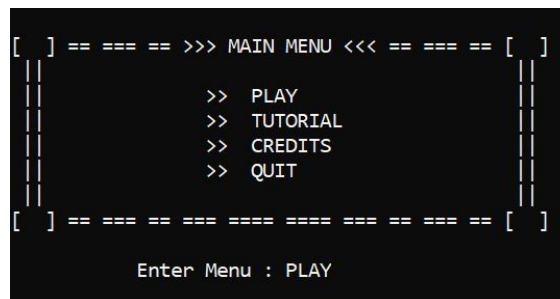
Test ini dilakukan untuk memastikan bahwa program sudah dapat berjalan dan menampilkan tampilan judul game serta main menu. Cara melakukan kompilasi program adalah menggunakan perintah `gcc gcc -o main MainProgram.c array//array.c command//command.c graph//graph.c listlinier//listlinier.c load//load.c matriks//matriks.c pcolor//pcolor.c player//player.c point//point.c queue//queue.c save//save.c stackt//stackt.c mesinkar_config//mesinkar_config.c mesinkata_config//mesinkata_config.c` pada command prompt atau terminal lainnya. Lalu jalankan executable code main tersebut.



Gambar 1. Tampilan Judul dan Main Menu

6.2. Data Test 2

Test ini digunakan untuk memastikan input “PLAY” pada main menu sudah berjalan sesuai.



STEI- ITB	IF2110-TB-08-03	Halaman 11 dari 24 halaman
Template dokumen ini dan informasi yang dimilikinya adalah milik Sekolah Teknik Elektro dan Informatika ITB dan bersifat rahasia. Dilarang me-reproduksi dokumen ini tanpa diketahui oleh Sekolah Teknik Elektro dan Informatika ITB.		

```
Do you want to load a previous game?  
Enter your input (Y/N) : N
```

Gambar 2. Tampilan setelah input “PLAY” dimasukkan

6.3. Data Test 3

Tes ini dimaksudkan untuk menguji fitur load dari game yang sudah pernah disimpan sebelumnya.

```
Do you want to load a previous game?  
Enter your input (Y/N) : Y  
Enter the name of your load file : simpan  
File loaded succesfully!
```

Gambar 3. Load file berhasil

File simpan.txt pada folder savefile sudah tersedia, sehingga load file dapat dilakukan. Namun saat tidak ada save file yang bersesuaian dengan input, maka load akan gagal.

```
Do you want to load a previous game?  
Enter your input (Y/N) : Y  
Enter the name of your load file : simpan  
Load Failed! There seems to be no file named simpan
```

Gambar 4. Load file gagal

6.4. Data Test 4

Test untuk menguji fitur saat game baru dimulai, yaitu fitur pemilihan warna untuk masing-masing pemain.

```
[#]---- Starting A New Game ----[#]  
  
Choose building color for Player 1!  
[ R | G | B | C | M ]  
Choose your color: R  
  
Choose building color for Player 2!  
[ _ | G | B | C | M ]  
Choose your color: G
```

Gambar 5. Pemilihan Warna Kedua Pemain di Awal Permainan

6.5. Data Test 5

Test ini digunakan untuk menguji tampilan peta dan kondisi permainan saat ini.

STEI- ITB	IF2110-TB-08-03	Halaman 12 dari 24 halaman
Template dokumen ini dan informasi yang dimilikinya adalah milik Sekolah Teknik Elektro dan Informatika ITB dan bersifat rahasia. Dilarang me-reproduksi dokumen ini tanpa diketahui oleh Sekolah Teknik Elektro dan Informatika ITB.		

```

[ # ] ---- MAP ---- [ # ]
+++++
+      V  T  C  +
+  C      +
+ T    V    C  +
+  F      +
+      F      +
+  T      +
+      T      +
+  C  V    T  +
+      C      +
+  C T      +
+++++

[ ] ===== Player 1 ===== [ ]

<= Active Effects =>
<> == <> == <> == <> == <> [ ] ===== [ ] == [ ]
[ ] [ ] [ ] [ ] || SKILL || IU ||
<> == <> == <> == <> == <> [ ] ===== [ ] == [ ]

[ ] ===== List of Buildings ===== [P1]
|| - [1.] Castle 40/40 lv. 1 (10, 1)

[ ] ===== >>> COMMANDS <<< ===== [ ]
\\      ATTACK      SKILL      //
||  MOVE  LEVEL_UP  UNDO  ||
//      SAVE  END_TURN  EXIT  \\
[ ] ===== [ ]

Your Command :

```

Gambar 6. Tampilan Peta dan Kondisi Game

6.6. Data Test 6

Test ini dimaksudkan untuk menguji command ATTACK.

```

Your Command : ATTACK

[ ] V ===== V ===== V ===== V ===== V ===== V [ ]

[ ] ===== List of Buildings ===== [P1]
|| - [1.] Castle 40/40 lv. 1 (10, 1)

Choose a building to attack from : 1

[ ] ===== List of Buildings to Attack ===== [P2]
|| - [1.] Castle 40/40 lv. 1(8, 2)
|| - [2.] Tower 30/20 lv. 1(10, 3)

Choose a building you want to attack : 2
Enter your desired amount of soldiers used to attack : 20
You failed to grab the building.

```

Gambar 7. Pemain Memilih Bangunan 2 untuk menyerang.

STEI- ITB	IF2110-TB-08-03	Halaman 13 dari 24 halaman
Template dokumen ini dan informasi yang dimilikinya adalah milik Sekolah Teknik Elektro dan Informatika ITB dan bersifat rahasia. Dilarang me-reproduksi dokumen ini tanpa diketahui oleh Sekolah Teknik Elektro dan Informatika ITB.		

Misalnya pemain memilih bangunan 1 untuk menyerang bangunan 2 dengan 20 pasukan. Tetapi jumlah pasukan yang menyerang belum cukup untuk menjadikan bangunan tersebut milik pemain.

6.7. Data Test 7

Test ini untuk menguji command MOVE.

```

Your Command : MOVE

[ ] V ===== V ===== V ===== V ===== V ===== V [ ]

[ ] ===== List of Buildings ===== [P1]
|| - [1.] Castle 20/40 lv. 1 (10, 1)

Choose a building to move soldiers from: 1

[ ] == List of Nearest Buildings == [P1]
....
It seems that this building does not have any other connected buildings.
Press enter to go back to the command center.
```

Gambar 8. Pemain mencoba melakukan command MOVE

Misalnya pemain akan memindahkan pasukanya dari bangunan 1, tetapi bangunan 1 tidak terkoneksi dengan bangunan lain, sehingga MOVE tidak bisa dilakukan.

6.8. DataTest 8

Test ini untuk menguji command LEVEL_UP

```

Your Command : LEVEL_UP

[ ] V ===== V ===== V ===== V ===== V ===== V [ ]

[ ] ===== List of Buildings ===== [P1]
|| - [1.] Castle 20/40 lv. 1 (10, 1)

Choose the building you want to level-up : 1
Your Castle has been leveled up to level 2!
```

Gambar 9. Pemain Melakukan LEVEL_UP dan Berhasil

Misalnya pemain melakukan LEVEL_UP pada bangunan 1. Karena jumlah pasukannya mencukupi, maka level up berhasil dan jumlah pasukkannya berkurang ½ menjadi 20.

```

Your Command : LEVEL_UP

[] V ===== V ===== V ===== V ===== V ===== V []

[ ] ===== List of Buildings ===== [P1]
|| - [1.] Castle 00/60 lv. 2 (10, 1)

Choose the building you want to level-up : 1
You don't have enough soldiers at your Castle to Level Up the building!

```

Gambar 10. Pemain Melakukan Command LEVEL_UP dan Gagal
Tetapi jika pemain kembali melakukan LEVEL_UP untuk bangunan 1, jumlah tentara yang ada tidak mencukupi sehingga level up gagal.

6.9. Data Test 9

Test digunakan untuk menguji command UNDO.

```

Your Command : UNDO

[] V ===== V ===== V ===== V ===== V ===== V []

You have undone your past action!

[ ] ===== List of Buildings ===== [P1]
|| - [1.] Castle 20/40 lv. 1 (10, 1)

```

Gambar 11. Pemain Berhasil Melakukan UNDO
Pemain sebelumnya telah melakukan command LEVEL_UP terhadap bangunan 1 dan berhasil, kini saat perintah UNDO dimasukkan dan berhasil, bangunan 1 kembali ke level 1.

```

Your Command : UNDO

[] V ===== V ===== V ===== V ===== V ===== V []

You cannot Undo at the moment!

```

Gambar 12. Pemain Tidak Bisa Melakukan UNDO lagi
Tetapi jika pemain kembali melakukan UNDO, command tersebut akan gagal karena sudah tidak ada aksi yang bisa dibatalkan lagi.

6.10. Data Test 10

Test ini digunakan untuk menguji command SKILL.

STEI- ITB	IF2110-TB-08-03	Halaman 15 dari 24 halaman
Template dokumen ini dan informasi yang dimilikinya adalah milik Sekolah Teknik Elektro dan Informatika ITB dan bersifat rahasia. Dilarang me-reproduksi dokumen ini tanpa diketahui oleh Sekolah Teknik Elektro dan Informatika ITB.		


```

[ ] ==== Player 1 ==== [ ]

<= Active Effects =>
<> == <> == <> == <> == <> [ ] ===== [ ] == [ ]
[ ] [ ] [ ] [ ] [ ] [ ] SKILL [ ] IU [ ]
<> == <> == <> == <> == <> [ ] ===== [ ] == [ ]

[ ] ===== List of Buildings ===== [P1]
[ ] - [1.] Castle 40/40 lv. 1 (10, 1)

[ ] ===== >>> COMMANDS <<< ===== [ ]
[ ] ATTACK SKILL [ ]
[ ] MOVE LEVEL_UP UNDO [ ]
[ ] SAVE END_TURN EXIT [ ]

Your Command : SKILL

[ ] V ===== V ===== V ===== V ===== V ===== V [ ]

!!! INSTANT UPGRADE !!!
All your buildings have been Leveled Up!!

```

Gambar 13. Pemain menggunakan SKILL

Pemain memasukkan command SKILL dan skill Instant Upgrade (IU) berada pada antrian queue paling depan, sehingga skill IU yang digunakan.

```

[ ] ==== Player 1 ==== [ ]

<= Active Effects =>
<> == <> == <> == <> == <> [ ] ===== [ ] == [ ]
[ ] [ ] [ ] [ ] [ ] [ ] SKILL [ ] -- [ ]
<> == <> == <> == <> == <> [ ] ===== [ ] == [ ]

[ ] ===== List of Buildings ===== [P1]
[ ] - [1.] Castle 40/60 lv. 2 (10, 1)

[ ] ===== >>> COMMANDS <<< ===== [ ]
[ ] ATTACK SKILL [ ]
[ ] MOVE LEVEL_UP UNDO [ ]
[ ] SAVE END_TURN EXIT [ ]

Your Command : SKILL

[ ] V ===== V ===== V ===== V ===== V ===== V [ ]

Oh no! You don't have any skills!

```

Gambar 14. Pemain Sudah Tidak Memiliki Skill

Ketika pemain sudah tidak memiliki skill apapun, command SKILL tidak akan memberi efek.

6.11. Data Test 11

Test ini digunakan untuk menguji command END_TURN

STEI- ITB	IF2110-TB-08-03	Halaman 16 dari 24 halaman
Template dokumen ini dan informasi yang dimilikinya adalah milik Sekolah Teknik Elektro dan Informatika ITB dan bersifat rahasia. Dilarang me-reproduksi dokumen ini tanpa diketahui oleh Sekolah Teknik Elektro dan Informatika ITB.		


```
Your Command : END_TURN

[ ] V ===== V ===== V ===== V ===== V ===== V [ ]

Changing turns..

[ # ] ---- MAP ---- [ # ]
+++++
+          V   T   C   +
+   C          V   C   +
+ T          F          +
+          T          F   +
+   T          T          +
+   C   V          T   C   +
+   C T          C          +
+++++

[ ] ===== Player 2 ===== [ ]

<= Active Effects =>
<> == <> == <> == <> == <> [ ] ===== [ ] == [ ]
[ ] [ ] [ ] [ ] [ ] [ ] || SKILL || IU ||
<> == <> == <> == <> == <> [ ] ===== [ ] == [ ]

[ ] ===== List of Buildings ===== [ P2 ]
|| - [1.] Castle 40/40 lv. 1 (1, 15)

[ ] ===== >>> COMMANDS <<< ===== [ ]
\\      ATTACK          SKILL          //
||      MOVE          LEVEL_UP          UNDO      ||
//      SAVE          END_TURN          EXIT      \\
[ ] ===== [ ]

Your Command :
```

Gambar 15. END_TURN Berhasil

Command END_TURN yang dimasukkan oleh player 1 akan mengakhiri gilirannya dan mengawali giliran player 2

6.12. Data Test 12

Test digunakan untuk menguji command SAVE

```
Your Command : SAVE

[ ] V ===== V ===== V ===== V ===== V ===== V [ ]

Enter the name of your save file (without .txt/.dat) : datasave
File has been saved succesfully!
```

Gambar 16. SAVE dari permainan

Command SAVE yang dimasukkan oleh player 2 akan membuat datasave.txt pada folder savefiles.

STEI- ITB	IF2110-TB-08-03	Halaman 17 dari 24 halaman
Template dokumen ini dan informasi yang dimilikinya adalah milik Sekolah Teknik Elektro dan Informatika ITB dan bersifat rahasia. Dilarang me-reproduksi dokumen ini tanpa diketahui oleh Sekolah Teknik Elektro dan Informatika ITB.		

6.13. Data Test 13

Test digunakan untuk menguji command EXIT.

```
Your Command : EXIT

[ ] V ===== V ===== V ===== V ===== V ===== V [ ]

Do you want to save the game before exiting the game?
Enter your input (Y/N/C) : N
```

Gambar 17. EXIT dari Permainan

Ketika command EXIT dimasukkan, program akan menanyakan terlebih dahulu pemain ingin melakukan save game atau tidak. Setelah itu program keluar dari permainan dan kembali ke tampilan judul dan main menu.

7. Test Script

Tabel 1. Test Script Fitur pada Program

No.	Fitur yang Dites	Tujuan Testing	Langkah-Langkah Testing	Input Data Test	Hasil yang Diharapkan	Hasil yang Keluar
1	Fitur Start Game	Memeriksa apakah permainan dapat dijalankan	Melakukan <i>compile file</i> “MainProgram.c” di terminal	Data Test 1	Permainan berhasil lalu menampilkan <i>opening message</i> di layar	Sesuai yang diharapkan
2	Fitur Play	Memeriksa apakah permainan dapat dimainkan	Memasukkan <i>input</i> PLAY saat permainan sudah bisa dijalankan	Date Test 2	Permainan berhasil dimainkan lalu mengeluarkan <i>prompt</i> untuk load atau tidak	Sesuai yang diharapkan
3	Fitur Load	Memeriksa apakah permainan dapat me-load file eksternal	Memasukkan <i>input</i> Y jika ingin melakukan load dan N jika tidak.	Data Test 3	Permainan berhasil melakukan <i>load</i> jika filename benar	Sesuai yang diharapkan
4	Fitur Pemilihan Warna	Memeriksa apakah pemain dapat memilih warna di awal permainan game	Memasukkan pilihan warna yang tersedia dimulai dari pemain pertama dan kemudian pemain kedua	Data Test 4	Pemain berhasil memilih warna dan selanjutnya warna yang dipilih sesuai saat game berlangsung	Sesuai yang diharapkan

5	Fitur Peta dan Tampilan	Memeriksa apakah peta dan kondisi game berhasil ditampilkan	Setelah player memilih warna ataupun setelah selesai input command akan ditampilkan peta dan kondisi game terbaru	Data Test 5	Pemain dapat melihat peta serta kondisi game yang terbaru	Sesuai yang diharapkan
6	Fitur Attack	Memeriksa apakah bangunan dapat menyerang bangunan lain	Memasukkan command ATTACK lalu menyerang bangunan yang terhubung dengan bangunan yang dipilih	Data Test 6	Pemain berhasil melakukan attack sesuai dengan aturan permainan	Sesuai yang diharapkan
7	Fitur Move	Memeriksa apakah bangunan dapat melakukan move ke bangunan lain	Memasukkan command MOVE lalu memilih bangunan, bangunan yang terhubung, dan jumlah pasukan yang ingin dipindahkan	Data Test 7	Pemain berhasil melakukan move sesuai dengan aturan permainan	Sesuai yang diharapkan
8	Fitur Level Up	Memeriksa apakah bangunan dapat dinaikkan levelnya	Memasukkan command LEVEL_UP lalu memilih bangunan yang ingin dinaikkan levelnya	Data Test 8	Pemain berhasil melakukan level up sesuai dengan aturan permainan	Sesuai yang diharapkan
9	Fitur Undo	Memeriksa apakah pemain dapat melakukan undo	Memasukkan command UNDO	Data Test 9	Pemain berhasil melakukan undo sesuai dengan aturan permainan	Sesuai yang diharapkan
10	Fitur Skill	Memeriksa apakah pemain dapat menggunakan skill	Memasukkan command SKILL	Data Test 10	Pemain berhasil menggunakan skill sesuai dengan aturan game	Sesuai yang diharapkan
11	Fitur End Turn	Memeriksa apakah player dapat melakukan	Memasukkan command END_TURN	Data Test 11	Pemain berhasil melakukan <i>end turn</i>	Sesuai yang diharapkan

		perintah untuk mengakhiri giliran pemain			sesuai dengan aturan game	
12	Fitur Save	Memeriksa apakah program dapat melakukan save permainan	Memasukkan command SAVE lalu memasukan nama file tempat save	Data Test 12	File berhasil disimpan	Sesuai yang diharapkan
13	Fitur Exit	Memeriksa apakah program dapat menjalankan command EXIT	Memasukkan command EXIT ketika program meminta input command saat permainan berlangsung	Data Test 13	Program menanyakan pemain akan melakukan save game atau tidak lalu keluar dari permainan dan kembali ke tampilan awal judul dan main menu	Sesuai yang diharapkan

8 Pembagian Kerja dalam Kelompok

Tabel 2. Pembagian Kerja Anggota Kelompok

No.	Nama / NIM	Tugas
1.	Muhammad Hasan / 13518012	ADT Command, ADT Save, ADT Load, Fungsi Copy
2.	Anna Elvira H. / 13518045	ADT Queue, ADT Player, beberapa display interface
3.	Daniel Riyanto / 13518075	ADT Point, ADT Array, ADT List, ADT Matriks, ADT Graph
4.	Faris M. Kautsar / 13518105	ADT Mesin Kata, ADT Mesin Karakter
5.	Gregorius Jovan K. / 13510135	ADT Player, ADT Stack, ADT Command, ADT PColor, Main Program, beberapa display interface

9 Lampiran

9.1 Deskripsi Tugas Besar 2

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,

STEI- ITB	IF2110-TB-08-03	Halaman 20 dari 24 halaman
Template dokumen ini dan informasi yang dimilikinya adalah milik Sekolah Teknik Elektro dan Informatika ITB dan bersifat rahasia. Dilarang me-reproduksi dokumen ini tanpa diketahui oleh Sekolah Teknik Elektro dan Informatika ITB.		

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.

9.2 Notulen Rapat

Senin, 21 Oktober 2019

Memahami spesifikasi tugas bersama, melakukan *break-down* per-ADT dan membagi *job-desc* untuk masing-masing anggota kelompok:

Muhammad Hasan : *Command, save, load*

Anna Elvira : *Display Condition, Skill*




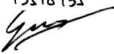

Daniel Riyanto : Peta, Bangunan

Faris M. Kautsar : *Configure*

Gregorius Jovan : ADT *player, main* program






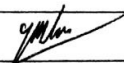
Rabu, 23 Oktober 2019

Asistensi pertama untuk klarifikasi spesifikasi bersama asisten dengan catatan sebagai berikut

Asistensi I		Catatan Asistensi:
Tanggal: 23 Oktober 2019		<p>1.) Boleh tambahkan prosedur sebatas mungkin tetapi ADT-ADT yang ada di spesifikasi harus dikerjakan.</p> <p>2.) File untuk di-save & load bisa sebatas mungkin untuk ekstensinya.</p> <p>3.) ADT Matrix digunakan untuk menggambarkan bangunan ke peta.</p> <p>4.) ADT Array Implisit tidak boleh menyimpan object terdapat hubungan antar bangunan. Yang menyimpan itu ADT Graph.</p> <p>5.) ADT List hanya menyimpan indeks bangunan yang dimiliki oleh player.</p> <p>6.) Saat nge-run program, bangunan-bangunan tidak berubah lagi.</p> <p>7.) Pemain boleh pilih warna.</p> <p>8.) Fungsi untuk mengecek untuk trigger skill itu bebas.</p> <p>9.) Undo itu sampai giliran saat turn pemain itu mulai.</p> <p>10.) Struktur penyimpanan file itu bebas.</p> <p>11.) Boleh membuat ADT Game State yang menyimpan kondisi permainan.</p>
Tempat: Seloas Bosdat		
Kehadiran Anggota Kelompok:		
No	NIM	
Tanda tangan		
1	13518045 	
2	13518075 	
3	13518105 	
4	13518135 	
5		
6		
		Tanda Tangan Asisten: 

Selasa, 19 November 2019

Asistensi kedua untuk menanyakan hal-hal yang masih kurang jelas yang ditemui selama pengerjaan tugas. Catatan yang diberikan selama asistensi :

Asistensi II		Catatan Asistensi:
Tanggal : 19 / 11 / 2019	Tempat : LabPro	
Kehadiran Anggota Kelompok:		1) setelah load game, boleh tidak tetap bisa undo. atau kalau ga bisa undo juga gapapa.
No	NIM	
Tanda tangan		
1	13518012	
		
2	13518045	
		
3	13518075	
		
4	13518005	
		
5	13518135	
		
6		
		Tanda Tangan Asisten: 

Selama asistensi kelompok juga berdiskusi tentang progress masing-masing dan mendaftar fitur apa saja yang masih belum dikerjakan atau yang masih belum sempurna.

9.3 Log Activity Anggota Kelompok

Muhammad Hasan / 13518012

21 Oktober 2019 : Menambahkan spek ke dalam *folder* dokumen *docs*

22 Oktober 2019 : Membuat kerangka kecil pada *file* "command.h" dan "command.c"

23 Oktober 2019 : Menambahkan kerangka pada *command* lalu membuat *file* "save.h", "save.c", "load.h", dan "load.c"

31 Oktober 2019 : Membuat *driver* untuk *command*

STEI- ITB	IF2110-TB-08-03	Halaman 22 dari 24 halaman
Template dokumen ini dan informasi yang dimilikinya adalah milik Sekolah Teknik Elektro dan Informatika ITB dan bersifat rahasia. Dilarang me-reproduksi dokumen ini tanpa diketahui oleh Sekolah Teknik Elektro dan Informatika ITB.		

19 November 2019 : Menambahkan *file* “includes.c”
 20 November 2019 : Membuat ‘mesin kata’ untuk *input* saat *command* berlangsung
 21 November 2019 : Membuat implementasi *move* pada “command.c”
 22 November 2019 : Membuat fungsi *copy* pada *array*, *list*, *player*, dan *state* lalu selesai membuat *save* dengan memodifikasi mesin kata
 23 November 2019 : Membuat *load* dan melengkapi laporan bagian *command*, *save*, dan *load*
 24 November 2019 : Memperbaiki *minor bugs* di *save* dan *load*

Anna Elvira Hartoyo / 13518045

24 Oktober 2019 : Membuat kerangka dasar laporan dan membuat bagian ringkasan
 26 Oktober 2019 : Menyesuaikan ADT *queue* agar nama fungsi tidak bertabrakan dengan fungsi dari ADT lain, menambahkan beberapa fungsi di ADT *queue*, menambahkan beberapa *skill* di ADT *player*, membuat driver *queue*
 9 November 2019 : membuat laporan bagian struktur data
 13 November 2019 : Edit *assets* untuk tampilan
 16 November 2019 : Memperbaiki fungsi di ADT *queue*
 18 November 2019 : Mencoba implementasi *skill* di ADT *command*
 20 November 2019 : Memperbaiki *skill Instant Reinforcement*, *Barrage*, dan menambahkan *detector* untuk *skill Instant Reinforcement*
 21 November 2019 : Memperbaiki tampilan *credits*, *main menu*, membuat tampilan *quit*
 23 November 2019 : Melengkapi laporan (struktur data, spesifikasi tambahan, dan program utama)
 24 November 2019 : Melengkapi laporan bagian data *test* dan *test script*

Daniel Riyanto / 13518075

24 Oktober 2019 : Membuat struktur data 4 ADT yaitu *point*, *array*, *list* dan matriks dan beberapa fungsi/prosedur yang berkaitan dengan 4 ADT tersebut.
 25 Oktober 2019 : Memperbaiki beberapa fungsi/prosedur yang ada di ADT *list*, menambahkan dan memperbaiki beberapa prosedur/fungsi di ADT *array*
 27 Oktober 2019 : Membuat ADT *graph* tetapi belum *graph* murni dan beberapa fungsi/prosedur
 8 November 2019 : Mengubah ADT *array* yang awalnya statis menjadi dinamis dan mengubah beberapa fungsi/prosedur dan memperbaiki suatu prosedur yang ada di ADT matriks
 11 November 2019 : Memperbaiki dan menambahkan suatu fungsi/prosedur di ADT *array* yang berhubungan dengan *Attack*
 18 November 2019 : Mengubah ADT *graph* yang lama menjadi ADT *graph* yang murni tetapi *header*-nya saja
 19 November 2019 : Mengerjakan *file body* ADT *graph*
 20 November 2019 : Mengganti suatu fungsi di ADT *array* yang berhubungan *Attack*
 23 November 2019 : Memperbaiki suatu prosedur di ADT *list*
 24 November 2019 : Memperbaiki suatu prosedur di ADT *array* yang berhubungan dengan *Attack*

Faris M. Kautsar / 13518105

4 November 2019 : Membuat *file* txt untuk konfigurasi permainan.

STEI- ITB	IF2110-TB-08-03	Halaman 23 dari 24 halaman
Template dokumen ini dan informasi yang dimilikinya adalah milik Sekolah Teknik Elektro dan Informatika ITB dan bersifat rahasia. Dilarang me-reproduksi dokumen ini tanpa diketahui oleh Sekolah Teknik Elektro dan Informatika ITB.		

8 November 2019 : Membuat ADT mesin karakter untuk memproses *file* konfigurasi.
 9 November 2019 : Membuat ADT mesin kata untuk memproses *file* konfigurasi.
 11 November 2019 : Membuat *driver* untuk mesin kata konfigurasi.
 19 November 2019 : Memperbaiki fungsi di *graph*, menyelesaikan *driver* mesin kata konfigurasi.
 21 November 2019 : Menambahkan prosedur pencetakan peta ke dalam program utama.
 22 November 2019 : Memperbaiki modul bangunan.
 24 November 2019 : Memisahkan modul ke *folder* masing-masing.

Gregorius Jovan Kresnadi / 13518135

17 Oktober 2019 : Membuat *repository* github
 23 Oktober 2019 : Membuat ADT *player*
 24 Oktober 2019 : Membuat *file* program utama, memodifikasi ADT pcolor
 27 Oktober 2019 : Membenahi tipe-tipe pada ADT *Player*
 9 November 2019 : Mengerjakan program utama
 10 November 2019 : Membuat ADT *stack*, merapikan kode
 13 November 2019 : Melanjutkan ADT *stack*
 18 November 2019 : Mencoba mengatasi masalah di ADT *Stack*
 19 November 2019 : Mengerjakan ADT *skill* dan ADT *command*, memodifikasi ADT pcolor, mengatasi masalah di ADT *stack*
 20 November 2019 : Mengerjakan UI program, melanjutkan ADT *skill*, merapikan *file header*
 21 November 2019 : Mengerjakan ADT *skill* dan *command*, mengerjakan UI program, memperbaiki tampilan peta
 23 November 2019 : Menyelesaikan *command UNDO* dan *ATTACK*, menyelesaikan MainProgram, *debugging*
 24 November 2019 : *Debugging*, mengerjakan laporan

STEI- ITB	IF2110-TB-08-03	Halaman 24 dari 24 halaman
Template dokumen ini dan informasi yang dimilikinya adalah milik Sekolah Teknik Elektro dan Informatika ITB dan bersifat rahasia. Dilarang me-reproduksi dokumen ini tanpa diketahui oleh Sekolah Teknik Elektro dan Informatika ITB.		