

LAPORAN TUGAS BESAR
IF2111/Algoritma dan Struktur Data

**Permainan Mobitangga: Modifikasi Ular Tangga berbasis
Command-Line Interface**

Dipersiapkan oleh:

Kelompok 14

18220017 Gratia Nindyaratri

18220053 I Putu Andika Bagus Jiwanta


18220052 Christopher Jie

18220033 Ayub Seipanya

18220006 Afkar Dhiya Ulhaq

Sekolah Teknik Elektro dan Informatika - Institut Teknologi Bandung

Jl. Ganesha 10, Bandung 40132

	Sekolah Teknik Elektro dan Informatika ITB	Nomor Dokumen		Halaman
		<i>IF2111-TB-14</i>		<i>10</i>
		<i>Revisi</i>	-	<i>28 November 2021</i>

Daftar Isi

Ringkasan	2
Penjelasan Tambahan Spesifikasi Tugas	3
Spesifikasi Fitur Tambahan 1: Support untuk 3 sampai 4 Pemain	3
Struktur Data (ADT)	4
3.1 ADT Array → array.h dan array.c	4
3.2 ADT Mesin Karakter dan Mesin Kata	4
3.3 ADT Stack → stack.h dan stack.c	5
3.4 ADT List → listlinier.h dan listlinier.c	5
3.5 ADT Map → map.h dan map.c	5
3.6 ADT Game → functions.h dan functions.c	6
Program Utama	6
Pembagian Kerja dalam Kelompok	7
Lampiran	7
Deskripsi Tugas Besar	7
Notulen Rapat	8
Log Activity Anggota Kelompok	9

1 Ringkasan

Laporan ini bertujuan untuk menjelaskan serta menyimpulkan hasil kerja yang kelompok kami dapatkan terkait Tugas Besar IF2111/Algoritma dan Struktur Data untuk membuat sebuah permainan berbasis *command-line interface* melalui struktur data dalam bahasa pemrograman C. Permainan yang kami rancang merupakan sebuah permainan yang relatif mirip dengan ular tangga namun dengan beberapa modifikasi melalui fitur-fitur tambahan. Permainan dapat diikuti oleh dua sampai empat peserta pada suatu *virtual board* melalui *command-line*. Setiap peserta memulai permainan dari lokasi awal (petak) yang sama dan berlomba untuk mencapai lokasi atau petak terakhir. Peserta yang pertama kali mencapai petak terakhir akan dideklarasikan sebagai pemenang permainan. Terdapat fitur-fitur menarik yang bisa dimanfaatkan oleh peserta untuk memperoleh kemenangan, antara lain adalah **SKILL**, **BUFF**, dan **TELEPORTER** yang mampu menjadi sarana bagi pemain untuk memudahkan jalan menuju kemenangan.

Laporan ini berisi daftar isi, ringkasan, lampiran, dan penjelasan dari spesifikasi tugas, struktur data, algoritma-algoritma menarik, dan program utama. Ringkasan berisi deskripsi umum persoalan serta kesimpulan yang kami dapat dari proses pengerjaan serta hasil Tugas Besar. Bagian lampiran berisi informasi-informasi detail dari proses pengerjaan tugas besar yang meliputi deskripsi tugas besar, notulen rapat, dan *log activity* kelompok. Bagian penjelasan tambahan spesifikasi Tugas berisi penjelasan terkait fitur-fitur tambahan yang ada di dalam program permainan. Bagian struktur data merupakan interpretasi kami terhadap struktur data yang digunakan menurut sudut pandang praktis dan aplikatif. Bagian algoritma-algoritma menarik merupakan perspektif kami tentang suatu algoritma yang secara subjektif unik dan menarik untuk diulas. Bagian program utama mengandung penjelasan terhadap program utama dari berbagai aspek seperti algoritma program utama, struktur program utama, peran program utama dalam permainan, serta logika dibalik eksistensi program utama yang kami rancang.

Simpulan dari tugas besar yang kami laksanakan adalah kami mendapat berbagai ilmu dan informasi tentang struktur data dengan mencoba membuat program dengan skala relatif besar dan membangun beberapa struktur data baru yang diperlukan untuk melengkapi sebuah program. Dengan bekerja secara kelompok, kami mendapatkan pengalaman untuk bekerja secara kolaboratif dan kontinu secara komunal. Kami tidak hanya mendapatkan pengalaman tambahan dalam menciptakan kode, tetapi juga menciptakan suatu program yang rasional, efisien, tepat guna, serta sesuai dengan apa yang diinstruksikan.

2 Penjelasan Tambahan Spesifikasi Tugas

2.1 Spesifikasi Fitur Tambahan 1: Support untuk 3 sampai 4 Pemain

Fitur ini memungkinkan permainan menerima jumlah pemain maksimal sebanyak 4 pemain. Untuk merealisasikan fitur ini, hal pertama yang dilakukan adalah memasukkan input dari user ke dalam variabel jumlahPemain (*integer*). Jika input jumlahPemain yang diterima bernilai kurang dari 2 atau lebih dari 4, program akan terus meminta user untuk memasukkan input yang sesuai (2-4 pemain) menggunakan *while loop*. Jumlah pemain yang diterima oleh program akan memengaruhi keberjalanan permainan secara relatif komprehensif. Jumlah pemain yang diterima

akan menjadi iterasi fundamental dari keberjalanan *void* dan *infotype* program seperti `printPosisiTiapPemain`, `FirstRound`, `NewRound`, `resetBuff`, `commandMap`, `endGame`, dan `gameLoop`.

3 Struktur Data (ADT)

3.1 ADT Array → `array.h` dan `array.c`

Array merupakan koleksi dari beberapa data dengan tipe yang serupa yang disimpan dalam lokasi dalam memori. ADT Array digunakan untuk menyelesaikan persoalan-persoalan pemrograman yang berkaitan dengan penyimpanan serta manajemen data dalam jumlah yang relatif banyak. Pada tugas besar ini, ADT Array diimplementasikan sebagai sarana untuk membangun ADT lainnya seperti ADT Map. ADT Array kami gunakan sebagai representasi dari tampilan peta dan representasi teleporter yang ada pada ADT Map.

Dalam membangun ADT Array, hal yang pertama dilakukan adalah untuk mendefinisikan indeks minimal, maksimal, dan indeks tak terdefinisi dari *array*. Setelah itu, kami mendefinisikan elemen dan koleksi objek serta mendeklarasikan variabel untuk memori tempat penyimpanan elemen serta variabel banyaknya elemen efektif. Untuk menciptakan tabel yang kosong di awal, kami menciptakan prosedur `MakeEmpty`. Terdapat deklarasi variabel lagi untuk menentukan banyaknya elemen serta mengirimkan banyaknya elemen maksimal yang dapat ditampung oleh tabel. Dalam berinteraksi dengan selektor indeks, kami menciptakan tipe data untuk mengirimkan indeks elemen pertama dan terakhir serta variabel untuk mengirimkan elemen ke-*i* dari tabel. Kami juga menciptakan beberapa prosedur untuk mengubah nilai serta elemen pada tabel. Untuk menentukan kevalidan teks indeks, kami menggunakan `IsIdxValid` (boolean) dan `IsIdxEff` (boolean). Untuk menciptakan *array* yang ideal, dideklarasikan juga `IsEmpty` (boolean) dan `IsFull` (boolean) yang berfungsi untuk mengecek apakah tabel kosong atau tabel penuh.

3.2 ADT Mesin Karakter dan Mesin Kata

ADT Mesin Karakter → `mesinkar.h` dan `mesinkar.c`

Mesin karakter merupakan sebuah mesin imajiner yang mampu menerima serta membaca suatu “pita” berisikan karakter secara prosedural (berurut). ADT Mesin Karakter memiliki tujuan utama untuk *scanning* suatu karakter secara satu persatu yang nantinya akan diaplikasikan pada ADT Mesin Kata. ADT Mesin Kata juga dapat diimplementasikan untuk membaca informasi konfigurasi dari *file* eksternal serta *command* dari interaksi *user* terhadap program.

ADT Mesin Karakter berisi pendefinisian MARK (‘\n’) dan *state* mesin serta beberapa *void* untuk memulai operasi mesin (`void START`) dan memajukan satu karakter dari “pita” (`void ADV`).

ADT Mesin Kata → `mesinkata.h` dan `mesinkata.c`

Mesin kata merupakan sebuah mesin imajiner yang mampu menerima serta membaca suatu rangkaian kata. ADT Mesin Kata merupakan bentuk serta implementasi lebih lanjut dari ADT

Mesin Karakter. ADT Mesin Kata dapat digunakan untuk membaca informasi konfigurasi dari *file* eksternal serta *command* dari interaksi *user* terhadap program.

ADT Mesin Kata berisi pendefinisian “*blank*”, *state* mesin kata, dan beberapa prosedur untuk mengabaikan *blank*, mengakuisisi dan menyimpan kata, memulai pemindaian kata, serta menandai akhir dari pemindaian kata.

a. 3.3 ADT Stack → *stack.h* dan *stack.c*

Stack merupakan struktur data linear yang beroperasi dengan suatu urutan tertentu. ADT stack dapat diimplementasikan dalam usaha untuk menciptakan ADT lainnya seperti ADT *Array* dan. Implementasi dari ADT Stack juga dijadikan landasan pada pembuatan fungsi-fungsi yang ada di file ***functions.h*** dan ***functions.c*** untuk diaplikasikan pada file main serta pembuatan fitur *undo*. Pada file ***functions.h***, *stack* diimplementasikan untuk mendefinisikan *Round*, *PlayerNow*, *TopGame*, dan menciptakan kondisi awal game dimana *buff* tidak ada, posisi masih awal, serta *skill* masih kosong.

ADT ini mengandung *gamestate* untuk tiap rondanya. Tiap *gameState* berisi array of *playerState*. *playerState* berisi *position* (posisi pemain), array of integer *buffs*, List *skills* (*skill* pemain yang diimplementasikan menggunakan Listlinier), dan *skillCount* yaitu jumlah *skill* saat ini. Tiap ronde, *InfoTop* stack akan dibuat copynya lalu di *push*, sehingga *state* ronde sebelumnya tidak bisa lagi diakses dengan yang bermain sekarang kecuali menggunakan kode *undo*.

ADT Stack kami berisikan deklarasi infotype, deklarasi *stack* dengan representasi berkaitan dengan *pointer*, definisi akses dengan selektor, serta beberapa fungsi untuk membuat *stack* kosong, pengecekan apakah *stack* kosong, penambahan elemen ke *stack*, serta menghapus elemen dari *stack*.

b. 3.4 ADT List → *listlinier.h* dan *listlinier.c*

List linier adalah sekumpulan elemen bertipe sama yang mempunyai keturunan tertentu, dan setiap elemennya terdiri dari dua bagian, yaitu informasi mengenai elemen nya dan informasi mengenai alamat elemen suksesornya. ADT *List* dapat diimplementasikan dalam upaya untuk menciptakan struktur data lainnya seperti ADT *Stack* serta penyimpanan *skill* pemain.

ADT List kami berisi deklarasi serta pendefinisian *list*, pembuatan list kosong, *address* Alokasi dan *Search*, dan beberapa fungsi fundamental untuk mengisi dan menghapus elemen list, informasi banyaknya elemen dalam list, konkat, serta menyalin list dari *src* ke *dest* secara *value*.

c. 3.5 ADT Map → *map.h* dan *map.c*

ADT Map merupakan sebuah struktur data yang berisikan realisasi dari peta permainan pada program. ADT Map digunakan sebagai jembatan antara program dengan *user* (pemain) terkait *playground* dan lokasi aktual pemain dalam permainan.

ADT Map berisikan pendefinisian *map* yang memiliki unsur konfigurasi peta, *teleporters*, dan maksimal *roll* yang *default*. Setelah *map* terdefinisi, kami menciptakan beberapa *void* untuk mengalokasikan memori (sebesar ukuran peta), inialisasi peta (panjang peta di *set* sebagai N), dealokasi map, menampilkan semua isi map, membaca *file* konfigurasi, dan pengecekan kevalidan peta.

d. 3.6 ADT Game → *functions.c* dan *functions.h*

ADT ini berisi segala informasi penting untuk game : nama masing masing pemain, giliran, ronde, *isRolled* (apakah dadu sudah di roll atau tidak), *isTurnEnd* (penanda akhir gerak), *isCerminUsed*(penanda cermin), *givenSkill* (skill random yang diberikan di awal ronde), *isUndoUsed* dan *undoTries* untuk keperluan undo, *numOfPlayers* (jumlah pemain), dan yang terpenting, *gameStack*, yaitu stack yang menyimpan setiap ronde game.

4 Program Utama

Program utama terdiri dari fungsi inialisasi game, game loop, dan akhir game. Inialisasi game dimulai dengan membaca file konfigurasi, kemudian memasukkan jumlah pemain dan memvalidasi apakah jumlah pemain sesuai range ($2 \leq \text{jumlahPemain} \leq 4$). Setelah berhasil setup jumlah pemain, program menerima masukan nama pemain dengan maksimal 20 karakter. Jika masukan nama pemain lebih dari 20 karakter, program akan meminta user untuk memasukkan kembali nama pemain. Program menerima nama pemain sesuai jumlah pemain yang sudah ditentukan sebelumnya.

Setelah inialisasi game dilakukan, program akan masuk ke game loop. Looping akan terus dilakukan hingga pemenang dari game didapatkan. Di dalam loop game, terdapat loop untuk tiap ronde. Ronde akan berubah ketika semua pemain telah menyelesaikan giliran mereka. Di dalam ronde, terdapat juga loop untuk giliran masing-masing pemain. Dalam satu giliran, setiap pemain akan mendapatkan satu skill random, kecuali jika jumlah skill milik pemain sudah mencapai batas maksimal. Pemain dapat memberikan beberapa command dalam satu giliran.

Command yang dapat diberikan oleh pemain dalam game loop adalah SKILL, MAP, BUFF, INSPECT, ROLL, ENDTURN, dan UNDO. Pada command SKILL, program akan menampilkan skill yang dimiliki oleh pemain dalam giliran. Kemudian, pemain akan diberikan pilihan untuk menggunakan skill, menghapus skill, atau keluar dari command skill. Command SKILL hanya boleh digunakan jika pemain belum menggunakan command ROLL. Pada command MAP, program akan menampilkan map dan posisi masing-masing pemain di saat itu. Pada command BUFF, program akan menampilkan semua buff dan status apakah buff menyala atau mati. Pada command INSPECT, pemain dapat memasukkan angka yang ingin diketahui apakah dia petak terlarang, petak kosong, atau teleporter. Pada command ROLL, pemain akan mendapatkan hasil dadu random. Hasil dadu dapat dipengaruhi jika pemain menyalakan buff pembesar hoki atau pengecil hoki. Setelah melakukan roll, program akan mengecek apakah pemain sudah mencapai akhir peta. Jika ya, maka pemenang didapatkan dan game berakhir. Pada command ENDTURN, program akan keluar dari loop turn dan masuk ke giliran pemain selanjutnya. Pada command UNDO, program akan mengembalikan keadaan game menjadi awal turn. Ketika UNDO kembali

dilakukan, program akan kembali ke keadaan game di ronde sebelumnya. UNDO dapat terus dilakukan hingga keadaan game menjadi keadaan awal.

Jika salah satu pemain sudah mencapai posisi akhir, maka program akan mengakhiri game loop dan masuk ke end game. Di akhir game, program akan menampilkan nama pemenang dan peringkat akhir berdasarkan posisi para pemain di akhir game. Setelah menampilkan peringkat, program akan melakukan dealokasi untuk map dan game.

5 Pembagian Kerja dalam Kelompok

No.	Fitur/ADT	NIM Coder	NIM Tester
1	ADT wajib dan adisional	18220053,18220052	18220006,18220053, 18220052
2	Program Utama	18220053, 18220017	18220006,18220017, 18220052, 18220053
3	Fitur Roll, Player, dan Turn	18220052, 18220053	18220052, 18220053
4	Fitur Buff	18220053, 18220033	18220053, 18220033
5	Fitur Skill-List	18220053, 18220033, 18220006	18220053, 18220033
6	Command	18220017, 18220053	18220006, 18220017, 18220053
7	Game Flow	18220053, 18220052	18220052, 18220053
8	Main menu	18220053	18220053, 18220006

6 Lampiran

6.1 Deskripsi Tugas Besar

Buatlah sebuah permainan berbasis CLI (*command-line interface*). Permainan ini dibuat dalam bahasa C dengan menggunakan struktur data yang sudah kalian pelajari di mata kuliah ini. Kalian boleh menggunakan (atau memodifikasi) struktur data yang sudah kalian buat untuk praktikum pada tugas besar ini.

Permainan ini merupakan permainan yang berbasis dari permainan ular tangga. Permainan dilakukan oleh 2 **sampai 4 (Bonus)** pemain di suatu papan satu dimensi yang memiliki panjang N. Pada awal permainan, setiap pemain akan mulai dari petak 1 dan berlomba-lomba untuk mencapai petak N. Saat permainan berlangsung, terdapat teleporter dan skill yang dapat digunakan oleh pemain untuk mencapai tujuan, atau untuk mencegah pemain lain mencapai tujuan. Permainan akan berakhir jika sudah ada satu pemain yang mencapai petak N. Peringkat pemain lain akan dilihat berdasarkan posisi pemain saat pemenang ditemukan.

6.2 Notulen Rapat

Asistensi I

Tanggal : 6 November 2021	Catatan Asistensi: Q: Map ada titik, sedangkan ADT Mesin Kata ada terminate pake titik, apakah ADT boleh dimodif? A: Boleh, ada mesin spesifik untuk baca peta dan baca file input yg savesavean Q: Di pedoman ada harus compile pake gcc 5.4.0, itu ada constraint tertentu? A: constraint gaboleh pake library external, harus pake ADT yang dispesifikasiin di halaman 13 Q: math.h gaboleh? A: boleh, karena bawaan tapi harus ditambah -lm ,script buat compile ditaruh di readme Q: contoh pembagian tugasnya bagaimana? A: bebas, bisa tiap orang ngerjain fitur, yg penting si main ini jalan dlu si consolenya. contoh: https://github.com/JonathanGun/Avatar-World-War-Multiplayer-Strategy-Game Q : skill sama buff itu bisa dijelaskan lagi ngga kak? A : Di awal tu nanti bakal dpet skill random, maksimal 10, ada skill yang kalo di aktifin itu dapet buff. Q : Klo kita mau make immunity teleport, sebelum roll berarti kita harus pake skill pintu ga kemana mana? A : Iya, jdi hrus make dlu (premove) Q : Klo chance gimana kak ? A : Di tiap giliran bakal giveaway skill, nah itu ada chancesnya gitu. Q : petak terlarang bisa dilewati? A : bisa dilewati, cuma gabisa berhenti di situ Q : apa pemain punya kebebasan untuk tidak bergerak? A : bisa pilih maju atau mundur, kecuali dua duanya terlarang jadi terpaksa diam Q : eh yg ditanya tadi apaan ga denger T_T Q : teleporter cuma sekali dalam satu move ya kak? A : misal di petak keluar ada teleport masuk, itu ga aktif. ga berantai teleportnya Q : Bonus itu seberapa besar poinnya kak A : Kecil sih, palingan 1/2 dari effort ngerjain yang biasa. Github jangan lupa sebelum push harus pull dulu, jadi langkahnya pull > edit > commit > push
Tempat : Google Meet	
Kehadiran Anggota Kelompok: No NIM Tanda tangan	
2. 18220017 Gratia Nindyaratri	
3. 18220053 I Putu Andika Bagas Jiwanta	
4. 18220052 Christopher Jie	
5. 18220033 Ayub Seipanya	
6. 18220006 Afkar Dhiya Ulhaq	
	Tanda Tangan Asisten:

Asistensi II

Tanggal : 16 November 2021	Catatan Asistensi: Q : bagian turn apa boleh pake queue?
Tempat : Google Meet	

Kehadiran Anggota Kelompok: No NIM Tanda tangan		A : boleh Q : kan ada 4 buff, berarti buat per prosedur? A : kalo buff perlu nunggu yang lain, ada 4 variabel boolean, misal abis jalan ada buffnya, hasilnya bakal beda. misal if buffnya aktif nanti cek dulu bagian buffnya. tidak bisa jadi satu file sendiri, masuk ke dalam logiknya. harus jadi dulu mekanisme gamenya. Q : kan ada command buff, bagaimana? A : command buff bisa dibuat sekarang, efeknya perlu nunggu yang lain. bisa buat perbedaan jika buffnya aktif atau tidak. bisa pakai if, tapi dikosongkan dulu. buff akan hilang setelah dipakai. Q : di daftar adt yang digunakan ada 4 poin, masing-masing ada sub-poin, apakah itu wajib atau opsional? A : daftar adt harus dipakai (peta harus pakai array, dll.). jika dibutuhkan adt lain, boleh ditambahkan. Q : command A : bisa ditulis di main.c trs while not exit bakal nerima input terus, pake adt mesinkar dan mesinkata. Q : kan kita diberi beberapa set adt, sejauh mana boleh diubah adtnya? A : tidak diberitahu seberapa banyak mengubahnya, yang penting variabel penting dalam struct jgn banyak diubah.
1.	18220017 Gratia Nindyaratri	
2.	18220053 I Putu Andika Bagas Jiwanta	
3.	18220052 Christopher Jie	
4.	18220033 Ayub Seipanya	
5.	18220006 Afkar Dhiya Ulhaq	
		Tanda Tangan Asisten:

6.3 Log Activity Anggota Kelompok

No	Hari	Tanggal	Kegiatan	Jam Kerja
1	Sabtu	6 November 2021	Asistensi 1	13:00
2	Senin-Selasa	8-16 November 2021	Mulai membuat prototipe dari fitur-fitur	asinkron
3	Selasa	16 November 2021	Asistensi 2	20:00

4	Rabu-Kamis	17-25 November 2021	Menggabungkan dan pengubahan prototipe jadi kode utuh	asinkron
5	Jumat	26 November 2021	Meeting kerja kelompok	20:00