

# **LAPORAN TUGAS BESAR**

## **IF2111/Algoritma dan Struktur Data**

### **PERMAINAN 'SNEK AND MADDER'**

Dipersiapkan oleh:

Kelompok 12

18220008 Zhillan Attarizal Rezyarifin

18220026 Annel Rashka Perdana


18220039 Alvito Rizqi Sobri

18220073 Umar Hakim

18220087 Mochammad Ramadhany

Sekolah Teknik Elektro dan Informatika - Institut Teknologi Bandung

Jl. Ganesha 10, Bandung 40132

	<b>Sekolah Teknik Elektro dan Informatika ITB</b>	<b>1</b>		<b>Halaman</b>
		<i>IF2111-TB-12</i>		<b>23</b>
		<i>Revisi</i>	<b>1</b>	<i>2021-11-28</i>

# Daftar Isi

<b>Ringkasan</b>	<b>3</b>
<b>Penjelasan Tambahan Spesifikasi Tugas</b>	<b>4</b>
(Bonus) Permainan $\leq 4$ orang	4
(Bonus) Mesin waktu	5
(Bonus) Baling-Baling Jambu	5
<b>Struktur Data (ADT)</b>	<b>5</b>
ADT player	5
ADT array_state	5
ADT array_map	6
ADT array_tp	6
ADT mesin_kar	6
ADT mesin_kata	6
ADT array_buff	6
ADT list	7
ADT stack	7
<b>Program Utama</b>	<b>7</b>
Main Menu	7
Inisialisasi Data Player	8
Pembacaan dan Inisialisasi File Konfigurasi	8
Loop Game dan Pemanggilan Command	8
Penambahan Skill	9
Command “SKILL”	9
SKILL “Pintu Ga Kemana Saja”	9
SKILL “Cermin Pengganda”	9
SKILL “Senter Pembesar Hoki”	9
SKILL “Senter Pengecil Hoki”	10
SKILL “Mesin Penukar”	10
SKILL Bonus “Mesin Waktu”	10
SKILL Bonus “Baling-Baling Jambu”	10
Command MAP	10
Command BUFF	10

Command INSPECT	11
Command ROLL	11
Command UNDO	11
Command ENDTURN	11
<b>Algoritma-Algoritma Menarik</b>	<b>11</b>
Validasi Input Integer	11
<b>Pembagian Kerja dalam Kelompok</b>	<b>11</b>
<b>Lampiran</b>	<b>12</b>
Deskripsi Tugas Besar	12
Notulen Rapat	12
Log Activity Anggota Kelompok	17
Form Asistensi	20

# 1 Ringkasan

Borakemon dan Mobita bekerja sama untuk memenangkan sebuah lomba *game dev* dengan membuat permainan yang menggabungkan permainan ular tangga lalu memodifikasi fitur-fitur dalam permainan tersebut. Permainan itu mereka sebut Mobitangga. Tetapi, Mobita tidak memiliki kemampuan memprogram permainan Mobitangga. Begitu juga dengan Borakemon. Akhirnya, Borakemon menculik beberapa *programmer* handal untuk membuat Mobitangga.

Mobitangga adalah sebuah *game* berbasis CLI (*command-line interface*) yang merupakan modifikasi dari permainan ular tangga dengan tambahan fitur-fitur yang dapat mengganggu pemain lain. Mobitangga dapat dimainkan oleh 2 sampai 4 pemain di atas papan satu dimensi dengan panjang N. Permainan diawali dengan para pemain bergerak dari petak 1 dan akan berakhir jika salah satu pemain berhasil mencapai petak N. Istilah penting dalam game ini ada dua. Pertama, Round yang berarti putaran permainan. Kedua, Turn yang berarti pemain dapat melakukan aksi. Ada tujuh *game mechanics* dalam program yang telah dibuat yaitu Main Menu, Peta, Roll, Teleporter, Skill, Buff, dan Command.

Laporan ini berisi tentang penjelasan program-program yang telah kami buat. Pada bagian pertama berisi tentang ringkasan yang di dalamnya terdapat deskripsi umum persoalan dan penjelasan singkat laporan. Bagian kedua berisi tentang penjelasan mengenai tambahan spesifikasi fitur yang terdapat pada Spesifikasi Tugas Besar. Bagian ketiga berisi tentang struktur data-struktur data (ADT) yang digunakan oleh kelompok kami. Bagian keempat berisi tentang penjelasan-penjelasan mengenai program utama. Bagian terakhir, kelima, berisi tentang algoritma-algoritma menarik yang kami gunakan atau temukan dalam proses pembuatan Tugas Besar.

Permainan ini dibuat memakai bahasa C dengan menggunakan ADT yang sudah dipelajari di mata kuliah IF2111 Algoritma dan Struktur Data. Beberapa ADT yang digunakan yaitu ADT player, ADT array\_state, ADT array\_map, ADT array\_tp, ADT mesin\_kar, ADT mesin\_kata, ADT array\_buff, ADT list, dan ADT stack.

## 2 Penjelasan Tambahan Spesifikasi Tugas

### 2.1 (Bonus) Permainan $\leq 4$ orang

Pada program ini, kami menggunakan ADT state (dengan tipe elemen ADT players) untuk menyimpan data pemain. Urutan pemain dideklarasikan secara implisit lewat indeks di array tersebut. Giliran player ke-berapa dapat dicatat dengan menggunakan suatu variabel untuk counter. Dengan demikian, dapat dibuat variabel-variabel berisi data tiap pemain dengan jumlah beragam dengan menggunakan loop dan tanpa menggunakan kondisional yang rumit.

## 2.2 (Bonus) Mesin waktu

Pada program ini, kami menggunakan ADT state (dengan tipe elemen ADT players) dari *instance* yang sedang terjadi untuk dapat mengupdate data seluruh player pada round tersebut. Kami juga menggunakan ADT array\_tp dan ADT array\_map. Mesin waktu memajukan player lawan yang dipilih sebanyak beberapa langkah dari 1 hingga MaxRoll yang tidak diketahui, yang baru akan ditentukan secara random setelah skill dipilih. Player kemudian memilih “korban” dan mengaktifkan skill, yang akan dijalankan hanya jika hasil pergerakan tidak membuat korban berakhir di luar map atau di petak terlarang. Jika gagal, skill tidak jadi dipilih. Memungkinkan terjadi teleport.

## 2.3 (Bonus) Baling-Baling Jambu

Pada program ini, kami menggunakan ADT state (dengan tipe elemen ADT players) dari *instance* yang sedang terjadi untuk dapat mengupdate data seluruh player pada round tersebut. Kami juga menggunakan ADT array\_tp dan ADT array\_map. Baling-Baling Jambu memundurkan player lawan yang dipilih sebanyak beberapa langkah dari 1 hingga MaxRoll yang tidak diketahui, yang baru akan ditentukan secara random setelah skill dipilih. Player kemudian memilih “korban” dan mengaktifkan skill, yang akan dijalankan hanya jika hasil pergerakan tidak membuat korban berakhir di luar map atau di petak terlarang. Jika gagal, skill tidak jadi dipilih. Memungkinkan terjadi teleport.

# 3 Struktur Data (ADT)

## 3.1 ADT player

ADT player digunakan untuk menyelesaikan masalah terkait player atau pemain. ADT player terdiri atas informasi player yang berupa nama player yang bertipe char, current\_petak bertipe integer yang merepresentasikan petak player berada, list skill yang dimiliki player, dan buff yang tersimpan dalam array\_buff. ADT player diperlukan untuk menyimpan informasi player selama permainan berlangsung.

## 3.2 ADT array\_state

ADT *array\_state* atau ADT state digunakan untuk menyelesaikan masalah terkait status permainan. ADT state terdiri atas array dengan panjang maksimal 100 dan menyimpan informasi player (diambil dari ADT player), nilai efektif, dan *integer round*. Nilai efektif disini merepresentasikan jumlah pemain yang bermain, sedangkan round merepresentasikan ronde *state* permainan yang bersangkutan. ADT state diperlukan agar semua data permainan setiap rondonya tersentralisasi yang nantinya akan mempermudah proses implementasi *command undo*. ADT state dideklarasikan di state.h, diimplementasikan di state.c, dan diuji di mstate.c

### 3.3 ADT array\_map

ADT array untuk map digunakan untuk menyelesaikan masalah terkait Map. Array map merupakan array dengan panjang maksimalnya 100 dan menyimpan informasi Petak yang merupakan karakter titik (.), pagar (#), dan *minus*(-) dengan titik berarti petak dapat ditempati pemain, pagar untuk tidak dapat ditempati pemain, dan *minus* untuk petak yang kosong (tidak digunakan). ADT array\_map diperlukan untuk mempermudah dan mempersingkat pemanggilan yang berulang setiap ronde atau giliran pemain, seperti memeriksa apakah petak ke-X merupakan petak terlarang atau tidak atau menampilkan peta untuk setiap pemain. ADT array\_map dideklarasikan di array\_map.h, diimplementasikan di array\_map.c, dan diuji di marray\_map.c.

### 3.4 ADT array\_tp

ADT array untuk teleportasi digunakan untuk menyelesaikan masalah terkait Teleport yang ada di petak peta. Array teleportasi merupakan array yang panjang maksimalnya 99 dan menyimpan informasi dengan indeks pada array menyatakan “pintu masuk” teleport dan nilai yang ada di indeks tersebut adalah “pintu keluar”-nya. Array teleportasi yang kosong ditandai dengan seluruh elemennya merupakan angka nol atau dengan kata lain setiap petak tidak menuju kemanapun. ADT array\_tp diperlukan untuk lebih mudah dalam pengidentifikasian teleport yang ada. ADT array\_tp dideklarasikan di array\_tp.h, diimplementasikan di array\_tp.c, dan diuji di marray\_tp.c.

### 3.5 ADT mesin\_kar

ADT mesin\_kar digunakan untuk menyelesaikan masalah terkait pembacaan file konfigurasi. Mesin karakter akan membaca “config.txt” untuk selanjutnya dibaca per karakternya dan disimpan di variabel CC dengan pemberhentian ditandai dengan mark berupa koma(.). ADT ini diperlukan untuk membaca setiap karakter yang ada di file konfigurasi untuk selanjutnya diproses dengan ADT lain. ADT mesin\_kar dideklarasikan di mesin\_kar.h, diimplementasikan di mesin\_kar.c, dan diuji bersama ADT mesin\_kata di mmesin\_kata.c.

### 3.6 ADT mesin\_kata

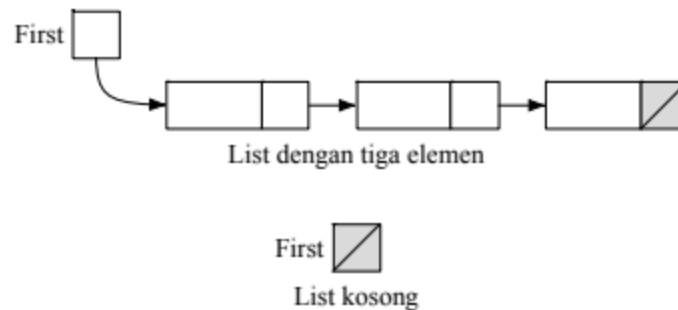
ADT mesin\_kata digunakan untuk menyelesaikan masalah terkait pembacaan file konfigurasi. Mesin kata akan menyimpan kata yang dibaca dalam tipe bentukan Kata yang memiliki komponen array dan panjang dari kata yang dibaca Mesin kata bersama mesin karakter diperlukan untuk membaca “config.txt” dengan pembacaan per kata dan disimpan ke sebuah array dengan tujuan isi array tersebut dapat selanjutnya disimpan ke variabel tertentu. ADT mesin\_kata dideklarasikan di mesin\_kata.h, diimplementasikan di mesin\_kata.c, dan diuji bersama ADT mesin\_kar di mmesin\_kata.c.

### 3.7 ADT array\_buff

ADT array\_buff digunakan untuk menyelesaikan masalah terkait buff yang dimiliki oleh pemain. ADT ini diperlukan untuk set awal nama-nama buff, kemudian display buff yang dimiliki pemain ke layar.

### 3.8 ADT list

Digunakan ADT list linier untuk menyelesaikan masalah Skill. List linier terdiri dari pointer First yang menunjuk ke address Node pertama dan Node-Node yang terdiri dari info dan pointer ke Node berikutnya, dengan Node terakhir “menunjuk” ke Nil.



List linier dipilih karena lebih cocok untuk suatu struktur data yang harus dapat dengan cepat menambahkan atau menghapus elemen dengan indeks yang dapat acak. Dengan menggunakan list linier, kita tidak perlu memusingkan pemindah-mindahan data atau nilai null yang dapat dihasilkan pada operasi Skill jika kita menggunakan array. Operasi search Skill juga cukup sederhana, sehingga tidak memerlukan ADT linier yang lebih kompleks. ADT list dideklarasikan di `list.h`, diimplementasikan di `list.c`, dan diuji di `mlist.c`. Setiap node mengandung tiga informasi: next address ke node berikutnya, info yang berisi “nomor id” skill, dan value yang dikandung untuk keperluan bonus.

### 3.9 ADT stack

ADT *stack* atau ADT *stack\_state* digunakan untuk menyelesaikan masalah terkait perubahan status dalam permainan. Informasi yang disimpan di ADT *stack* merupakan status permainan (diambil dari *array\_state*). ADT *stack* diperlukan untuk menyimpan status-status permainan setiap ronde supaya status ronde-ronde sebelumnya dapat diakses. ADT *stack\_state* dideklarasikan di `stack_state.h`, diimplementasikan di `stack_state.c`, dan diuji di `mstack_state.c`.

## 4 Program Utama

### 4.1 Main Menu

Pada awal permainan, pemain dapat memilih untuk bermain atau keluar dari game. Main Menu berisi tiga *command* yaitu New Game, Exit, dan Help. Saat memilih New Game, program akan meminta input nama file konfigurasi, jumlah pemain yang bermain, dan nama masing-masing dari pemain. Lalu, permainan baru akan dimulai. Exit berfungsi untuk menutup program dan keluar dari permainan. Help berisi tentang cara memulai permainan dan list dari command saat permainan berlangsung.

## 4.2 Inisialisasi Data Player

Saat memilih New Game di Main Menu, program akan meminta beberapa data yang perlu diisi sebelum memulai permainan seperti nama file, jumlah player, dan nama masing-masing player. Data-data tersebut akan tersimpan dalam variabel *currentState* dan akan tersimpan dalam stack setiap round berganti. *currentState* berisi *TabPlayer* yang merupakan memori penyimpanan elemen player, banyaknya elemen efektif (*Neff*), dan round permainan (*Round*). *TabPlayer* menyimpan informasi pemain seperti nama, *current\_petak*, *skill*, dan *buff*. Awalnya data *Neff* dan *Round* akan diinisialisasi dengan fungsi *SetNeff* dan *SetRound*. Data yang akan diinisialisasi dengan *SetNeff* adalah jumlah pemain. Setelah itu, prosedur *insert\_players* akan meminta nama dari tiap player dan menyimpannya di *TabPlayer*. Petak para player juga menjadi 1 serta terbentuk list dan array kosong untuk menyimpan *skill* dan *buff* player selama permainan. Lalu, value dari *SetRound* akan berubah menjadi 1 karena permainan akan dimulai.

## 4.3 Pembacaan dan Inisialisasi File Konfigurasi

Pembacaan dan inisialisasi file konfigurasi akan dilakukan dengan menggunakan prosedur *ReadConfigFile* yang ada di file console dengan memanfaatkan ADT *mesin\_kar*, *mesin\_kata*, *array\_tp*, dan *array\_map*. Untuk setiap karakter yang berupa angka pada file konfigurasi akan diolah terlebih dahulu menjadi tipe *integer* menggunakan fungsi *strToInt* dan *charToInt* saat akan disimpan ke suatu variabel. Pertama, akan dilakukan pembacaan file konfigurasi yang *line* pertamanya merupakan panjang dari peta dan disimpan ke sebuah variabel. Kedua, dilanjutkan dengan pembacaan *line* selanjutnya yang merupakan peta yang sudah dibuat dan disimpan ke sebuah array peta menggunakan prosedur *KataToTabPeta* dengan terlebih dahulu membuat petanya menggunakan prosedur *MakeEmptyPeta*. Ketiga, pembacaan *line* selanjutnya merupakan nilai terbesar dari dadu yang bisa didapatkan pemain dan kemudian disimpan ke sebuah variabel. Keempat, *line* berikutnya merupakan informasi dari banyaknya teleportasi dan nilai itu kemudian disimpan ke sebuah variabel. Terakhir, dilakukan *loop* sebanyak teleportasi yang ada dan menyimpan informasi terkait “pintu masuk” dan “pintu keluar” teleportasi ke sebuah array teleportasi dengan terlebih dahulu membuat array teleportasi menggunakan prosedur *MakeEmptyArrTP*.

## 4.4 Loop Game dan Pemanggilan Command

*Game* akan terus berlangsung selama belum ada pemain yang sampai ke petak ujung peta. Informasi itu dapat diketahui melalui variabel *EndGame* yang bertipe *boolean* dengan nilai *true* menandakan sudah ada pemain yang sampai ke petak ujung dan *false* jika tidak ada. lalu akan dimulai ronde baru yang diawali dengan giliran pemain pertama serta untuk setiap awal ronde akan ditampilkan peta setiap pemain secara terpisah menggunakan prosedur *DisplayPetaPemain*. Selanjutnya akan dimulai giliran untuk setiap pemain secara berurutan hingga semuanya sudah menyelesaikan gilirannya, salah satu pemain memanggil *command* UNDO, atau ada pemain yang sudah sampai ke ujung petak dari peta. Pada setiap gilirannya, pemain dapat memanggil *command* SKILL, BUFF, INSPECT, UNDO, ROLL, atau ENDTURN. Jika belum ada pemanggilan *command* ROLL maka *command* yang dapat digunakan adalah SKILL, BUFF,



INSPECT, dan UNDO. Namun, jika *command* ROLL sudah dipanggil, *command* yang akan dapat digunakan hanyalah BUFF, INSPECT, UNDO, dan ENDTURN. Keadaan tersebut akan terus berulang hingga pemain yang sedang giliran menggunakan *command* ENDTURN. Jika EndGame bernilai *true* maka permainan akan selesai dan akan ditampilkan pemenang, serta peringkat untuk tiap pemain lainnya.

#### **4.5 Penambahan Skill**

Penambahan skill dilakukan lewat fungsi *gacha\_skill*. Pertama, dilakukan pengecekan apakah kapasitas skill masih ada atau tidak. Kemudian dilakukan roll ([1..100]) dengan seed dari clock, dan player akan mendapatkan skill tergantung hasil proses roll tersebut. Besar maju atau mundur mesin waktu juga di-roll ([1..MaxRoll]) dan disimpan di value di slot list, sementara value skill selain kedua hal itu diisi VALUE\_UNDEF.

#### **4.6 Command “SKILL”**

Command ini digunakan untuk menunjukan menu skill yang berisi skill - skill apa saja yang dimiliki oleh pemain yang memakai command “SKILL”

#### **4.7 SKILL “Pintu Ga Kemana Saja”**

Skill “Pintu Ga Kemana Saja” merupakan skill yang menambahkan buff “imunitas teleport” ke pemain yang menggunakan skill ini sehingga ketika pemain yang menginjak teleporter mengaktifkan skill ini, maka pemain dapat memilih untuk tidak kemana-mana. Saat awal pemanggilan skill ini, akan divalidasi apakah pemain sudah memiliki buff “imunitas teleport”. Jika sudah maka akan dicetak kelayar bahwa pemain sudah memiliki buff “imunitas teleport”. Jika belum maka buff “imunitas teleport” akan ditambahkan ke array buff pemain, dan Skill “Pintu Ga Kemana Saja” akan dihapus (-1) dari array skill pemain

#### **4.8 SKILL “Cermin Pengganda”**

Skill “ Cermin Pengganda” pada dasarnya merupakan skill “ buang 1 dapat 2”. Jadi ketika pemain menggunakan skill ini, maka skill cermin pengganda akan di buang dari array skill pemain, lalu pemain akan mendapat 2 skill baru sesuai presentase yang sudah ditentukan. Selain itu, ketika pemain mengaktifkan skill ini, pemain akan mendapatkan buff “cermin pengganda” yang menandakan bahwa skill “cermin pengganda” sudah digunakan dalam ronde tersebut. Ketika pemain sudah memiliki buff “cermin pengganda” di ronde tertentu, maka pemain tersebut tidak bisa menggunakan skill “ cermin pengganda”.

#### **4.9 SKILL “Senter Pembesar Hoki”**

Skill “ Senter Pembesar Hoki” merupakan skill yang memberikan buff “ Senter Pembesar Hoki” ke pemain yang menggunakan skill tersebut. Buff ini mengubah hasil roll menjadi antara floor(MaxRoll/2) dan MaxRoll. Ketika pemain sudah memiliki buff “Senter Pembesar Hoki”, Pemain tidak bisa menggunakan skill “ Senter Pembesar Hoki” .

#### **4.10 SKILL “Senter Pengecil Hoki”**

Skill “Senter Pengecil Hoki” pada dasarnya sama dengan skill “Senter Pembesar Hoki”, sama-sama mengubah hasil roll, tetapi Skill “Senter Pengecil Hoki” mengubah hasil roll menjadi 1 dan floor(MaxRoll/2). Skill ini juga akan menambahkan buff “Senter Pengecil Hoki” ke pemain sehingga pemain yang memiliki buff “Senter Pengecil Hoki” tidak bisa menggunakan skill ini.

#### **4.11 SKILL “Mesin Penukar”**

Skill “Mesin Penukar” merupakan skill yang mengakibatkan pemain bisa menukar posisinya saat ini dengan posisi pemain lain. Penukaran dilakukan dengan mengetahui idx player dan current petak tersebut kemudian current petak saling ditukar

#### **4.12 SKILL Bonus “Mesin Waktu”**

Sebagaimana diterangkan pada 2.2., “Mesin Waktu” memajukan player lawan yang dipilih sebanyak beberapa langkah dari 1 hingga MaxRoll yang tidak diketahui, yang baru akan ditentukan secara random setelah skill dipilih. Player kemudian memilih “korban” dan mengaktifkan skill, yang akan dijalankan hanya jika hasil pergerakan tidak membuat korban berakhir di luar map atau di petak terlarang. Jika gagal, skill tidak jadi dipilih. Memungkinkan terjadi teleport.

#### **4.13 SKILL Bonus “Baling-Baling Jambu”**

Sebagaimana diterangkan pada 2.3., “Baling-Baling Jambu” memundurkan player lawan yang dipilih sebanyak beberapa langkah dari 1 hingga MaxRoll yang tidak diketahui, yang baru akan ditentukan secara random setelah skill dipilih. Player kemudian memilih “korban” dan mengaktifkan skill, yang akan dijalankan hanya jika hasil pergerakan tidak membuat korban berakhir di luar map atau di petak terlarang. Jika gagal, skill tidak jadi dipilih. Memungkinkan terjadi teleport.

#### **4.14 Command MAP**

Command ini akan menampilkan peta untuk setiap pemain sesuai dengan posisinya masing-masing serta terpisah, serta menampilkan pula pada petak berapa pemain itu berada disamping peta.

#### **4.15 Command BUFF**

Command ini akan menampilkan buff apa saja yang dimiliki oleh pemain yang menggunakan command “BUFF” pada gilirannya

#### 4.16 Command *INSPECT*

Command ini membaca peta yang didapat pada awal inisialisasi permainan, kemudian melakukan akses array langsung lewat indeks untuk menentukan isi petak, kemudian mem-print out hasilnya.

#### 4.17 Command *ROLL*

Command ini menerima seluruh data *instance* suatu round yang sedang berlangsung. Kemudian, program “melempar dadu” secara acak dengan dipengaruhi oleh *buff-buff* “senter” yang ada. Setelah itu, program menentukan apakah ada gerakan legal yang dapat dijalankan lewat kondisional, dan menjalankannya. Jika pemain mendarat di petak yang memiliki teleport,

#### 4.18 Command *UNDO*

Command ini akan memundurkan status permainan ke ronde sebelumnya. Jika player melakukan undo saat ronde ke-x, maka status permainan akan berada di akhir ronde ke-(x-1) sehingga ronde ke-x akan dimulai kembali dari player giliran pertama. Jika pada ronde ke-1 player melakukan undo, maka status permainan akan kembali ke sebelum game mulai.

#### 4.19 Command *ENDTURN*

Command ini akan mengakhiri giliran pemain dan hanya dapat dilakukan jika pemain yang bersangkutan sudah melakukan *roll*.

## 5 Algoritma-Algoritma Menarik

### 5.1 Validasi Input Integer

Tidak seperti bahasa-bahasa pemrograman terbaru, bahasa C cenderung primitif dalam menerima inputan user, apalagi jika menangani perubahan tipe data yang berbeda-beda. Dalam program ini, kami menggunakan gabungan fungsi *scanf* dan fungsi buatan sendiri, *str\_to\_int\_idx0*, untuk memvalidasi inputan integer. Pertimbangannya adalah sistem validasi yang baru ditambahkan di akhir-akhir pengerjaan, juga banyaknya *scanf* yang telah terlanjur digunakan, sehingga tim kami memperkirakan bahwa akan meninggal jika program didesain ulang agar dapat mengakomodasi fungsi penerima input yang sejenis seperti *fgets*. *str\_to\_int\_idx0* mengecek inputan user yang disimpan dalam bentuk string, kemudian mengembalikan integer yang dimasukkan jika inputan tersebut dapat diubah menjadi integer, dan mengembalikan nilai *ILLEGAL* (didefinisikan sebagai -999) jika tidak.

## 6 Pembagian Kerja dalam Kelompok

No.	Fitur/ADT	NIM Coder	NIM Tester
1	ADT player (beserta primitif dan driver)	18220039, 18220008	18220008

STEI- ITB	<i>I</i>	Halaman 11 dari 23 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.		

2	ADT array state (beserta primitif dan driver)	18220039, 18220008	18220087
3	ADT array map (beserta primitif dan driver)	18220073	18220073
3	ADT array tp	18220073	18220073
4	ADT mesin kar (beserta primitif dan driver)	18220073	18220073
5	ADT mesin kata (beserta primitif dan driver)	18220073	18220073
6	ADT array buff (beserta primitif dan driver)	18220026, 18220008	18220008
7	ADT listlinier (beserta primitif dan driver)	18220008	18220087
8	ADT stack (beserta primitif dan driver)	18220087	18220073
9	Main menu	18220039, 18220008	18220008
10	Inisialisasi data player (bonus > 2 player)	18220039, 18220008	18220008
11	Pembacaan dan inisialisasi file konfigurasi	18220073	18220073, 18220008
12	Loop game, pemanggilan command (termasuk end turn), pemutusan loop (akhir game)	18220073	18220073, 18220087
13	Pemanggilan penambahan skill	18220008	18220008
13	Command Skill	18220026, 18220008	18220008
14	Skill "Pintu Ga Kemana Saja", "Cermin Pengganda", "Senter Pembesar Hoki", "Senter Pengecil Hoki", "Mesin Penukar"	18220026, 18220008	18220008
15	Command Buff, Buff ("Imunitas Teleport", "Cermin Pengganda", "Senter Pembesar Hoki", "Senter Pengecil Hoki")	18220026	18220008
16	Skill bonus "Mesin waktu", :Baling-Baling Jambu"	18220008	18220008
17	Command Inspect	18220008	18220008
18	Command Roll	18220008	18220008
19	Command End Turn	18220087	18220087
20	Command Undo	18220087	18220087

## 7 Lampiran

### 7.1 Deskripsi Tugas Besar

Dibuat sebuah permainan berbasis CLI (*command-line interface*). Permainan ini dibuat dalam bahasa C dengan menggunakan struktur data yang sudah dipelajari di mata kuliah ini. Digunakan (atau dimodifikasi) struktur data yang sudah dibuat untuk praktikum pada tugas besar ini.

### 7.2 Notulen Rapat

#### Rapat 2021-11-06

Rapat perdana membahas rencana "kita harus ngapain". Tiap spesifikasi dibahas, kemudian dibuat daftar hal-hal yang harus dibuat atau dilakukan. Didapat hal-hal sebagai berikut:

1. Driver2 dan ADT Array, Mesin Kar, Mesin kata, Stack, List, beserta fungsi2nya yang udah di praktikum, tiap ADT ada drivernya
2. Main menu: bikin
  - a. Printf pilihan2 command (mungkin nanti ada help)
  - b. Scanf main game atau keluar

- c. Scanf NEW GAME
    - i. Jumlah pemain [bisa > 2 karena kita dah bikin ADT Pemain, itu bisa dimasukkan array of Pemain dengan indeks sebagai urutan, giliran bisa pake variabel counter dsb]
    - ii. Nama tiap pemain
    - iii. Simpan nama2 pemain dan urutan masuknya di variabel nama pemain
    - iv. Input nama file config, panggil hal2 inisialisasi file
  - d. Scanf EXIT
  - e. Validasi (notif ga sesuai, terus ulang)
- Terus load fungsi2nya
3. Game mulai (kalo new game): untuk
    - a. Baca file config, gunakan mesin kata. Terus misal butuh diubah jadi karakter buat MAP, bikin sistem lagi yang pake mesin kar
    - b. Bikin while loop
    - c. Buat list skill per pemain, buat array buff per pemain [BISA DIGANTI: masukkan aja ke ADT pemain, berarti bikin ADT nya. Bisa juga array diganti sama variabel boolean biasa]
    - d. MAP dieksekusi
    - e. Reset buff cermin, senter, senter
    - f. Ngaish skill. Kalau sudah penuh, display bahwa sudah penuh dan tidak ada skill yang didapat.. Kalau belum, kasih 1 skill, dengan  $\text{int } r = \text{rand}() \% 100$ . Angka2 taroh variabel:
      - i. Jika 0-13, insertlast pintu ga ke mana saja ke List
      - ii. Jika 14-19, insertlast cermin pengganda
      - iii. Jika 20-38, insertlast senter pembesar ke list
      - iv. Jika 39-57, insertlast senter pengecil ke list
      - v. Jika 58-66, insertlast mesin penukar posisi ke list
      - vi. Jika 66-99, display bahwa teknologi gagal
    - g. Aktivasi turn mulai dari pemain yang pertama diinput
    - h. Command2
    - i. (setelah roll) Baca array teleport
    - j. Jika ada yang nyampe ujung (ada boolean endgame),
      - i. End game, display bahwa permainan selesai
      - ii. Tampilkan siapa yang menang (eh tapi cek bagian roll)
      - iii. Tampilkan peringkat (cuma 2 orang sih)
  4. Sebuah turn
 

Bisa memasukkan command2

    - a. SKILL
      - i. Bikin list yang memuat daftar skill yang dimiliki (max 10). Untuk operasi pilih2 list bisa pake counter
      - ii. Bikin operasi “pakai” dan “buang”

- iii. (while) Baca isi list, Display command dan “Masukkan skill: “
- iv. Validasi input
- v. Skill2nya
  - 1. Pintu ga kemana saja
    - a. Buff imunitas teleport untuk pemain = true
    - b. (if) sudah punya imunitas, konfirmasi lagi apakah beneran mau dipake? Jika ya, display, gaada yang terjadi, jika tidak, display, + validasi jawaban (while)
    - c. Jika belum, display, terus set imunitas true
  - 2. Mesin waktu (kalo sempet)
    - a. Memundurkan pemain
  - 3. Baling baling jambu (kalo sempet)
    - a. Memajukan pemain
  - 4. Cermin pengganda
    - a. Jika buff Cermin Pengganda aktif atau jumlah udah 10, tidak dapat digunakan dan proses dicancel, balik ke loop skill
    - b. Jika tidak aktif, buang si skill dari List, ulang kasih 1 skill 2x, buff CerminPengganda = true, didisplay skill apa yang didapat
  - 5. Senter pembesar
    - a. Jika either buff senter pembesar or senter pengecil aktif, tidak dapat digunakan dan proses dicancel, balik ke loop skill
    - b. Jika tidak aktif, aktifkan
  - 6. Senter pengecil
    - a. Jika either buff senter pembesar or senter pengecil aktif, tidak dapat digunakan dan proses dicancel, balik ke loop skill
    - b. Jika tidak aktif, aktifkan
  - 7. Mesin penukar
    - a. Tukar posisi. (jika ternyata mau lebih dari 2 pemain, kasih pilihan pemain mana yag dipilih, jika nggak maka gausah), [GANTI] swap integer posisi pemain
  - 8. Teknologi gagal
    - a. Bukan skill
- b. MAP
  - i. Diinisialisasi dengan file config
  - ii. Bikin array sepanjang N
  - iii. Mesin kata membaca, dilanjut mesin kar membaca peta, mengisi array map dengan karakter2 yang dibaca

- iv. Untuk ngeprint map untuk tiap pemain, pakai for atau while loop dengan 2 kondisi print (jika indkes non injek, print isi peta, jika indeks injek print \*)
- c. BUFF
  - i. Buff disimpan di array boolean, tiap pemain punya 1 array [mungkin arraynya disatukan di ADT pemain], tiap buff punya indeks array sendiri
  - ii. Display isi array buff
  - iii. Buff2nya
    - 1. Imunitas teleport: jika true, mengaktifkan pilihan tidak kemana2 ketika roll terus land di teleport
    - 2. CerminPengganda: melarang penggunaan skill cermin pengganda jika true
    - 3. Senter pembesar: melarang penggunaan kedua senter jika true. Jika true, proses roll diubah minroll nya
    - 4. Senter pengecil: melarang penggunaan kedua senter jika true. Jika true, proses roll diubah maxrollnya.
- d. INSPECT: cek apakah ada teleporter atau # di petak ke-x
  - i. Get elmt indeks ke-x (?) di array Map, jika isinya “#” display bahwa dia petak terlarang
  - ii. Jika tidak, get elmt array teleport, cari apakah ada elemen petak asal yang = x, jika ada, display teleport ke mana
  - iii. Jika tidak, display kalo si petak x kosong
- e. ROLL
  - i. Memutar dadu
    - 1. Jika buff senter pembesar aktif,  $\text{num} = (\text{rand}() \% (\text{MaxRoll} - \text{floor}(\text{MaxRoll}/2) + 1)) + \text{floor}(\text{MaxRoll}/2)$
    - 2. Else if senter pengecil aktif,  $\text{num} = (\text{rand}() \% (\text{floor}(\text{MaxRoll}/2) - 1 + 1)) + 1$
    - 3. Else  $\text{num} = (\text{rand}() \% (\text{Maxroll} - 1 + 1)) + 1$
  - ii. Didapat num. display
  - iii. Cek posisi, cek posisi - num dan + num
  - iv. Cek apakah value2 itu melebihi indkes array, atau isinya #
  - v. Skenario dapat maju, dapat mundur, bisa maju mundur (pilih mau ke mana + validasi (berarti ada loop)), tidak bisa ke mana2. Variabel posisi pemain kemudian ubah
  - vi. Kalo maju, cek dah finish belum
- f. ENDTURN:
  - i. Pindah ke pemain berikutnya: search pemainnya lagi siapa, buat dia false, bikin pemain setelahnya true di array pemain
  - ii. Assign pemain sekarang siapa di variabel

- iii. Jika end round, push ADT pemain (intinya yang mengandung data round yang baru saja terjadi) dari kedua pemain. Kemudian print nomor ronde ke-berapa sekarang (berarti ada variabelnya)
- g. UNDO
  - i. Jika stacknya empty, display ga boleh
  - ii. Jika tidak, ubah array pemain lagi di siapa jadi sesuai urutan
  - iii. Pop stack dan ganti data2 pemain dengan yang baru aja di pop. Display, harus ada nomor ronde ke-berapa
  - iv. Jika stack nya masih ga empty, tawarkan mau undo lagi ga (+ validasi input), kalo iya maka ulang proses

Setelah itu, beban tugas kemudian didistribusikan menggunakan *wheel of names*:

Pengelompokkan bagi tugas:

1. Semua ADT , fungsi2, dan drivernya; Main menu: Vito [Nanti bagi2 biar rata]
2. Game mulai, MAP: Umar
3. SKILL, BUFF: Annel [SKILL bisa dipisah, skill dan buff sejenis jangan dipisah]
4. INSPECT, ROLL: zhillan
5. ENDTURN, UNDO: modan {Mungkin ADTstack di sini solanya cuma dipake di sini}

### Rapat 2021-11-07

Rapat ini menindaklanjuti hasil revisi-revisi Asistensi 1. ADT diperjelas, beberapa proses yang masih buram dijabarkan lagi dengan lebih detail. Didapat ADT sebagai berikut, yang kemudian harus ada fungsi-fungsi pengolah dan driver pengetesan untuk berbagai skenarionya:

ADT player: isinya nama, posisi, list Skill, arr Buff. Macro2

ADT list: buat skill. Isi skill = nomor skill

ADT array\_buff of boolean buat buff

ADT array\_map of char buat MAP

ADT array\_state of player buat daftar pemain

ADT arra\_teleport of integer buat teleport

ADT mesin\_kar (termasuk pembacaan file)

ADT mesin\_kata (di ignore, ada dua “blank” : space dan \n)

ADT stack of array\_state (representasi list)

Setelah itu, pembagian tugas diperbaiki:

Pengelompokan 2

1. Vito: Main menu, inisialisasi data2 player, ADT player, array\_state
2. Umar: Map, proses main: isi skill, panggil command (termasuk memanggil end turn), akhir main, ADT array\_map, mesin\_kata, mesin\_kar
3. Annel:SKILL pintu, cermin, senter, senter, mesin tukar, Buff semuanya, ADT array\_buff
4. Zhillan: SKILL maju, mundur, Inspect, Role, ADT list, bantu2in ADT player, array\_state
5. Modan: ENDTURN, UNDO, ADT Stack (bentuknya List) + Fungsi2nya (Push, Pop, CreateEmpty) + Driver



### 7.3 Log Activity Anggota Kelompok

Waktu	NIM	Keterangan
2021-11-06 15:00	18220008, 18220026, 18220039, 18220073, 18220087	Rapat perdana membahas rencana tubes mau melakukan apa dan bagi tugas
2021-11-06 19:00	18220073	Membuat sketsa awal untuk prosedur memulai game dan mengerjakan ADT mesin kata dan ADT mesin karakter
2021-11-07 20:00	18220008, 18220039, 18220073, 18220087	Asistensi 1
2021-11-07 21:00	18220008, 18220039, 18220073, 18220087	Rapat membahas hasil Asistensi 1, pembahasan ADT dan perbaikan struktur-struktur lain, revisi pembagian tugas
2021-11-07 22:30	18220008	Pengisian dan manajemen dokumen-dokumen relevan
2021-11-08 00:00	18220073	Pengerjaan pada file console dengan menambahkan fungsi mulai game, karakter <i>to</i> integer, dan <i>string to</i> integer.
2021-11-08 00:20	18220008, 18220073	Revisi 2 pembagian tugas
2021-11-08 19:00	18220073	Pengerjaan ADT array map, array teleportasi, serta melengkapi dan menyesuaikan pada file console
2021-11-08 23:00	18220073	Commit file konfigurasi, ADT array map, ADT array teleportasi, ADT mesin kata, ADT mesin karakter, ADT list (untuk sementara),

		boolean.h, dan console
2021-11-09 19.00	18220073	Pengerjaan kembali file console dan pembagian tugas untuk tiap anggota serta pembuatan prosedur untuk membaca file konfigurasi
2021-11-10 19.00	18220073	Pengerjaan kembali file console
2021-11-10 23.00	18220073	Commit revisi file console
2021-11-12 19.00	18220073	Pengerjaan driver untuk ADT array map dan ADT array teleportasi
2021-11-12 23.00	18220073	Commit driver untuk ADT array map dan ADT array teleportasi
2021-11-14 08:30	18220008	Pengerjaan ADT List
2021-11-14 13.30	18220073	Pengerjaan driver untuk ADT mesin kata dan ADT mesin karakter
2021-11-14 15.00	18220073	Commit driver untuk ADT mesin kata dan ADT mesin karakter
2021-11-14 21:30	18220008	Commit ADT list, pengerjaan Command Inspect
2021-11-14 23:45	18220008	Commit command Inspect, pengerjaan command Roll
2021-11-15 00:40	18220008	Commit command Roll, mengisi bagian dokumen, beres-beres README dan administrasi lain
2021-11-15 20:30	18220008	Dummy ADT player dan array_buff untuk fungsi2 di

		roll, fungsi2 tersebut. Penyelesaian command roll
2021-11-17 04:00	18220087	Commit ADT stack_state
2021-11-17 14.30	18220039	Commit main_menu
2021-11-17 19:30	18220008	Perbaikan roll
2021-11-17 20:00	18220026	Commit Array Buff, dan Skills.c
2021-11-17 23.30	18220073	Commit untuk update ADT array_map, array_tp, mesin_kata dan merapihkan console.
2021-11-20 23:25	18220087	Update ADT stack dan tambah ADT state
2021-11-21 02:30	18220087	Implementasi state dan stack ke console
2021-11-21 03:00	18220008	Pengerjaan dan commit kedua skill bonus, perbaikan roll
2021-11-22 00:30	18220087	Membetulkan inialisai stack dan state
2021-11-22 20:00	18220026	Commit Penambahan menuSkill, minorFix di skill dan penambahan ke consolenya
2021-11-24 01:14	18220087	Menambahkan driver ADT stack_state dan driver ADT state, serta memberi perubahan sedikit di console
2021-11-25 15:30	18220039	Menambahkan inialisasi player pada console.c dan command Help
2021-11-25 21:00	18220026	Commit pembuatan driver adt

		array_buff, MinorFix di bagian skill( scanf nya kebalik)
2021-11-25, 26, 27, 28	18220008, 18220087, 18220073, 18220039	Nguli hal2 banyak sampai meninggal
2021-11-28	18220039, 18220026	Nguli laporan

## 7.4 Form Asistensi

### Form Asistensi Tugas Besar IF2110/Algoritma dan Struktur Data

Sem. 1 2021/2022

No. Kelompok/Kelas : 12/K02






Nama Kelompok : SNEK AND MADDER

Anggota Kelompok (Nama/NIM) :

1. Zhillan Attarizal Rezyarifin/18220008
2. Annel Rashka Perdana/18220026
3. Alvito Rizqi Sobri/18220039
4. Umar Hakim/18220073
5. Mochammad Ramadhany/18220087






Asisten Pembimbing : Morgen Sudyanto/13518093

## Asistensi I

<b>Tanggal : 07 November 2021</b>	<b>Catatan Asistensi:</b>
<b>Tempat : Republik Indonesia</b>	
<b>Kehadiran Anggota Kelompok:</b>	
No NIM Tanda tangan  1 18220008   2 18220039   3 18220073   4 18220087 	
	<b>Tanda Tangan Asisten:</b>  

1. Q: Main file buat driver ADT terpisah dengan program? Berarti ada 2 dong?  
A: Dibuat driver untuk tiap ADT
2. Q: Makefile itu apa, gimana cara pakenya  
A: Buat mempersingkat command untuk compile
3. Q: Bagaimana tentang kompilasi di C kalo foldernya beda2  
A: Menyesuaikan dari directory filenya misal : gcc -o main src/main.c src/
4. Q: Masalah pada push di Github  
A: Masalah folder (mungkin karena namanya sama), hapus dulu yang di local, link, clone
5. Q: Tiap pemain perlu dalam map yang berbeda gak?  
A: Array map jadi integer aja jadi satu map cukup

## Asistensi II

Tanggal : 17 Oktober 2021	Catatan Asistensi:
Tempat : Republik Indonesia	
<p><b>Kehadiran Anggota Kelompok:</b></p> <p>No</p> <p>NIM</p> <p>Tanda tangan</p> <p>1</p> <p>18220008</p>  <p>2</p> <p>18220026</p>  <p>3</p> <p>18220039</p>  <p>4</p> <p>18220073</p>  <p>5</p> <p>18220087</p> 	
	Tanda Tangan Asisten:

1. Makefile. Gila kita. Kalo ga dilakukan, gila juga tapi.  
A: gaperlu makefile sebenarnya, asal bisa dicopas aja
2. ADT player, console, roll. Gimana cara deklarasi string tanpa batasan karakter?  
Soalnya kan kita harus ngevalidasi dengan kemungkinan inputan yang tak terhingga, kan.  
A: pake char\*

3. Requirement driver harus ada apa aja sih jadi? (Mungkin kita liatin driver2 yang udah jadi kek gimana, dan apakah udah memenuhi syarat)
4. Untuk array of player (state) mending declare di main atau bikin ADT baru?  
A : Mending bikin ADT baru yang isinya status dari game (ronde, array of player,dll)
5. ...  
A: Bikin parameter (?)
6. Command endturn otomatis kepanggil pas player udah selesai turn nya atau harus player yang manggil?  
A :command endturn dipanggil oleh masing2 player ketika udah selesai roll