

# **LAPORAN TUGAS BESAR**

## **IF2110/Algoritma dan Struktur Data**

### **Mobilita**

Dipersiapkan oleh:

Kelompok 10 Kelas 03

13520166 – Raden Rifqi Rahman

13520124 – Owen Christian Wijaya


13520121 – Nicholas Budiono

13520120 – Afrizal Sebastian

13520127 – Adzka Ahmadetya Zaidan

Sekolah Teknik Elektro dan Informatika - Institut Teknologi Bandung

Jl. Ganesha 10, Bandung 40132

	<b>Sekolah Teknik Elektro dan Informatika ITB</b>	<b>Nomor Dokumen</b>		<b>Halaman</b>
		<i>IF2110-TB-10-03</i>		22
		<i>Revisi</i>	<i>01</i>	<i>17 – 11 - 2021</i>

# Daftar Isi

1 Ringkasan.....	4
2 Penjelasan Tambahan Spesifikasi Tugas .....	4
2.1 Spesifikasi VIP Item.....	4
2.2 Spesifikasi Fungsi RETURN / Return to Sender .....	4
2.3 Spesifikasi Gadget Senter Pengecil.....	4
2.4 Spesifikasi Fungsi Save File .....	4
3 Struktur Data (ADT).....	5
3.1 ADT Mesin Karakter (Character Machine) .....	5
3.2 ADT Mesin Kata (Word Machine) .....	5
3.3 ADT List Statis.....	5
3.4 ADT Matriks .....	5
3.5 ADT Queue .....	6
3.6 ADT Stack.....	6
3.7 ADT Point .....	6
3.8 ADT Location .....	6
3.9 ADT Gadget .....	7
3.10 ADT Item.....	7
3.11 ADT Linked List (ItemList) .....	7
3.12 ADT List Dinamis (LocationList) .....	7
3.13 ADT Peta (GameMap).....	7
3.14 ADT State.....	7
4 Program Utama .....	8
5 Algoritma-Algoritma Menarik.....	8
5.1 Algoritma Konversi Word ke String .....	8
5.2 Algoritma Komparasi Dua String .....	9
5.3 Algoritma Pick Up .....	9
5.4 Algoritma Drop Off.....	9
5.5 Algoritma To Do List dan In Progress .....	10
5.6 Algoritma Buy dan Inventory .....	10
5.7 Algoritma Move .....	10
5.8 Algoritma Pembaruan State .....	10
6 Data Test .....	11
6.1 Pengujian Loading File Konfigurasi / File Save .....	11
6.2 Pengujian Pergerakan (MOVE) dan Peta (MAP) .....	11
6.3 Pengujian Pengambilan Barang (PICK_UP).....	12
6.4 Pengujian Pengantaran Barang (DROP_OFF) .....	12
6.5 To-Do dan In-Progress List.....	13
6.6 Pengujian Sistem Pembelian dan Penggunaan Gadget .....	13
6.7 Pengujian Efek dari Drop-Off Barang .....	14
6.8 Pengujian Gadget .....	16
7 Test Script .....	18
8 Pembagian Kerja dalam Kelompok .....	20
9 Lampiran .....	21
9.1 Deskripsi Tugas Besar.....	21

9.2	Notulen Rapat.....	21
9.3	Log Activity Anggota Kelompok.....	21

# 1 Ringkasan

“Mobilita” adalah aplikasi permainan berbasis CLI (command-line interface) tentang simulasi pengantaran barang. Isi laporan ini mencakup spesifikasi fitur-fitur tugas yang ada, tipe-tipe struktur data abstrak (ADT) dan modifikasi lebih lanjut ADT, algoritma-algoritma menarik yang digunakan, dan dokumentasi percobaan penggunaan aplikasi. Kesimpulan dari tugas ini adalah Bahasa C adalah Bahasa yang versatile dan baik untuk digunakan dalam penerapan algoritma, pembuatan struktur data abstrak, dan pengaplikasian bersifat CLI.

## 2 Penjelasan Tambahan Spesifikasi Tugas

### 2.1 Spesifikasi VIP Item

VIP Item adalah jenis *item* tambahan selain *heavy* dan *perishable item*. VIP Item harus di-pickup terlebih dahulu apabila ada di *to-do list*. Pemain masih dapat melakukan drop-off apabila ada VIP item di *to-do list*, namun pemain tidak dapat melakukan pick-up apabila ada VIP item di *in-progress list*. Apabila semua *VIP item* di *to-do list* sudah di-pickup, item-item lainnya dapat di-pickup. Saat di-dropoff, *VIP Item* akan memberikan pemain akses ke fungsi RETURN. Fungsi RETURN dapat dilakukan sebanyak jumlah VIP Item yang telah di-dropoff (misalnya dropoff VIP Item 2 kali, maka fungsi RETURN dapat dilakukan 2 kali).

### 2.2 Spesifikasi Fungsi RETURN / Return to Sender

Fungsi RETURN dapat diakses pemain apabila pemain berhasil melakukan drop-off VIP Item. Fungsi RETURN memberikan hak kepada pemain untuk mengembalikan item teratas pada bag ke to-do list, kecuali untuk tipe item VIP yang tidak bisa dikembalikan. Fungsi mengembalikan item ke posisi terakhir to-do list. Apabila item bertipe PERISHABLE, maka waktu hangus item akan diulang ke waktu awalnya. Fungsi ini dapat dieksekusi dari posisi manapun di peta.

### 2.3 Spesifikasi Gadget Senter Pengecil

Gadget 'Senter Pengecil' adalah tambahan *gadget* untuk daftar *gadget*. Apabila *item* teratas di tas bertipe *heavy*, maka efek dari *heavy item* tersebut hilang. Waktu perpindahan pemain berkurang 1 (apabila ada *heavy item*). Saat melakukan drop-off/return, efek tersebut akan hilang.

### 2.4 Spesifikasi Fungsi Save File

Fungsi SAVE digunakan untuk menyimpan data permainan ke suatu file .txt menggunakan modul stdio.h. Fungsi akan menuliskan isi konfigurasi permainan dengan melihat *global state* yang ada, kemudian menuliskan isinya dengan format tertentu. Fungsi lainnya adalah *loadfile parser* yang membaca isi dari *save file* sebelumnya dan menyocokkan *global state* ke isi dari *save file*. Fungsi *parser* dapat diakses dari menu awal dengan perintah 'LOAD GAME'. Pemain kemudian memasukkan nama file penyimpanan dan program akan membaca isinya dan menyocokkan dengan *global state* menggunakan mesin kata.

## 3 Struktur Data (ADT)

### 3.1 ADT Mesin Karakter (Character Machine)

ADT Mesin Karakter adalah ADT yang menangani berbagai jenis masukan dari pengguna. ADT ini bekerja dengan cara membaca masukan dari pengguna baik melalui *console* ataupun file eksternal yang berupa sebuah pita karakter '*character tape*'. ADT ini tersusun atas *state* yang berupa karakter saat ini '*current character*' dan akhir dari pita '*end of tape*'. Untuk setiap pita karakter, mesin karakter akan membaca satu demi satu karakter dalam pita hingga dicapai akhir pita. Ketika akhir dari pita dicapai, maka *state end of tape* pada mesin karakter akan menyala sebagai tanda bahwa pembacaan telah selesai. ADT mesin karakter digunakan karena merupakan ADT yang fundamental/utama dalam penanganan berbagai masukan dari pengguna dalam program yang dibuat.

### 3.2 ADT Mesin Kata (Word Machine)

ADT Mesin Kata adalah ADT yang merupakan hasil implementasi ADT Mesin Karakter. ADT ini mensimulasikan sebuah mesin kata yang membaca sebuah pita berisi karakter dan kemudian mengakuisisi kata-kata yang ada dalam pita tersebut. ADT Mesin Kata digunakan untuk mengakuisisi *command* dan *input* dalam aplikasi. Selain itu, ADT ini juga digunakan untuk mengakuisisi *state game* dari file eksternal. Alasan pemilihan ADT Mesin Kata adalah penggunaan yang lebih efisien dalam mengakuisisi satu kata dibandingkan *library stdio.h*, mempertimbangkan juga bentuk *input* dengan panjang satu kata atau satu karakter. Selain itu, ADT Mesin Kata juga bisa membaca input dari file eksternal, sehingga dapat digunakan untuk mengakuisisi *input* dari file konfigurasi. File terletak di folder `modules/io/machines/wordmachine.c`.

### 3.3 ADT List Statis

ADT List Statis adalah ADT yang mereplikasikan struktur data *list*, dimana setiap data di dalam *list* tersusun secara kontigu rapat kiri. ADT List Statis dimodifikasi dalam bentuk *gadget list* digunakan untuk menyimpan *gadget* yang dipilih user. ADT List ini juga digunakan dalam menu *inventory* untuk menampilkan gadget yang ada. ADT List digunakan karena dapat diakses dari indeks manapun (tidak terbatas di ujung-ujung struktur data) dan konstruktor dan selektor yang mudah diakses untuk sistem inventarisasi sederhana. Penerapan ADT list dapat dilihat di `models/gadget_list.c`.

### 3.4 ADT Matriks

ADT Matriks adalah ADT yang merepresentasikan struktur data matriks/array dua dimensi. ADT Matriks digunakan untuk menyimpan matriks *adjacency* yang menunjukkan hubungan rute antar lokasi. Selain itu, matriks juga digunakan untuk membuat peta map berdasarkan daftar dari lokasi yang dibaca di file dokumentasi eksternal. ADT Matriks juga digunakan untuk menyimpan daftar lokasi dalam bentuk matriks. ADT Matriks digunakan karena cocok untuk merepresentasikan sebuah "peta" secara dua-dimensi dan menyimpan data secara lokasi dalam skala yang lebih baik dibandingkan ADT lainnya. Penerapan ADT matriks dapat dilihat di file `models/boolean_matrix.c`, `game_map.c`, atau `location_matrix.c`.

### 3.5 ADT Queue

ADT Queue adalah ADT yang digunakan untuk menyimpan data yang diatur secara antrian. ADT Queue mensimulasikan sebuah "antrian", di mana data yang diakses adalah data di urutan paling depan (indeks 0/head) dan data yang dimasukkan selalu dari belakang (tail). Elemen ditangani secara FIFO (*First In First Out*), di mana penyisipan elemen selalu dilakukan setelah elemen terakhir, dan penghapusan elemen dilakukan pada elemen pertama. Dalam tugas ini, implementasi queue dilakukan dalam bentuk array, dengan elemen yang bisa diakses adalah elemen terdepan dan elemen belakang (head/tail).

Dalam tugas ini, implementasi ADT Queue dapat dilihat di berkas `models/item_queue.c`. ADT Queue digunakan untuk mensimulasikan antrian *item* sesuai urutan masuknya. ADT Queue dipilih karena penanganan elemen secara FIFO cocok untuk digunakan untuk mensimulasikan antrian barang yang masuk.

### 3.6 ADT Stack

ADT Stack adalah ADT yang digunakan untuk menyimpan dan mengolah data yang berbentuk seperti "tumpukan". Mempunyai kesamaan dengan ADT Queue, ADT Stack mensimulasikan sebuah "tumpukan", di mana elemen yang diolah adalah elemen yang berada di bagian teratas. ADT Stack mengolah data secara LIFO (*Last In First Out*), di mana penyisipan dan penghapusan dilakukan terhadap elemen teratas.

Dalam tugas ini, ADT Stack digunakan untuk menyimpan daftar *item* tidak secara antrian, namun digunakan secara spesifik pada bagian *inventory*. Hal ini disebabkan efek-efek dari *item gadget* yang mengolah item-item pada posisi atas, sehingga penggunaan ADT Stack lebih efektif dibandingkan ADT Queue. Implementasi ADT Stack dapat dilihat di berkas `models/item_stack.c`.

### 3.7 ADT Point

ADT Point digunakan untuk merepresentasikan koordinat pemain dan bangunan di permainan. ADT ini digunakan bersama-sama dengan ADT Matriks karena ADT Matriks yang merepresentasikan lokasi secara koordinat, sehingga ADT ini efektif untuk digunakan karena sama-sama merepresentasikan lokasi secara koordinat Cartesian.

### 3.8 ADT Location

ADT Location adalah ADT yang digunakan untuk merepresentasikan lokasi-lokasi yang ada di dalam file konfigurasi. Sebuah *Location* berisi data-data id lokasi, simbol lokasi, koordinat lokasi dalam representasi ADT Point, dan empat tanda '*flag*' yang digunakan untuk menandakan bahwa sebuah lokasi adalah tempat pemain berada, tempat mengambil *pick up item*, tempat *drop off item*, atau tempat yang dapat dicapai '*reachable*' relatif terhadap tempat pemain berada. ADT Location digunakan untuk membungkus '*wrap*' semua informasi/data yang direpresentasikan oleh sebuah lokasi pada peta permainan sehingga data-data tersebut tidak diproses secara terpisah atau tersebar '*scattered*'.

### 3.9 ADT Gadget

ADT Gadget adalah ADT yang merepresentasikan gadget. ADT Gadget terdiri atas ID gadget, nama gadget, dan harga gadget. ADT Gadget nantinya akan ditambahkan dan dihapus dari gadget list setiap ada pembelian atau pemakaian gadget. Karena gadget dapat dipilih dari urutan 1-5, maka ADT List digunakan untuk menampung gadget.

### 3.10 ADT Item

ADT Item digunakan untuk merepresentasikan item-item yang ada. Terdapat informasi tentang waktu antrian item, tipe item, lokasi pick-up dan drop-off, serta waktu hangus (apabila tipe item *perishable*). Sama halnya dengan ADT Location, ADT Item digunakan untuk membungkus 'wrap' semua informasi/data yang direpresentasikan oleh sebuah item/pesanan dalam permainan sehingga data-data tersebut tidak diproses secara terpisah atau tersebar 'scattered'.

### 3.11 ADT Linked List (ItemList)

ADT Linked List adalah tipe data list yang berbasis *node*. Dalam program yang dibuat, ADT linked list digunakan sebagai list item yang disebut dengan ItemList untuk menyimpan data *todo list* dan *in progress list* dalam satu permainan. Dalam ItemList, setiap node berisi satu data bertipe Item dan referensi (*pointer*) ke *node* berikutnya. ItemList digunakan untuk menyimpan data *todo list* dan *in progress list* karena *linked list* tidak memiliki kapasitas dan operasi *insert* dan *delete* sering dilakukan pada *todo list* dan *in progress list*. Dengan demikian, operasi pada kedua list tersebut tidak perlu memperhitungkan kapasitas list sehingga operasi-operasi yang dilakukan lebih efisien.

### 3.12 ADT List Dinamis (LocationList)

ADT List Dinamis adalah list dengan kapasitas yang dinamis. Dalam program yang dibuat, ADT list digunakan untuk list lokasi yang ada, yang disebut dengan LocationList. LocationList menggunakan list dinamis karena banyaknya lokasi yang ada dalam satu permainan ditentukan oleh file konfigurasi yang dimasukkan. Dengan demikian, memori dapat lebih dihemat dengan menyesuaikan kapasitas LocationList sesuai dengan banyaknya list yang ada dalam file konfigurasi.

### 3.13 ADT Peta (GameMap)

ADT GameMap adalah ADT yang digunakan sebagai representasi peta 'map' dari permainan yang berlangsung. ADT ini berisi data ukuran peta dan data-data yang berkaitan dengan lokasi dalam permainan. GameMap adalah pembungkus 'wrapper' dari data-data lokasi yang didefinisikan dalam permainan, matriks ketetanggaan '*adjacency matrix*' dari lokasi-lokasi tersebut, dan matriks lokasi. ADT ini digunakan agar operasi atau proses-proses yang berkaitan dengan mekanisme lokasi dan pergerakan pemain dilakukan dalam satu buah variabel yang terpusat.

### 3.14 ADT State

ADT State adalah ADT yang digunakan sebagai variabel global untuk permainan. ADT State mengandung peta dari permainan, matriks *adjacency*, daftar item, daftar gadget, dan properti-

properti lainnya. ADT State dibuat agar pengolahan variabel secara global dapat dilakukan sehingga modularitas program terjaga dengan baik.

## 4 Program Utama

Program utama terletak di file `main.c`. Saat dijalankan, akan ada 3 opsi, yaitu opsi New Game, Load Game dan Exit. Saat memilih opsi New Game/Load Game, pemain akan memasukkan direktori dari file konfigurasi. Apabila file konfigurasi sesuai dengan persyaratan, maka program akan membaca isi file konfigurasi dan membuat ADT State berdasarkan informasi yang ada. Setelah itu, program akan menampilkan menu utama dari permainan. Pemain dapat mengetikkan "HELP" untuk menampilkan menu permainan. Ada beberapa *command* dalam permainan:

1. MOVE → Untuk berpindah ke lokasi yang diinginkan
2. PICK\_UP → Untuk mengambil item di lokasi sekarang
3. DROP\_OFF → Mengantarkan item ke lokasi sesuai tumpukan pesanan
4. MAP → Untuk melihat peta
5. TO\_DO → Untuk melihat pesanan yang masuk ke to-do list
6. IN\_PROGRESS → Untuk melihat pesanan yang sedang dikerjakan
7. BUY → Untuk membeli item (hanya bisa dipanggil di HQ)
8. INVENTORY → Untuk melihat isi inventory
9. RETURN → Mengembalikan item teratas di tas
10. SAVE → Menyimpan konfigurasi permainan
11. HELP → Untuk menampilkan bantuan
12. STATS → Untuk menampilkan *ability* yang sedang aktif dan efek *item*
12. MENU → Untuk kembali ke menu utama
12. EXIT → Untuk keluar dari game

Implementasi dari fungsi-fungsi dalam game dapat dilihat lebih lanjut di Bab 5 dan Bab 6.

Pemain bertugas untuk mengantarkan semua *item* yang ada di *to-do list*. Pemain dapat membeli *gadget* yang mampu membantu mereka bermain di *headquarters* setelah mengumpulkan uang. Apabila pemain berhasil mengantarkan semua *item* (tidak ada pesanan lagi di *to-do list* atau pesanan yang harus diantarkan di *in-progress list*), maka pemain akan kembali ke *headquarters* untuk menyelesaikan permainan.

## 5 Algoritma-Algoritma Menarik

### 5.1 Algoritma Konversi Word ke String

*Word* adalah tipe data hasil akuisisi ADT mesin kata yang berisi panjang kata dan *array* karakter. Akan tetapi, program yang dibuat tidak memproses *Word* secara langsung, melainkan mengkonversinya terlebih dahulu menjadi sebuah *string* atau *array* karakter. Algoritma konversi *Word* ke *string* diimplementasi pada fungsi `stringify()` pada file utilitas *Word* (`word_utils.c`) dengan parameter sebuah *Word*. Algoritma ini menarik karena dengan memanfaatkan perilaku '*behavior*' bahasa C bahwa setiap *string* selalu berakhiran karakter *null* '\0', maka dengan menambah karakter *null* ke dalam *array* karakter yang sudah ada pada sebuah *Word*, data

STEI- ITB	IF2110-TB-10-03	Halaman 8 dari 22 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.		



panjang kata dalam sebuah Word dapat dihapus sehingga Word dapat dikonversi menjadi sebuah *string* (array karakter). Selanjutnya, *string* hasil konversi inilah yang diproses dalam program yang dibuat.

## 5.2 Algoritma Komparasi Dua String

Algoritma komparasi dua *string* juga diimplementasi dalam file *word\_utils.c*, yaitu dalam fungsi *isStringEquals()*. Algoritma ini digunakan untuk membandingkan masukan '*input*' dari pengguna dengan perintah '*command*' yang ada. Algoritma ini menarik karena dengan memanfaatkan sifat *string* yang selalu berakhiran karakter *null*, maka komparasi dua buah *string* dapat mudah dilakukan dengan cara mengiterasi *string* sebagai *array* karakter. Komparasi dilakukan dengan cara membandingkan setiap karakter dalam dua buah *string* pada indeks yang sama secara berurutan hingga ditemukan karakter *null* tersebut. Apabila pada saat perbandingan terdapat karakter yang berbeda di antara kedua *string* pada indeks yang sama atau dicapai karakter *null* hanya pada salah satu *string*, maka jelas bahwa kedua *string* tidaklah sama.

## 5.3 Algoritma Pick Up

Algoritma *command* PICK UP dipanggil di *main.c* sebagai prosedur *pick\_up()*. Saat *command* dipanggil maka prosedur *pick\_up()* akan terpanggil dan melakukan pengecekan pada To Do List untuk mencari ada atau tidak item berupa VIP item, pengecekan dilakukan dengan fungsi *isVIPintodo(ItemList todo)*. Jika VIP item ada pada To Do List maka prosedur *pick\_up* akan mendahulukan VIP item. Ketika prosedur *pick\_up()* dijalankan pada suatu Location dan ada VIP item di To Do List, maka akan dilakukan mencari index VIP item di To Do List. Jika index sama dengan -1 maka VIP item tidak berada di Location tersebut dan akan ditampilkan sebuah pesan. Jika index tidak -1, maka pada Location tersebut ada VIP item.

Saat melakukan Pick Up VIP item dan sebuah Flag yang menandakan ada sebuah VIP item pada In Progress menjadi True. Jika tidak ada VIP item pada To Do List, saat dijalankan prosedur *pick\_up()* pada suatu Location maka akan dilakukan pengecekan apakah ada item yang dapat diambil. Jika tidak ada maka akan ditampilkan sebuah pesan, jika tidak maka akan dilakukan pengecekan jenis item yang diambil dan jika yang diambil adalah Heavy item, maka Flag yang menandakan ada sebuah Heavy item pada tas True. Item akan ditambahkan ke In Progress dan di Push ke Bag dan dilakukan set Drop Off Location untuk melakukan pewarnaan pada map, serta dilakukan pengecekan apakah ada item lain yang berada di To Do List yang memiliki Pick Up Location yang sama dengan Item yang baru saja di Pick Up agar Location tersebut dapat tetap menjadi Pick Up Location atau tidak.

## 5.4 Algoritma Drop Off

Algoritma *command* DROP OFF, ketika dipanggil di *main.c* sebagai prosedur *drop\_off()*, pertama-tama melakukan pengecekan apakah isi tas kosong atau lokasi karakter saat ini tidak cocok dengan lokasi tujuan dari *top item* pada *bag*. Jika terpenuhi, maka akan ditampilkan pesan pada layar "Tidak dapat pesanan yang dapat diantarkan!". Sedangkan jika kedua kondisi tersebut tidak terpenuhi, berarti karakter berada pada lokasi yang tepat dan akan berlanjut. Setelah itu, prosedur akan menghapus isi *list* dari *inProgressList* pada urutan akhir dan menghapus *item* pada

*stack* teratas *bag*. Setelah penghapusan, prosedur akan mencocokkan jenis *item* yang dihapus dan akan memberikan *reward* dan/atau efek tergantung dengan jenis *item* tersebut.

## 5.5 Algoritma To Do List dan In Progress

Command TO DO dan IN PROGRESS dipanggil di main.c sebagai prosedur *to\_do()* dan *in\_progress()*, masing-masing menampilkan isi *list* *todoList* dan isi *list* *inProgressList*. Kedua prosedur tersebut akan melakukan *print* isi dari *list* tersebut dan akan diulang secara berurutan hingga isi dari *list* habis. Jika pada kondisi awal adalah *list* kosong, maka prosedur akan menampilkan pesan bahwa isi dari *list* tersebut kosong.

## 5.6 Algoritma Buy dan Inventory

Pada saat pemain menjalankan perintah *buy*, maka program akan menampilkan suatu menu berisi gadget-gadget yang akan dibeli. Pemain dapat membeli gadget yang diinginkan apabila pemain mempunyai uang yang cukup untuk membeli gadget. Gadget-gadget yang dibeli dapat diakses di menu *inventory* dan digunakan. Gadget yang digunakan mempunyai prekondisi yang apabila tidak terpenuhi, maka akan terpakai sia-sia. Efek dari gadget yang digunakan juga akan langsung diimplementasikan ke permainan.

## 5.7 Algoritma Move

Pada saat pemain menjalankan perintah *MOVE*, program akan menampilkan lokasi-lokasi apa saja yang dapat diakses (*adjacent*) dari lokasi pemain. Setelah itu, program akan meminta input dari pengguna. Apabila input pengguna valid, maka program akan memindahkan pemain ke tempat yang diinginkan, dan meng-unset lokasi *adjacent* sebelumnya agar tidak konflik dengan lokasi-lokasi *adjacent* dari lokasi baru. Apabila input tidak valid, program akan meminta input ulang hingga input benar.

## 5.8 Algoritma Pembaruan State

*State* di permainan diperbarui terus-menerus setiap kali ada perubahan waktu dalam permainan. Setiap kali waktu berubah, *queue* pesanan akan dicek untuk melihat apakah ada pesanan yang bisa dimasukkan ke *to-do list*. *Tas* akan dicek untuk melihat apakah ada *item perishable* yang sudah mencapai waktu hangus. Dengan demikian, *state game* dapat berubah otomatis sesuai dengan perubahan waktu.

## 6 Data Test

### 6.1 Pengujian Loading File Konfigurasi / File Save

Fitur: saat permainan dimulai, pemain memasukkan nama file (apabila file diletakkan di folder yang sama dengan terminal) atau beserta *path* dari file (apabila disimpan di folder lain). Program akan memvalidasi file konfigurasi dan memulai permainan menggunakan data yang ada di file konfigurasi. Apabila fungsi LOAD digunakan, maka fungsi akan mengambil data yang disimpan di *savefile* dan melanjutkan permainan berdasarkan spesifikasi yang ada.

```
=====
[~:|/ 0 \0 }||| | |C 3/0\
 |N/| \ /0 }||| | |C 3/0\
=====
Selamat datang di MOBILITA!

Ketikkan NEW GAME untuk memulai permainan baru!
Ketikkan LOAD GAME untuk membuka file penyimpanan!
Ketikkan EXIT untuk keluar dari game!
=====
ENTER COMMAND: NEW GAME
Masukkan file konfigurasi
>>> config.txt

Lokasi Mobita: 8 (1, 1)
Waktu: 0
Uang: 0
Anda sedang berada di Headquarters.
```

```
=====
[~:|/ 0 \0 }||| | |C 3/0\
 |N/| \ /0 }||| | |C 3/0\
=====
Selamat datang di MOBILITA!

Ketikkan NEW GAME untuk memulai permainan baru!
Ketikkan LOAD GAME untuk membuka file penyimpanan!
Ketikkan EXIT untuk keluar dari game!
=====
ENTER COMMAND: LOAD GAME
Masukkan save file
>>> save.txt
File save berhasil di-load!

Lokasi Mobita: 6 (3, 8)
Waktu: 6
Uang: 0
```

### 6.2 Pengujian Pergerakan (MOVE) dan Peta (MAP)

Fitur: program menunjukkan lokasi-lokasi yang *adjacent* terhadap lokasi pemain, pemain memasukkan angka yang mewakili lokasi yang ingin dituju. Apabila ada *item* yang dapat di-*pick up* atau di-*drop off*, warna lokasi akan berubah menjadi merah (*pick up*) atau biru (*drop off*)

Hasil: program memindahkan lokasi pemain di *state* menjadi lokasi yang dituju

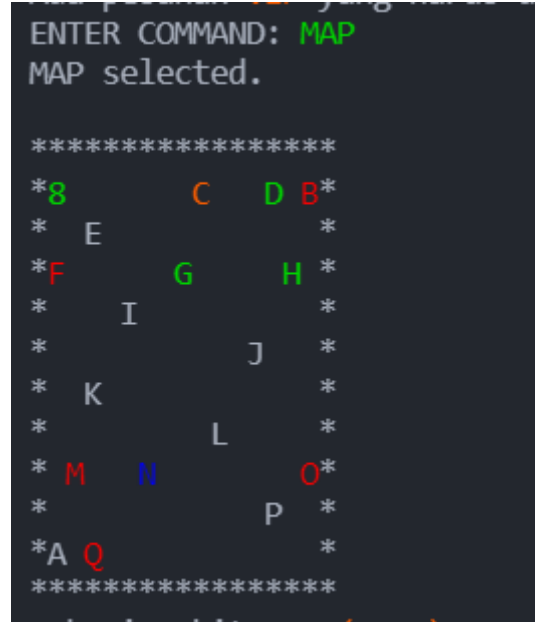
```
ENTER COMMAND: MOVE
MOVE selected.

Lokasi-lokasi yang dapat diakses:
1. 8 (1, 1)
2. D (1, 13)
3. G (3, 8)
4. H (3, 14)
Posisi yang dipilih? (ketik 0 jika ingin kembali)
ENTER COMMAND: 3

Mobita sekarang berada di titik G(3, 8)!
Waktu : 2 (+1)
```

Pemain juga dapat melihat peta yang akan menampilkan lokasi-lokasi di peta. Lokasi pemain diwakili oleh warna jingga, lokasi yang dapat diakses diwakili oleh warna hijau, lokasi-lokasi di

mana ada *item* yang bisa di-*pick up* diwakili oleh warna merah, dan lokasi *drop off* dari item teratas di tas diwakili warna biru.



### 6.3 Pengujian Pengambilan Barang (PICK\_UP)

Fitur: program akan menuliskan pesan “Ada barang yang bisa di-pick up!” apabila ada barang yang bisa diambil di lokasi pemain

Hasil: *item* dimasukkan ke tas dan dapat dilihat di *in progress list* apabila tas pemain tidak penuh, namun jika tas pemain penuh, *item* tidak dapat diambil. Lokasi *drop off* dari *item* akan diaktifkan sehingga barang bisa diantar.

Lokasi Mobita: G (3, 8) Waktu: 2 Uang: 0 Ada pesanan yang dapat di-pickup di sini!  ENTER COMMAND: PICK_UP PICK_UP selected.  Pesanan berupa Normal Item berhasil diambil! Tujuan Pesanan : N	Lokasi Mobita: G (3, 8) Waktu: 12 Uang: 200 Ada pesanan yang dapat di-pickup di sini!  ENTER COMMAND: PICK_UP PICK_UP selected.  Tas telah penuh
--	--

### 6.4 Pengujian Pengantaran Barang (DROP\_OFF)

Fitur: program akan menampilkan pesan “Ada barang yang bisa di-dropoff!” apabila lokasi pemain mewakili lokasi *drop off* dari *item* teratas di bag.

Hasil: *item* akan diantarkan, efek dari *item* akan segera terimplementasi. Lokasi *drop off* yang aktif akan diubah menjadi lokasi *drop off* item teratas.

STEI- ITB	IF2110-TB-10-03	Halaman 12 dari 22 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.		

```
Lokasi Mobita: D (1, 13)
Waktu: 8
Uang: 0
Ada pesanan yang dapat di-dropoff di sini!

ENTER COMMAND: DROP_OFF
DROP_OFF selected.

Pesanan Normal Item berhasil diantarkan!
Uang yang didapatkan: 200 Yen
```

```
Lokasi Mobita: N (8, 6)
Waktu: 22
Uang: 200
Ada pesanan yang dapat di-dropoff di sini!

ENTER COMMAND: DROP_OFF
DROP_OFF selected.

Pesanan Heavy Item berhasil diantarkan!
Uang yang didapatkan: 400 Yen
Speed Boost activated!
```

```
Lokasi Mobita: N (8, 6)
Waktu: 12
Uang: 0
Ada pesanan yang dapat di-dropoff di sini!

ENTER COMMAND: DROP_OFF
DROP_OFF selected.

Pesanan Perishable Item berhasil diantarkan!
Uang yang didapatkan: 400 Yen
Kapasitas tas bertambah 1!
```

```
Lokasi Mobita: P (9, 13)
Waktu: 25
Uang: 800
Ada pesanan yang dapat di-pickup di sini!
Ada pesanan yang dapat di-dropoff di sini!

ENTER COMMAND: DROP_OFF
DROP_OFF selected.

Pesanan VIP Item berhasil diantarkan!
Uang yang didapatkan: 600 Yen
Efek 'Return to Sender' bertambah menjadi 1.
```

Pengujian drop-off di *item* yang bertipe beda (*normal*, *heavy*, *perishable*, *VIP*)

## 6.5 To-Do dan In-Progress List

Fitur: pemain dapat melihat *item-item* yang dapat diambil beserta tipe dan lokasi *item* menggunakan perintah TO\_DO. Pemain dapat melihat daftar *item* yang telah diambil menggunakan perintah IN\_PROGRESS.

```
Lokasi Mobita: C (1, 9)
Waktu: 6
Uang: 0

ENTER COMMAND: TO_DO
TO_DO selected.

Ada 4 pesanan pada To Do List:
1. M -> B (Heavy Item)
2. B -> M (Normal Item)
3. F -> E (Normal Item)
4. G -> N (Perishable Item, sisa waktu 10)
```

```
Lokasi Mobita: C (1, 9)
Waktu: 6
Uang: 0

ENTER COMMAND: IN_PROGRESS
IN_PROGRESS selected.

Ada 3 pesanan yang sedang diantarkan :
1. Normal Item (Tujuan : N)
2. Heavy Item (Tujuan : N)
3. Normal Item (Tujuan : D)
```

## 6.6 Pengujian Sistem Pembelian dan Penggunaan Gadget

Fitur: saat pemain berada di *headquarters*, pemain dapat menggunakan perintah BUY untuk membeli *gadget* yang diinginkan apabila uang yang telah dikumpulkan pemain mencukupi. *Gadget* yang dibeli dapat digunakan menggunakan perintah INVENTORY. Apabila

STEI- ITB	IF2110-TB-10-03	Halaman 13 dari 22 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.		

*prerequisites* keadaan tidak terpenuhi untuk penggunaan *gadget* tertentu (mis. barang teratas harus bertipe *heavy/perishable*), maka *gadget* akan terpakai sia-sia.

<pre> Lokasi Mobita: 8 (1, 1) Waktu: 25 Uang: 2800 Anda sedang berada di Headquarters.  ENTER COMMAND: BUY BUY selected.  Uang anda sekarang: 2800 Yen Gadget yang tersedia: 1. Kain Pembungkus Waktu (800 Yen) 2. Senter Pembesar (1200 Yen) 3. Pintu Kemana Saja (1500 Yen) 4. Mesin Waktu (3000 Yen) 5. Senter Pengecil (800 Yen) Gadget mana yang ingin kau beli? (ketik 0 jika ingin kembali)  ENTER OPTION: 2 Gadget 'Senter Pembesar' berhasil dibeli! </pre>	<pre> Lokasi Mobita: 8 (1, 1) Waktu: 25 Uang: 100 Anda sedang berada di Headquarters.  ENTER COMMAND: INVENTORY INVENTORY selected.  1. Senter Pembesar 2. Pintu Kemana Saja 3. - 4. - 5. - Gadget mana yang ingin digunakan? (ketik 0 jika ingin kembali)  ENTER OPTION: 2 Gadget 'Pintu Kemana Saja' berhasil digunakan! Gunakan perintah 'MOVE' untuk berpindah tanpa menambah waktu! </pre>
--	---

## 6.7 Pengujian Efek dari Drop-Off Barang

Fitur: saat pemain melakukan *drop off*, maka efek samping dari item tersebut akan langsung diimplementasikan ke permainan.

Efek *Speed Boost* akan aktif apabila pemain berhasil melakukan *drop off item* bertipe *heavy*. Efek ini memungkinkan pemain untuk bergerak dua kali dengan menambah waktu sekali.

<pre> ENTER COMMAND: MOVE MOVE selected.  Lokasi-lokasi yang dapat diakses: 1. K (6, 3) 2. L (7, 10) 3. M (8, 2) 4. Q (10, 3) Posisi yang dipilih? (ketik 0 jika ingin kembali) Speedboost aktif! Berlaku untuk 5 kali pemakaian.  ENTER COMMAND: 4  Mobita sekarang berada di titik Q(10, 3)! Waktu : 12 (+0) </pre>	<pre> ENTER COMMAND: MOVE MOVE selected.  Lokasi-lokasi yang dapat diakses: 1. A (10, 1) 2. M (8, 2) 3. N (8, 6) Posisi yang dipilih? (ketik 0 jika ingin kembali) Speedboost aktif! Berlaku untuk 5 kali pemakaian.  ENTER COMMAND: 2  Mobita sekarang berada di titik M(8, 2)! Waktu : 13 (+1) </pre>
---	---

Pemain dapat menggunakan perintah RETURN apabila berhasil mengantarkan *item* bertipe *VIP*. Perintah RETURN memungkinkan pemain untuk mengembalikan *item* teratas pada bag ke urutan terakhir di *to-do list*. Perintah ini tidak dependen terhadap lokasi pemain, berarti pengembalian *item* dapat dilakukan dari mana saja. Perintah ini dapat dilakukan sebanyak jumlah pengantaran *item* *VIP* (apabila berhasil melakukan *drop-off* sebanyak 2 kali, maka perintah dapat dipakai 2 kali)

```
Lokasi Mobita: M (8, 2)
Waktu: 22
Uang: 2200

ENTER COMMAND: RETURN
RETURN selected.

Efek 'Return to Sender' digunakan!
Item teratas pada bag telah dikembalikan ke pengirim
Sisa penggunaan: 1 kali

Lokasi Mobita: M (8, 2)
Waktu: 22
Uang: 2200

ENTER COMMAND: IN_PROGRESS
IN_PROGRESS selected.

Tidak ada item yang sedang diantarkan!
```

Apabila pemain berhasil mengantarkan *item* bertipe *perishable*, kapasitas tas dari pemain akan bertambah 1.

```
Ada pesanan yang dapat di-dropoff di sini!
Ada pesanan VIP yang harus didahulukan!
ENTER COMMAND: DROP_OFF
DROP_OFF selected.

Pesanan Perishable Item berhasil diantarkan!
Uang yang didapatkan: 400 Yen
Kapasitas tas bertambah 1!

Lokasi Mobita: N (8, 6)
Waktu: 20
Uang: 2600
Ada pesanan yang dapat di-dropoff di sini!
Ada pesanan VIP yang harus didahulukan!
ENTER COMMAND: IN_PROGRESS
IN_PROGRESS selected.

Kapasitas tas: 7
Ada 2 pesanan yang sedang diantarkan :
1. Normal Item (Tujuan : N)
2. Heavy Item (Tujuan : N)
```

## 6.8 Pengujian Gadget

Apabila *gadget* Pintu Kemana Saja berhasil digunakan, maka pemain dapat berpindah ke satu tempat tanpa menambah waktu.

```
Ada pesanan VIP yang harus diantarkan!
ENTER COMMAND: INVENTORY
INVENTORY selected.

1. Kain Pembungkus Waktu
2. Senter Pembesar
3. Pintu Kemana Saja
4. Mesin Waktu
5. Senter Pengecil
Gadget mana yang ingin digunakan?
(Ketik 0 jika ingin kembali)

ENTER OPTION: 3
Gadget 'Pintu Kemana Saja' berhasil digunakan!
Gunakan perintah 'MOVE' untuk berpindah tanpa menambah waktu!

Ada pesanan VIP yang harus diantarkan!
ENTER COMMAND: MOVE
MOVE selected.

Lokasi-lokasi yang dapat diakses:
1. C (1, 9)
2. E (2, 3)
3. F (3, 1)
Posisi yang dipilih? (ketik 0 jika ingin kembali)
Gadget 'Pintu Kemana Saja' akan digunakan!

ENTER COMMAND: 3

Mobita sekarang berada di titik F(3, 1)!
Waktu : 10 (+0)
Lokasi Mobita: F (3, 1)
Waktu: 10
Uang: 2200
```

Apabila *gadget* Kain Pembungkus Waktu berhasil digunakan, maka item teratas di tas (dalam konteks ini item paling bawah di in-progress list) akan dicek. Apabila item tersebut *perishable*, maka waktu *perishable* akan diulang dari awal. Jika tidak, *gadget* akan terpakai sia-sia.

```
Ada pesanan VIP yang harus diantarkan!
ENTER COMMAND: INVENTORY
INVENTORY selected.

1. Kain Pembungkus Waktu
2. Senter Pembesar
3. -
4. Mesin Waktu
5. Senter Pengecil
Gadget mana yang ingin digunakan?
(Ketik 0 jika ingin kembali)

ENTER OPTION: 1
Gadget 'Kain Pembungkus Waktu' berhasil digunakan!

Ada pesanan VIP yang harus diantarkan!
ENTER COMMAND: IN_PROGRESS
IN_PROGRESS selected.

Ada 3 pesanan yang sedang diantarkan :
1. Normal Item (Tujuan : N)
2. Heavy Item (Tujuan : N)
3. Perishable Item (Tujuan : N, Waktu Sisa : 6)
Ada pesanan VIP yang harus diantarkan!
ENTER COMMAND: IN_PROGRESS
IN_PROGRESS selected.

Ada 3 pesanan yang sedang diantarkan :
1. Normal Item (Tujuan : N)
2. Heavy Item (Tujuan : N)
3. Perishable Item (Tujuan : N, Waktu Sisa : 10)
```

Apabila *gadget* 'Senter Pembesar' digunakan, maka kapasitas tas akan membesar sebanyak dua kali lipat.

```
Ada pesanan VIP yang harus diantarkan!
ENTER COMMAND: INVENTORY
INVENTORY selected.

1. Mesin Waktu
2. Senter Pengecil
3. Senter Pembesar
4. -
5. -
Gadget mana yang ingin digunakan?
(Ketik 0 jika ingin kembali)

ENTER OPTION: 3
Gadget 'Senter Pembesar' berhasil digunakan!
Kapasitas tas Mobita membesar menjadi dua kali lipat...

Ada pesanan VIP yang harus diantarkan!
ENTER COMMAND: IN_PROGRESS
IN_PROGRESS selected.

Kapasitas tas: 6
Ada 3 pesanan yang sedang diantarkan :
1. Normal Item (Tujuan : N)
2. Heavy Item (Tujuan : N)
3. Perishable Item (Tujuan : N, Waktu Sisa : 6)
```



Penggunaan *gadget* ‘Senter Pengecil’ akan mengurangi beban satu *heavy item* apabila *heavy item* ada di posisi teratas tas.

```

ENTER COMMAND: INVENTORY
INVENTORY selected.

1. -
2. -
3. -
4. Mesin Waktu
5. Senter Pengecil
Gadget mana yang ingin digunakan?
(Ketik 0 jika ingin kembali)

ENTER OPTION: 5
Gadget 'Senter Pengecil' berhasil digunakan!
Beban dari satu heavy item berhasil dikurangi.

```

```

ENTER COMMAND: MOVE
MOVE selected.

Lokasi-lokasi yang dapat diakses:
1. I (4, 5)
2. J (5, 12)
3. K (6, 3)
4. N (8, 6)
5. P (9, 13)
Posisi yang dipilih? (ketik 0 jika ingin kembali)
ENTER COMMAND: 4

Mobita sekarang berada di titik N(8, 6)!
Waktu : 22 (+2)

```

```

ENTER COMMAND: MOVE
MOVE selected.

Lokasi-lokasi yang dapat diakses:
1. K (6, 3)
2. L (7, 10)
3. M (8, 2)
4. Q (10, 3)
Posisi yang dipilih? (ketik 0 jika ingin kembali)
ENTER COMMAND: 1

Mobita sekarang berada di titik K(6, 3)!
Waktu : 23 (+1)

```

Penggunaan *gadget* Mesin Waktu akan mengurangi waktu sebanyak 50. *Item* yang sudah masuk ke to-do list tidak akan dimasukkan lagi. *Item* yang belum dimasukkan harus menunggu sampai waktu selanjutnya muncul.

```

ENTER COMMAND: INVENTORY
INVENTORY selected.

1. -
2. -
3. -
4. Mesin Waktu
5. -
Gadget mana yang ingin digunakan?
(Ketik 0 jika ingin kembali)

ENTER OPTION: 4
Gadget 'Mesin Waktu' berhasil dipakai!
Waktu berkurang sebanyak 50...
Lokasi Mobita: F (3, 1)
Waktu: 0
Uang: 2600

```

## 7 Test Script

No.	Fitur yang Dites	Tujuan Testing	Langkah-Langkah Testing	Input Data Test	Hasil yang Diharapkan	Hasil yang Keluar
1	Loading file konfigurasi dan save file	Menguji apakah loading dapat dilakukan	Mulai program, ketikkan perintah “NEW GAME” untuk file konfig atau “LOAD GAME” untuk file save, lalu ketikkan direktori/nama file save.	NEW GAME/LOAD GAME, <direktori/nama file konfig>	Permainan dimulai menggunakan data yang ada di file	Permainan dimulai menggunakan data yang ada di file
2	MOVE	Mengecek apakah perpindahan dapat dilakukan	Mulai game, gunakan perintah “MOVE” dan pilih tempat yang digunakan	MOVE, (angka bebas selama ada di opsi)	Pesan berhasil berpindah tempat, lokasi diupdate	Pesan berhasil berpindah tempat, lokasi diupdate
3	Pengambilan barang	Mengecek apakah barang dapat diambil	Mulai game, gunakan perintah MOVE untuk bergerak sampai lokasi pickup, lalu gunakan perintah pickup	MOVE (hingga ada pilihan ada item yang bisa dipickup), PICK_UP	Item masuk ke in-progress list dan bag apabila tidak ada item VIP yang harus didahulukan atau tas penuh	Item masuk ke in-progress list dan bag apabila tidak ada item VIP yang harus didahulukan atau tas penuh
4	Pengantaran barang	Mengecek apakah barang dapat diantarkan	Mulai game, pastikan ada barang yang diantar, gunakan perintah MOVE untuk bergerak sampai lokasi dropoff, lalu gunakan perintah dropoff	MOVE (hingga ada pilihan ada item yang bisa dipickup), DROP_OFF	Item teratas di tas (terbawah di in-progress list) akan dikirimkan, uang akan masuk dan efek barang akan teraktivasi	Item teratas di tas (terbawah di in-progress list) akan dikirimkan, uang akan masuk dan efek barang akan teraktivasi
5	ToDo dan In-Progress List	Mengecek daftar to-do list dan in-progress list	Mulai game, gunakan perintah TO_DO dan IN_PROGRESS	TO_DO, IN_PROGRESS	Daftar pesanan akan bertambah seiring berjalannya waktu, item teratas di bag akan diwakili sebagai item Terbawah di inprogress list	Daftar pesanan akan bertambah seiring berjalannya waktu, item teratas di bag akan diwakili sebagai item Terbawah di inprogress list
6	Pembelian dan penggunaan gadget	Menguji apakah gadget bisa dibeli dan terimplementasi	Pastikan sedang berada di HQ, gunakan perintah BUY untuk membeli gadget dan	Perintah BUY/INVENTORY, angka (1 – 5)	Gadget yang dibeli akan bisa dipakai di <i>inventory</i> , dan gadget dibeli apabila uang	Gadget yang dibeli akan bisa dipakai di <i>inventory</i> , dan gadget dibeli apabila uang

No.	Fitur yang Dites	Tujuan Testing	Langkah-Langkah Testing	Input Data Test	Hasil yang Diharapkan	Hasil yang Keluar
			INVENTORY untuk menggunakan gadget		pemain mencukupi	pemain mencukupi
7	Efek Drop-Off Barang	Menguji apakah efek dari dropoff item tertentu berjalan	Pastikan telah melakukan dropoff Untuk heavy item: ability speedboost akan menyala Untuk perishable item: kapasitas tas bertambah 1 Untuk VIP item: bisa menggunakan perintah RETURN	DROP_OFF (tipe item yang diinginkan)	Efek akan langsung terimplementasi di permainan	Efek akan langsung terimplementasi di permainan
8	Pengujian Gadget	Menguji apakah efek gadget langsung terpakai	Gunakan perintah INVENTORY	INVENTORY	Efek akan langsung terpakai, gadget yang dipakai hilang dari inventory	Efek akan langsung terpakai, gadget yang dipakai hilang dari inventory

## 8 Pembagian Kerja dalam Kelompok

No	NIM	Nama	Pembagian kerja
1	13520166	Raden Rifqi Rahman	Pembuatan ADT dasar + driver ADT Pembuatan <i>script</i> kompilasi Pembuatan <i>groundwork</i> pengerjaan dan <i>state</i> algoritma Pengecekan ulang algoritma <i>Debugging</i> dan <i>update</i> source code
2	13520124	Owen Christian Wijaya	Integrasi algoritma ke program utama Algoritma fungsi buy dan inventory <i>Debugging</i> dan <i>update</i> source code Konfigurasi <i>loading</i> awal dan <i>save file</i> Pembagian tugas dan
3	13520121	Nicholas Budiono	Algoritma move Petanggungjawab laporan Pengecekan ulang algoritma
4	13520120	Afrizal Sebastian	Algoritma in-progress dan pick-up <i>Debugging</i> dan <i>update</i> source code Pengecekan ulang algoritma
5	13520127	Adzka Ahmadetya Zaidan	Algoritma to-do dan drop-off Notulis asistensi Pengecekan ulang algoritma

## 9 Lampiran

### 9.1 Deskripsi Tugas Besar

Dengan adanya pandemi COVID-19, usaha orang tua Mobita mengalami penurunan pendapatan. Setelah diringankannya PPKM, Mobita ingin membantu orang tuanya mendapat penghasilan menjadi seorang kurir. Tugas kalian adalah membantu Mobita membuatkan aplikasi yang akan membantunya melacak pesanan, navigasi peta, mengambil dan menurunkan barang.

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

### 9.2 Notulen Rapat

#### Rapat Awal

- *Outline* algoritma awal (lebih spesifik dijelaskan di Bab 5)
- Pembagian tugas dan pengerjaan ADT lanjutan
- Pengerjaan *gadget* dan ADT *State*

#### Asistensi 1

- Detil tambahan: Baris ke 0 dan kolom ke 0 itu adalah HQnya untuk matrix adjacency
- Saat pick up, menambah list in progress. Saat drop off, menghapus list in progress dan to do yang sesuai.
- Mesin waktu tidak berpengaruh ke perishable item.
- Return to Sender: sekali pakai, tunggu dapet lagi (one use) bukan ability permanent, kayak speed boost.

#### Asistensi 2

- Untuk debug, kalau mau, bikin input dari file biar bisa di ulang-ulang, terus bandingin sama yang benarnya.
- Bukan multiplier, heavy item timenya +1, nge-stack overall udah oke, udah cukup.
- Paling tinggal debugging. Jangan lupa laporannya.
- ADTnya disimpan yang di centralized, bagus sih, jadi rapi.
- Log activity gak usah detail, biasa aja, NIM ini ngerjain apa, tidak perlu detil sampe waktu jam segini ngerjain apa aja.

### 9.3 Log Activity Anggota Kelompok

*Log activity* dilihat dari sejarah *push* di *GitHub* dan pengerjaan di grup koordinasi.

13520166 – Raden Rifqi Rahman

- 24 Oktober 2021: Dokumentasi dan implementasi model ADT
- 26 – 28 Oktober 2021: Pembaruan dan bugfix ADT, script kompilasi
- 2 November 2021: Integrasi program lainnya ke program utama
- 14 – 16 November: Debugging, merapikan ulang file

13520124 – Owen Christian Wijaya

- 22 Oktober 2021: Pembuatan *parser* file konfigurasi dan struktur program utama

STEI- ITB	IF2110-TB-10-03	Halaman 21 dari 22 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.		

- 24 Oktober 2021: Pembagian tugas, algoritma BUY dan INVENTORY
- 26 Oktober 2021: Algoritma berhasil diimplementasikan
- 1 November 2021: Integrasi program lainnya ke program utama
- 5 November 2021: Integrasi ulang
- 9 November: algoritma RETURN diimplementasikan
- 13 – 16 November: algoritma save file diimplementasikan, debugging ulang

13520121 – Nicholas Budiono

- 24 Oktober 2021: Pembagian tugas, algoritma MOVE
- 27 Oktober 2021: Algoritma MOVE berhasil diimplementasikan
- 10 November 2021: Pembuatan laporan awal

13520120 – Afrizal Sebastian

- 24 Oktober 2021: Pembagian tugas, algoritma IN\_PROGRESS dan PICK\_UP
- 29 Oktober 2021: Algoritma berhasil diimplementasikan
- 5 November 2021: Bugfix
- 10 November 2021: Pembuatan laporan awal

13520127 – Adzka Ahmadetya Zaidan







- 24 Oktober 2021: Pembagian tugas, algoritma DROP\_OFF dan TO\_DO
- 29 Oktober 2021: Algoritma berhasil diimplementasikan
- 10 November 2021: Pembuatan laporan awal

**Form Asistensi Tugas Besar  
IF2110/Algoritma dan Struktur Data  
Sem. 1 2021/2022**

No. Kelompok/Kelas : 10/03  
Nama Kelompok : #1 Mobita Fan Club  
Anggota Kelompok (Nama/NIM) :  
1. 13520166 / Raden Rifqi Rahman  
2. 13520124 / Owen Christian Wijaya  
3. 13520121 / Nicholas Budiono  
4. 13520120 / Afrizal Sebastian  
5. 13520127 / Adzka Ahmadetya Zaidan  
6. -  
  
Asisten Pembimbing : Matthew Kevin Amadeus

---

# Asistensi I

<b>Tanggal : Rabu, 27 Oktober 2021</b>	<b>Catatan Asistensi:</b>
<b>Tempat : Google Meet</b> <b>(<a href="https://meet.google.com/mno-ppzu-vof">https://meet.google.com/mno-ppzu-vof</a>)</b>	
<b>Kehadiran Anggota Kelompok:</b> No NIM Tanda tangan  1  2  3  4  5  6	<p>Notasi Algoritmik belum tau bakal perlu atau enggak, tetapi jangan dulu dipikirin, selesaikan dulu programnya, ADTnya, etc.</p> <p>Detil tambahan: Baris ke 0 dan kolom ke 0 itu adalah HQnya untuk matrix adjacency</p> <p>Saat pick up, nambah list in progress          Saat drop off, menghapus list in progress dan to do yang sesuai.</p> <p>Kalau konfigurasinya gak match gimana? Itu wordingnya salah, terserah jumlah nge loadnya, boleh 1, boleh 2.</p> <p>Mesin waktu gak berpengaruh ke persihable item.          Return to Sender: sekali pakai, tunggu dapet lagi (one use) bukan ability permanent, kayak speed boost.</p>
	<b>Tanda Tangan Asisten:</b> 



## Asistensi II

<b>Tanggal :</b>	<b>Catatan Asistensi:</b>
<b>Tempat :</b>	
<b>Kehadiran Anggota Kelompok:</b>	
<div style="text-align: center;"> No  NIM  Tanda tangan   1    2    3    4    5    6 </div>	
	<b>Tanda Tangan Asisten:</b>

### Asistensi III

<b>Tanggal :</b>	<b>Catatan Asistensi:</b>
<b>Tempat :</b>	
<b>Kehadiran Anggota Kelompok:</b> <div style="text-align: center;"> No  NIM  Tanda tangan </div> <div style="text-align: center;">1</div> <div style="text-align: center;">2</div> <div style="text-align: center;">3</div> <div style="text-align: center;">4</div> <div style="text-align: center;">5</div> <div style="text-align: center;">6</div>	
	<b>Tanda Tangan Asisten:</b>





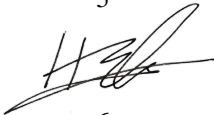
## Asistensi IV

<b>Tanggal :</b>	<b>Catatan Asistensi:</b>
<b>Tempat :</b>	
<b>Kehadiran Anggota Kelompok:</b>	
No	
NIM	
Tanda tangan	
1	
2	
3	
4	
5	
6	
	<b>Tanda Tangan Asisten:</b>

## Asistensi V

<b>Tanggal :</b>	<b>Catatan Asistensi:</b>
<b>Tempat :</b>	
<b>Kehadiran Anggota Kelompok:</b>	
No	
NIM	
Tanda tangan	
1	
2	
3	
4	
5	
6	
	<b>Tanda Tangan Asisten:</b>

## Asistensi II

Tanggal : 9 November 2021		Catatan Asistensi:  Untuk debug, kalau mau, bikin input dari file biar bisa di ulang-ulang, terus bandingin sama yang benarnya.  Bukan multiplier, heavy item timenya +1, nge-stack  Overall udah oke, udah cukup. Paling yang nge bug tinggal dibetulin. Jangan lupa laporannya, ADTnya disimpen yang di centralized, bagus sih, jadi rapi.  Log activity gak usah detail, biasa aja, NIM ini ngerjain apa, gak perlu detil sampe waktu jam segini ngerjain apa aja.
Tempat : Google Meet		
Kehadiran Anggota Kelompok: <div>No</div> <div>NIM</div> <div>Tanda tangan</div> <div>1</div> <div></div> <div>2</div> <div></div> <div>3</div> <div></div> <div>4</div> <div></div> <div>5</div> <div></div> <div>6</div>		
		Tanda Tangan Asisten:  