

LAPORAN TUGAS BESAR

IF2111 Algoritma dan Struktur Data

BNMO


Dipersiapkan oleh:

Kelompok 8

1. Michael Sihotang (18221054)
2. Muhammad Dastin Fauzi (18221062)
3. Imanuel Raditya (18221112)
4. Miralistya Cahya Fatimah (18221116)
5. Seren Elizabeth Siahaan (18221160)

Sekolah Teknik Elektro dan Informatika - Institut Teknologi Bandung

Jl. Ganesha 10, Bandung 40132

	Sekolah Teknik Elektro dan Informatika ITB	Nomor Dokumen		Halaman
		<i>IF2111-TB1-08</i>		35
		<i>Revisi</i>	<i>01</i>	<i>28 Oktober 2022</i>

Daftar Isi

1 Ringkasan	3
2 Penjelasan Tambahan Spesifikasi Tugas	4
2.1 Fitur Baner	4
2.2 Fitur Quit	5
2.3 Fitur Game Tambahan	5
3 Struktur Data	6
3.1 Struktur Data Array	6
3.2 Struktur Data Mesin Karakter	6
3.3 Struktur Data Mesin Kata	7
3.4 Struktur Data Queue	8
3.5 Struktur Data Map	8
3.6 Struktur Data Stack	9
4 Program Utama	9
5 Algoritma-Algoritma Menarik	10
5.1 Pemakaian fungsi rand() dan srand()	10
5.2 Pemakaian prosedur STARTCOMMAND	11
6 Data Test	11
6.1 Data Test START	11
6.2 Data Test LOAD	12
6.3 Data Test CREATE GAME	14
6.4 Data Test LIST GAME	14
6.5 Data Test DELETE GAME	15
6.6 Data Test QUEUE GAME	16
6.7 Data Test SAVE	16
6.8 Data Test PLAY GAME	17
6.9 Data Test Command Lain	17
6.10 Data Test Diner Dash	17
6.11 Data Test SKIP GAME	21
6.12 Data Test QUIT	21
6.13 Data Test Help	24
6.14 Data Test RNG	24
6.15 Data Test BONUS	26
7 Test Script	27
8 Pembagian Kerja dalam Kelompok	29
9 Lampiran	29
9.1 Deskripsi Tugas Besar 1	29
9.2 Notulen Rapat	31
9.3 Log Activity Anggota Kelompok	34

1 Ringkasan

BNMO merupakan suatu program robot video *game* yang dapat menjalankan suatu permainan di dalamnya. Pada BNMO ini ada beberapa hal yang dapat dilakukan oleh pengguna yaitu memainkan *game*, menambahkan *game*, menghapus *game*, dan mengurutkan *game* yang akan dimainkan. Semua perubahan yang kita lakukan terhadap program *game* pada saat bermain seperti menambahkan dan juga menghapus *game* dapat disimpan pada sebuah *file* dengan *type* txt. Berikut adalah penjelasan mengenai cara kerja dari program BNMO ini. Pada program ini, terdapat dua *game* yang pasti bisa dimainkan oleh pengguna, yaitu RNG dan DINER DASH. Namun, pada sistem ini tentunya ada fitur untuk melakukan pembuatan *game* yang baru sehingga *game* yang lain juga dapat dimainkan dalam sistem ini.

Pada saat awal memulai program BNMO, program akan menampilkan sebuah *main menu* yang berisi *welcome page*. Pada fase tersebut, pengguna diberi dua pilihan opsi mengenai cara untuk mulai menjalankan program *game*. Opsi tersebut adalah START dan LOAD. *Command* START atau LOAD adalah *command* pertama yang harus diberikan oleh pengguna. Para pengguna dapat memilih satu di antara dua *command* tersebut. Ketika pengguna memberikan *command* lain (selain START dan LOAD) sebagai perintah pertama, program akan menampilkan bahwa masukan dari pengguna salah, dan akan diminta untuk memberikan perintah kembali sampai perintah yang diberikan oleh pengguna adalah salah satu di antara START dan LOAD.

Perbedaan di antara dua opsi perintah pertama (START dan LOAD) terletak pada kondisi *state* yang digunakan pada *game*. Ketika pengguna memilih START untuk memulai sebuah *game* lalu menekan ENTER, maka program akan membaca *file* konfigurasi *default* yang berisi *list game* yang dapat dimainkan oleh pengguna. Isi dari *list game* tersebut adalah *game - game* yang merupakan ‘bawaan’ *standar* dari program BNMO. Sedangkan ketika pengguna memilih LOAD untuk memulai sebuah *game*, pengguna harus melengkapi perintah LOAD tersebut dengan sebuah nama *file* yang merepresentasikan suatu *save file* yang ingin dibuka. Setelah menekan ENTER, maka *file* <filename> yang berisi *list game* yang dapat dimainkan oleh pengguna akan dibaca. Isi dari *list game* tersebut belum tentu sama dengan *list game default*. Bisa saja terdapat beberapa *game* tambahan yang pernah ditambahkan oleh pengguna pada *list game* tersebut. Dengan kata lain, *file* yang akan dibuka pada *command* LOAD adalah suatu *state* pada *game* yang sudah pernah disimpan oleh pengguna.

Saat pengguna telah memilih cara dalam memulai menjalankan program, pengguna dapat menjalankan fitur-fitur yang terdapat pada program BNMO melalui pemanggilan *command*. Daftar *command* yang dapat diberikan adalah antara lain CREATE GAME, LIST GAME, QUEUE GAME, PLAY GAME, SKIP GAME, DELETE GAME, SAVE, HELP, dan QUIT. Apabila pengguna memberikan *command* selain *command-command* yang telah disebutkan, maka masukan dari pengguna tersebut akan dianggap tidak *valid* dan tidak akan diproses oleh program. Penjelasan singkat mengenai *command* fitur - fitur pada program BNMO adalah sebagai berikut.

Pertama, *Command* CREATE GAME adalah perintah yang digunakan untuk menambahkan *game* baru pada *list game*. Pengguna akan diminta untuk memasukkan nama *game* yang akan ditambahkan. Selanjutnya, *command* LIST GAME adalah *command* yang digunakan untuk menampilkan daftar *game* yang dapat dimainkan oleh pengguna. Ketiga, *command* QUEUE GAME yaitu perintah yang digunakan untuk mendaftarkan permainan kedalam *list* antrian *game* yang akan dimainkan oleh pengguna. *List* dalam *queue* akan hilang ketika pengguna menjalankan *command* QUIT. Selanjutnya ada *command* PLAY GAME yang digunakan untuk memainkan permainan yang terdapat pada urutan pertama antrian *game* milik

pengguna. Pada program ini, terdapat dua *game* yang pasti bisa dimainkan oleh pengguna, yaitu RNG dan DINER DASH. Selanjutnya, terdapat *command* SKIP GAME yang digunakan untuk melangkahi *n* buah *game* pada antrian untuk memainkan *game* selanjutnya. *Command* keenam adalah *command* DELETE GAME yang digunakan untuk menghapus sebuah *game* dari *list game*. *Command* ketujuh adalah SAVE. *Command* ini digunakan untuk menyimpan *state game* pemain saat ini ke dalam suatu *file*. *Command* ini memiliki satu argumen yaitu *filename* yang akan disimpan dalam disk. Selain itu, terdapat *command* HELP yang digunakan untuk memberikan penjelasan atau bantuan mengenai *command* - *command* yang terdapat pada program. Terakhir, terdapat *command* QUIT yang digunakan untuk keluar dari program BNMO.

Program BNMO ini menggunakan bahasa pemrograman C dan akan menerapkan beberapa struktur data yang sesuai. ADT yang digunakan pada program ini adalah ADT Mesin Karakter dan Mesin Kata, ADT Queue, dan ADT Array. ADT array yang kami gunakan adalah array yang bertipe dinamis dan berupa array of Word. Begitu juga dengan ADT Queue yang bertipe dinamis dan berupa queue of Word. Hal - hal lain mengenai ADT yang digunakan pada program ini akan dijelaskan dengan lebih *detail* pada poin nomor 3. Pada program BNMO yang kami buat juga terdapat beberapa fitur tambahan yang akan dijelaskan pada poin nomor dua. Selain itu, terdapat program utama atau *main* yang digunakan sebagai wadah sinkronisasi antar program *command* sehingga program BNMO dapat dijalankan secara utuh dan selaras. Hal - hal mengenai program utama akan dijelaskan secara lebih lanjut pada poin nomor empat, diikuti dengan Data Test dan juga Test Script mengenai program BNMO secara keseluruhan yang akan dijabarkan pada poin nomor enam dan tujuh.

2 Penjelasan Tambahan Spesifikasi Tugas

2.1 Fitur Baner

Fitur ini sebenarnya hanya fitur kosmetik untuk memperindah tampilan sehingga benar-benar merepresentasikan permainan yang telah dirancang. Dapat dilihat pada gambar 2.1.1, terdapat judul dari permainan ini.

Gambar 2.1.1 Opening banner game



Pada fitur ini juga menyampaikan bahwa *command* pertama yang bisa digunakan hanya START atau LOAD.

2.2 Fitur Quit

Pada fitur ini kami menambahkan bahwa pengguna program dapat melakukan SAVE sebelum keluar dari program dan dapat memilih tidak melakukan save juga. Maka, ketika pengguna bisa bermain kembali melanjutkan apa yang sudah ia mainkan sebelumnya.

Gambar 2.2.1 Tambahan fitur Quit

```
Masukkan command: QUIT
Apakah anda ingin melakukan SAVE sebelum keluar dari program? (Y/N): Y
Masukkan nama file: Test.txt
Save file berhasil disimpan

Anda keluar dari game BNMO
Bye bye ...
```

2.3 Fitur Game Tambahan

Pada bagian game, kami menambahkan game tambahan yang bernama "Tower of Hanoi". Pada game ini kami memanfaatkan struktur data STACK yang direpresentasikan dengan struktur berkait.

Gambar 2.3.1 Tampilan awal Tower of Hanoi

```
Masukkan command: PLAY GAME
Berikut adalah daftar antrian game-mu
1. TOWER OF HANOI

Loading TOWER OF HANOI . . .

=====
- TOWER OF HANOI -
=====

Masukkan jumlah cakram: 3

-----

Tower 1: [3, 2, 1]
Tower 2: []
Tower 3: []

=
===
=====
1      2      3
```

Gambar 2.3.2 Tampilan akhir Tower of Hanoi

```
Memindahkan cakram dari tower (1 - 3): 1
Ke tower (1 - 3): 3

-----

Tower 1: []
Tower 2: []
Tower 3: [3, 2, 1]

=
===
=====
1      2      3

Jumlah langkah: 7

Selamat, Anda berhasil menyelesaikan Tower of Hanoi!
Skor Anda: 100
```

3 Struktur Data (ADT)

Pada program yang kami buat, kami menggunakan beberapa struktur data (ADT) untuk menyelesaikan permasalahan yang ada dan membuat program BNMO menjadi program yang dapat berjalan dengan baik dan tepat. Beberapa struktur data (ADT) yang kami gunakan adalah ADT Array, ADT Mesin Karakter, ADT Mesin Kata, ADT Queue, ADT Map, dan ADT Stack.

3.1 Struktur Data Array

Pada pengerjaan program BNMO ini, kami menggunakan struktur data *Array*. Struktur ini diberi nama struktur *ArrayDin* yang menunjukkan bahwa struktur *array* yang kami gunakan adalah struktur *array* yang direpresentasikan secara eksplisit dengan alokasi dinamik. Hal tersebut menandakan bahwa ada proses alokasi dan dealokasi yang dilakukan. Struktur *array* yang kami gunakan menyimpan beberapa data seperti tabel penyimpanan elemen *ElType*, *Capacity*, dan *Neff*. Tabel penyimpanan elemen *ElType* disini adalah tempat menyimpan elemen - elemen pada *array*. Elemen pada *ArrayDin* disimpan dalam tipe *Word* atau kata, hasil bentukan dari ADT Mesin Kata. Data *Capacity* pada struktur *ArrayDin* bertipe *integer* dan merupakan sebuah bilangan yang menandakan kapasitas dari *ArrayDin* yang tersedia. Sedangkan data *Neff* pada struktur *ArrayDin* memiliki tipe *integer* dan berperan sebagai penanda jumlah elemen efektif pada *array*. Struktur data *array* dinamis ini tersimpan pada *file* source/ADT/arraydin.c dan pada source/ADT/arraydin.h untuk header program.

Struktur data *ArrayDin* diterapkan pada program BNMO yang kami buat dalam persoalan penyimpanan *list game*. Oleh karena itu, *ArrayDin* sudah mulai berperan sejak awal pengguna memberikan *command* START / LOAD hingga akhir yaitu pengguna memberikan *command* QUIT. Pada saat *command* START / LOAD, program akan membaca daftar game yang terdapat pada *file*, kemudian akan memasukkannya ke dalam *list game* yang berstruktur *ArrayDin* tersebut. *ArrayDin* juga menyelesaikan persoalan *command* LIST GAME yaitu dengan menampilkan semua isi elemen yang tersimpan pada *list game* berbentuk *ArrayDin*. Selain itu, *ArrayDin* juga digunakan dalam *command* CREATE GAME dan juga DELETE GAME, yaitu dengan menambahkan sebuah *game* menjadi elemen dari *list game* dan juga menghapuskan sebuah *game* dari *list game*.

Kami memilih untuk menggunakan struktur data *Array* dengan tipe *array* dinamis ini dengan alasan karena struktur ini dapat menjadi tabel penyimpanan yang tepat bagi *list game* yang tersedia pada program. Daftar game tersebut tersimpan secara terurut pada tabel penyimpanan dan untuk batasan jumlah penyimpanan cenderung lebih *fleksible* karena dapat melakukan alokasi dan dealokasi sehingga lebih efektif.

3.2 Struktur Data Mesin Karakter

Program BNMO yang kami buat juga memanfaatkan pengaplikasian sebuah struktur data bertipe Mesin Karakter. Struktur ini berperan dalam pembacaan sebuah input baik dari pengguna secara langsung, maupun pembacaan dari sebuah *file*. Pada ADT Mesin Karakter yang memanfaatkan struktur data mesin karakter, dilakukan proses berupa pembacaan suatu pita karakter yang diperoleh dari sebuah *file* ataupun masukan dari pengguna (*stdin*) yang kemudian akan disimpan pada sebuah *variabel* *currentChar* yang bertipe *character*. Proses pembacaan tersebut akan berlanjut hingga suatu kondisi berhenti yang telah ditentukan. Struktur data mesin

karakter ini tersimpan pada *file* `source/ADT/mesinkarakter/mesinkarakter.c` dan pada `source/ADT/mesinkarakter/mesinkarakter.h` untuk *header* program.

Penerapan struktur data Mesin Karakter ini terletak pada proses pembacaan isi dari *file* yang berisi *list game* pada *command* START maupun LOAD, namun pada proses ini Mesin Karakter perlu modifikasi tambahan yang akan dilengkapi pada ADT Mesin Kata. Selain itu, struktur data Mesin Karakter juga dimanfaatkan pada proses pembacaan input dari pengguna yang diperlukan pada program utama untuk menerima sebuah *command* dari pengguna, maupun masukan tambahan dari pengguna yang diperlukan pada beberapa *command*. Beberapa masukan tambahan dari pengguna yang dimaksud adalah masukan nama *game* yang ingin ditambahkan pada *command* CREATE GAME, masukan nomor *game* yang akan dihapus pada *command* DELETE GAME, serta masukan nomor *game* yang akan ditambahkan pada antrian *game* pada QUEUE GAME.

Kami memilih untuk menggunakan struktur data Mesin Karakter dengan alasan karena pada program BNMO ini terdapat pembatasan dalam menerima input dari pengguna yaitu tidak dapat menggunakan *scanf*, maka mesin karakter ini menjadi solusi yang tepat untuk mengatasi persoalan menerima masukan dari pengguna, yaitu dengan menggunakan *fscanf* terhadap sebuah pita karakter yang merupakan *stdin*. Selain itu, konsep pembacaan *currentChar* pada pita karakter juga dinilai cocok dalam membaca sebuah barisan karakter baik dari *file* maupun *stdin*.

3.3 Struktur Data Mesin Kata

Sebagai pelengkap atau penyempurna dari struktur data Mesin Karakter, kelompok kami juga mengaplikasikan struktur data Mesin Kata. Struktur ini berperan dalam menyempurnakan peran Mesin Karakter dalam pembacaan sebuah masukan baik dari pengguna secara langsung, maupun dari *file txt*. Pada struktur data Mesin Kata ini, karakter per karakter yang dibaca pada mesin karakter akan disimpan pada tabel penyimpanan berupa Word (kata) yang terdiri dari tabel TabWord yaitu *string* kumpulan karakter dan juga *Length* yaitu *integer* penunjuk panjang *string*. Proses pembacaan pada mesin karakter tersebut akan berhenti pada suatu kondisi tertentu yaitu seperti MARK atau '.' dan juga ENTER '\n'. Pada proses pembacaan *file* atau dari *input* pengguna, kumpulan *character* yang dibaca akan secara bergantian menduduki variabel *currentWord* yang memiliki tipe Word. Melalui variabel tersebut, kata dapat diproses dengan lebih lanjut. Struktur data Mesin Kata ini tersimpan pada *file* `source/ADT/mesinkata/mesinkata.c` dan pada `source/ADT/mesinkata/mesinkata.h` untuk *header* program.

Penerapan struktur data Mesin Kata ini sama dengan penerapan struktur data Mesin Karakter karena keduanya saling melengkapi dan berbagi peran. Contoh penerapan yaitu pada pembacaan isi dari sebuah *file* dan pembacaan masukan dari pengguna, seperti pembacaan *command* yang diberikan pengguna, pembacaan nama *game* yang ingin ditambahkan pada *command* CREATE GAME, serta pembacaan nomor *game* yang diberikan pada *command* QUEUE GAME dan DELETE GAME.

Kami memilih untuk menggunakan struktur ini dengan alasan untuk menyempurnakan dan melengkapi peran dari struktur data Mesin karakter. Terlebih, dengan adanya struktur ini, masukan dari pengguna dapat diproses dengan lebih lanjut dan lebih mudah, seperti mengidentifikasi masukan *command* dari pengguna dengan menggunakan fungsi `isWordEqual` sehingga proses kondisional pada program utama dapat berjalan dengan lebih mudah.

3.4 Struktur Data Queue

Ada dua jenis ADT Queue yang kami gunakan dalam program BNMO ini. Hal ini dikarenakan ada dua tipe data yang digunakan yaitu word dan integer. Oleh karena itu, ADT QUEUE digunakan untuk tipe data word dan ADT QUEUE DINER DASH digunakan untuk tipe data integer. Berikut penjelasan dari kedua ADT tersebut.

a. ADT Queue

Program BNMO yang kami buat juga memanfaatkan ADT queue khususnya dengan tipe data word. Pada struktur data ini, definisi ADT queue direpresentasikan dengan array secara eksplisit dan alokasi dinamik. Hal ini berarti ada proses alokasi dan dealokasi yang digunakan. Tipe data Queue akan menyimpan empat jenis data yaitu tabel penyimpan elemen, alamat penghapusan, alamat penambahan, dan elemen maksimum queue. Tipe data dari tempat penyimpanan elemen adalah sebuah tipe word sedangkan tipe data dari alamat penghapusan dan penambahan adalah address. Tipe data address ini adalah tipe data integer yang menyimpan indeks tabel. Struktur data queue ini tersimpan pada file source/ADT/queue/queue.c dan source/ADT/queue/queue.h untuk header.

Struktur data queue ini kami gunakan pada bagian antrian game yang ingin dimainkan yaitu pada command QUEUE GAME. Antrian ini tentunya sama seperti prinsip queue yang menerapkan prinsip FIFO (First In First Out). Game yang sudah diinput pada queue game akan tersimpan menjadi elemen pertama dari queue. Maka, ketika pengguna program input command PLAY GAME, hal ini berarti secara otomatis memainkan game pada elemen pertama dari queue tersebut. Namun, apa sudah ada di antrian game juga bisa di skip dengan command SKIP GAME. Hal ini sebenarnya menerapkan prinsip queue juga yaitu dequeue (menghapus elemen awal). Setiap elemen queue ini juga tentunya adalah sebuah game dengan tipe data word.

b. ADT Queue DINER DASH

Program BNMO yang kami buat juga memanfaatkan ADT queue khususnya untuk digunakan dalam program DINER DASH. Pada struktur data ini juga memanfaatkan representasi queue dengan array statik dan array yang digunakan adalah array of integer. Hal ini berarti kapasitas dari arraynya sudah dipersiapkan yaitu 100. Tipe data Queueint ini akan menyimpan tiga jenis data. yaitu buffer yang menyimpan tipe data ElTypeInt, idxHead yang menyimpan tipe data integer, dan idxTail yang menyimpan tipe data integer. Tipe data ElTypeInt ini juga adalah tipe data yang menyimpan tiga jenis data yaitu cookTime, expTime, dan price. Ketiga jenis data ini menggunakan tipe data integer dan merupakan angka yang random dalam program.

Struktur data queue untuk representasi array of integer yang statik ini kami gunakan pada bagian antrian pada game DINER DASH. Pada game ini tentunya ada antrian pada apa yang harus melakukan proses COOK dan apa yang harus melakukan proses SERVE. Pada program ini juga bisa melakukan proses SKIP yang tentunya menerapkan prinsip queue yang akan menghapus elemen dari elemen paling awal. Hal ini tentu menggunakan perintah dequeue. Pada program ini juga ada menerapkan prinsip antrian dengan menyesuaikan ke sisa durasi memasak dan sisa ketahanan makanan.

3.5 Struktur Data Map

Program BNMO yang kami buat juga memanfaatkan ADT map khususnya untuk program DINER DASH. ADT map ini kami buat sebagai pelengkap program DINER DASH dimana terdapat keterbatasan jika hanya menggunakan ADT queue dengan array statik dalam

menjalankan program khususnya dalam memindahkan data pesanan ke dalam proses COOK maupun berpindah ke dalam proses SERVE. Kami tidak menggunakan ADT Queue secara utuh karena Queue hanya bisa digunakan untuk menghapus(dequeue) atau menambahkan(enqueue) berdasarkan konsep FIFO (First In First Out). Penggunaan ADT Map ini akan membuat program DINER DASH menjadi efisien dan fleksibel seperti contoh ketika user menginput command “COOK My” atau “SERVE Mx”, maka akan lebih mudah ketika menggunakan fungsi Insert dengan memanfaatkan key y atau x yang akan memanggil value berupa durasi masak, ketahanan kedalam tampilan masak ataupun tampilan sajian yang sesuai dengan apa yang ditampilkan dalam tampilan pesanan game. ADT Map yang kami buat juga menggunakan representasi array statis dengan kapasitas yang sudah ditentukan yaitu sebesar 10 elemen.

3.6 Struktur Data Stack

Pada bagian bonus dari tugas besar ini kami membuat game sederhana yang bernama "Tower of Hanoi". Pada game ini kami memanfaatkan ADT Stack yang direpresentasikan dengan list berkait. Kami memanfaatkan ADT stack karena pada game "Tower of Hanoi" memanfaatkan prinsip LIFO (Last In First Out). Game ini bekerja dengan cara berpindah dari tempat satu ke tempat yang lain.

4 Program Utama

Pada program buatan kami, alur program dimulai dari main.c. Program ini kemudian akan menampilkan banner program dan memerintahkan untuk input START atau LOAD. Apabila melakukan input START maka secara otomatis akan menampilkan HELP yang terdiri dari command-command yang dapat digunakan dalam program dan penjelasan dari program yang kami buat. Apabila melakukan input LOAD maka harus menginput file yang akan dijalankan. Hal ini akan melanjutkan permainan pada file yang sudah tersimpan terlebih dahulu.

Pada saat menginput START maka akan menampilkan command apa saja yang bisa di input oleh pengguna. Command yang dapat diinput antara lain :

- SAVE : menyimpan state game pemain saat ini
- CREATE GAME : menambahkan game baru pada daftar game
- LIST GAME : menampilkan daftar game yang disediakan
- DELETE GAME : menghapus sebuah game dari daftar game
- QUEUE GAME : mendaftarkan game ke dalam antrian game pribadi
- PLAY GAME : memainkan sebuah game
- SKIP GAME : melewati game sebanyak yang diinginkan
- QUIT : keluar dari program game

Pada saat melakukan QUEUE GAME maka akan menginput game yang ingin dimainkan maka pada saat PLAY GAME akan secara otomatis memainkan game pada urutan pertama. Game yang dapat dimainkan hanya game yang tersedia pada program yaitu RNG dan DINER DASH lalu untuk game yang lain akan memberikan pesan game tidak tersedia.

Setelah pengguna selesai menggunakan program tentu pengguna akan melakukan command QUIT untuk keluar dari program. Pada saat menginput command QUIT, maka akan ada dua pilihan yaitu untuk menyimpan file yang sudah diproses saat menggunakan program atau tidak menyimpan file.

5 Algoritma-Algoritma Menarik

5.1 Pemakaian fungsi rand () dan srand ()

Pada dasarnya, fungsi rand() merupakan fungsi yang dapat melakukan generate angka secara acak. Namun, jika dilakukan secara berurutan, angka yang dihasilkan akan sama setiap kali program dijalankan. Oleh karena itu, diperlukan fungsi srand() guna mengatur titik awal untuk menghasilkan sekumpulan *pseudo-random integers*. Srand() tidak mengembalikan nilai apapun, hanya menginisialisasi bilangan acak semu yang dihasilkan oleh fungsi rand(). Fungsi - fungsi ini dapat digunakan ketika program telah men-include library stdlib.h.

Gambar 5.1.1 Header file pada RNG

```
#include<stdio.h>
#include<stdlib.h>
#include<time.h>
```

Fungsi rand() dan srand() pada program BNMO digunakan pada program RNG dan DINER DASH. Pada program RNG adalah untuk mengambil angka sembarang dari rentang 1 sampai 500 untuk ditebak oleh pengguna program. Pada program DINER DASH juga digunakan untuk mengambil angka sembarang pada bagian durasi memasak, ketahanan, dan harga. Angka yang dihasilkan memiliki batas angka maksimal yang disesuaikan dengan file konfigurasi. Dalam implementasinya, fungsi rand() dan srand() didukung oleh fungsi time_t sebagai benih pengacakan yang "tidak dapat diprediksi", kecuali jika program dijalankan pada waktu yang sama. Fungsi time_t ini diperoleh melalui library time.h yang berfungsi untuk memanipulasi waktu.

Gambar 5.1.2 Implementasi rand() dan srand() pada program RNG

```
/*Mengambil angka random untuk ditebak di rentang 1 sampai 500*/
int randomx()
{
    int a;
    srand(time(NULL));
    a = rand() % (500 + 1 - 1) + 1;
    return a;
}
```

Gambar 5.1.3 Implementasi rand() pada program DINER DASH

```
int randomNumber(int max, int min){
    return ( (rand() % max) + min);
}
```

Algoritma - algoritma ini dianggap menarik karena dengan adanya keterlibatan srand() dan time_t, ternyata dapat memberikan perbedaan hasil yang sangat signifikan bila dibandingkan dengan fungsi yang hanya menggunakan fungsi rand() saja. Hal ini terlihat pada implementasi program RNG dan DINER DASH. Pada program RNG, hasil proses generate diharapkan selalu acak setiap waktunya sehingga diperlukan fungsi time_t untuk memanipulasi waktu. Sebaliknya, program DINER DASH digunakan sebagai implementasi output acak yang memiliki value yang sama. Oleh karena itu, pada program DINER DASH fungsi yang digunakan hanya fungsi rand() saja.

5.2 Pemakaian prosedur STARTCOMMAND ()

Dengan dibatasinya penggunaan fungsi *scanf* untuk membaca *input* dari user, kami membuat tipe data bentukan Word untuk menyimpan *input* kata dan pengelolaan *input* dilakukan melalui penggunaan sejumlah primitif yang dihasilkan dari ADT mesin kata dan mesin karakter untuk mengakuisisi kata yang menjadi command dan masukan.

Oleh karena itu, penerimaan input dilakukan melalui primitif STARTCOMMAND untuk menerima masukan command yang ingin dipanggil oleh pengguna yang kemudian di copy dalam *currentCommand*. Akuisisi command serta penyimpanan command dalam *currentCommand* dilakukan melalui primitif CopyCommand. Pengelolaan tipe data Word menjadi bentuk lainnya seperti integer yang digunakan pada command SKIP GAME, QUEUE GAME, dan DELETE GAME serta string seperti masukan nama save file untuk command SAVE dan LOAD dilakukan melalui primitif wordToInt, wordToString, dan stringToWorld. Penggunaan primitif turunan dari ADT mesin kata dan mesin karakter sebagai pengganti *scanf* merupakan salah satu hal menarik dari algoritma program kami.

6 Data Test

Program BNMO ini terdiri dari beberapa fitur atau *command* yang dapat diberikan oleh pengguna. Oleh karena itu, diperlukan *testing* pada beberapa kondisi untuk memastikan apakah tiap fitur tersebut dapat berjalan dengan tepat dan benar. Berikut adalah *testing* yang kami lakukan terhadap *command - command* yang terdapat pada program BNMO ini, beserta dengan penjelasan mengenai hasil atau *output* yang akan diberikan oleh program pada tiap kondisinya.

6.1 Data Test START

Pada awal menjalankan program, pengguna BNMO diharuskan memilih opsi antara START atau LOAD untuk memulai program. Ketika pengguna BNMO memilih START, maka *game - game* yang dapat dimainkan adalah *game* yang merupakan *default* dari program BNMO. Pada saat *file* konfigurasi berhasil dibuka, maka program akan secara otomatis memberikan *help menu* untuk memberikan bantuan kepada pengguna berupa *list* daftar *command* sebagai bantuan daftar fitur apa saja yang dapat dijalankan pada program tersebut. Apabila pengguna memberikan *command* selain START atau LOAD pada awal program, maka program BNMO akan memberikan pesan *invalid command* dan meminta pengguna untuk kembali memberikan *command* sampai masukan START atau LOAD diberikan.

Gambar 6.1.1 Tampilan ketika berhasil melakukan start

```
*****
      c h o o s e << START >> o r << LOAD [file.txt] >>

Masukkan command: START
File konfigurasi sistem berhasil dibaca. BNMO berhasil dijalankan.

=====
+.*+.*+.*+.*+.*+ H E L P M E N U .+.*+.*+.*+.*+.*+

      daftar dan cara kerja command:
1. LIST GAME      : menampilkan daftar game yang disediakan
2. QUEUE GAME     : mendaftarkan game ke dalam antrian game pribadi
3. PLAY GAME      : memainkan sebuah game
4. SKIP GAME <n>  : melewati game sebanyak yang diinginkan
5. CREATE GAME    : menambahkan game baru pada daftar game
6. DELETE GAME    : menghapus sebuah game dari daftar game
7. QUIT           : keluar dari program game ^^
8. SAVE <filename.txt> : menyimpan state game pemain saat ini
=====

Masukkan command:
```

Gambar 6.1.2 Tampilan ketika pengguna memberikan command selain start atau load pada awal memulai program

```
*****
      c h o o s e << START >> o r << LOAD [file.txt] >>

Masukkan command: LIST GAME
Command tidak dikenali, silahkan masukkan command yang valid.

Masukkan command: CREATE GAME
Command tidak dikenali, silahkan masukkan command yang valid.

Masukkan command: START
File konfigurasi sistem berhasil dibaca. BNMO berhasil dijalankan.

=====
+.*+.*+.*+.*+.*+ H E L P M E N U .+.*+.*+.*+.*+.*+

      daftar dan cara kerja command:
1. LIST GAME      : menampilkan daftar game yang disediakan
2. QUEUE GAME     : mendaftarkan game ke dalam antrian game pribadi
3. PLAY GAME      : memainkan sebuah game
4. SKIP GAME <n>  : melewati game sebanyak yang diinginkan
5. CREATE GAME    : menambahkan game baru pada daftar game
6. DELETE GAME    : menghapus sebuah game dari daftar game
7. QUIT           : keluar dari program game ^^
8. SAVE <filename.txt> : menyimpan state game pemain saat ini
=====

Masukkan command: _
```

6.2 Data Test LOAD

Pada awal menjalankan program, selain memulai BNMO dengan menggunakan *command* START, pengguna juga memiliki opsi untuk memulai BNMO dengan *command* LOAD. *Command* tersebut harus diikuti dengan nama file yang ingin dibuka. Ketika pengguna memilih LOAD, *game* yang dapat dimainkan pada saat itu adalah *game* yang berasal dari suatu *state* yang sudah pernah disimpan oleh pengguna, bukan hanya *game* yang berasal dari *default* program BNMO. Ketika *command* LOAD diberikan pengguna, diikuti nama *file* yang *valid*, maka program akan menampilkan *help menu* untuk memberikan bantuan kepada pengguna berupa *list* daftar *command* sebagai bantuan daftar fitur apa saja yang dapat dijalankan pada

program tersebut. Apabila pengguna memberikan suatu nama file yang tidak *valid* yaitu tidak ada *file* dengan nama tersebut, maka program akan menampilkan pesan kesalahan dan meminta pengguna untuk kembali memasukkan *command* beserta nama *file* yang valid. Kasus ini juga berlaku jika pengguna memasukkan nama *file* namun dengan format selain '.txt' atau '.TXT'. Ketika pengguna hanya memberikan *command* LOAD tanpa memberikan nama *file*, maka program akan menganggapnya sebagai *command* yang *invalid* dan meminta pengguna untuk kembali memberikan *command* yang *valid*.

Gambar 6.2.1 Tampilan ketika berhasil melakukan load

```
*****
  c h o o s e << START >>  o r << LOAD [file.txt] >>

Masukkan command: LOAD save3.txt
LOADNG FILE . . . . . [SUCCEEDED //]

Save file berhasil dibaca. BNMO berhasil dijalankan.

=====
+!*+.+!*+.+!*+.+!*+. H E L P M E N U .+!*+.+!*+.+!*+.
=====

      daftar dan cara kerja command:
1. LIST GAME      : menampilkan daftar game yang disediakan
2. QUEUE GAME     : mendaftarkan game ke dalam antrian game pribadi
3. PLAY GAME      : memainkan sebuah game
4. SKIP GAME <n>  : melewati game sebanyak yang diinginkan
5. CREATE GAME    : menambahkan game baru pada daftar game
6. DELETE GAME    : menghapus sebuah game dari daftar game
7. QUIT           : keluar dari program game ^^
8. SAVE <filename.txt> : menyimpan state game pemain saat ini
=====

Masukkan command:
```

Gambar 6.2.2 Tampilan ketika pengguna memasukkan nama file yang invalid

```
*****
  c h o o s e << START >>  o r << LOAD [file.txt] >>

Masukkan command: LOAD asalnyobahehe.txt
LOADNG FILE . . . . . [FAILED XX]

Load file gagal. File tidak ditemukan.

Masukkan command: _
```

Gambar 6.2.3 Tampilan ketika pengguna memasukkan command load tanpa diikuti nama file

```
*****
  c h o o s e << START >>  o r << LOAD [file.txt] >>

Masukkan command: LOAD
Command tidak dikenali, silahkan masukkan command yang valid.

Masukkan command: _
```

Gambar 6.2.4 Tampilan ketika pengguna memasukkan format file yang invalid

```
*****
  c h o o s e << START >>  o r << LOAD [file.txt] >>

Masukkan command: LOAD aiueo.docx
Format file tidak valid.
```

6.3 Data Test CREATE GAME

Pengguna program dapat memanggil command CREATE GAME untuk membuat game yang baru. Game yang berhasil di buat akan masuk menjadi bagian dari LIST GAME. Apabila pengguna program melakukan input command dengan nama game yang sudah ada pada LIST GAME, maka program akan memberikan perintah bahwa game sudah ada di dalam list.

Gambar 6.3.1 Tampilan ketika berhasil melakukan create game

```
Masukkan command: CREATE GAME
Masukkan nama game yang ingin ditambahkan: MOBILE LEGEND
Game berhasil ditambahkan.
```

Gambar 6.3.2 Tampilan ketika tidak berhasil melakukan create game

```
Masukkan command: CREATE GAME
Masukkan nama game yang ingin ditambahkan: RNG
Game sudah ada di dalam list.
```

6.4 Data Test LIST GAME

Pengguna program dapat memanggil command LIST GAME untuk menampilkan list game yang tersimpan dalam program. Ada lima jenis game yang menjadi file konfigurasi awal sehingga ketika pertama menggunakan program, kelima game ini sudah otomatis ada dalam program. Ketika pengguna melakukan CREATE GAME, maka LIST GAME juga akan bertambah ketika CREATE game berhasil.

Gambar 6.4.1 Tampilan ketika menampilkan LIST GAME awal

```
Masukkan command: LIST GAME
Berikut adalah daftar game yang tersedia
1. RNG
2. DINER DASH
3. DINOSAUR IN EARTH
4. RISEWOMAN
5. EIFFEL TOWER
6. TOWER OF HANOI
Masukkan command: _
```

Gambar 6.4.2 Tampilan ketika ada game yang berhasil diciptakan

```
Masukkan command: CREATE GAME
Masukkan nama game yang ingin ditambahkan: MOBILE LEGEND
Game berhasil ditambahkan.

Masukkan command: LIST GAME
Berikut adalah daftar game yang tersedia
1. RNG
2. DINER DASH
3. DINOSAUR IN EARTH
4. RISEWOMAN
5. EIFFEL TOWER
6. TOWER OF HANOI
7. MOBILE LEGEND
Masukkan command: _
```

6.5 Data Test **DELETE GAME**

Pengguna program dapat menggunakan command **DELETE GAME** untuk menghapus game dari **LIST GAME**. Game yang menjadi file konfigurasi awal yang berjumlah lima game tidak bisa dihapus.

Gambar 6.5.1 Tampilan ketika ingin menghapus game konfigurasi awal

```
Masukkan command: DELETE GAME
Berikut adalah daftar game yang tersedia
1. RNG
2. DINER DASH
3. DINOSAUR IN EARTH
4. RISEWOMAN
5. EIFFEL TOWER
6. TOWER OF HANOI
7. MONOPOLI
Masukkan nomor game yang akan dihapus: 2

Game gagal dihapus.
```

Pengguna program juga tidak bisa menghapus game yang ada di **QUEUE GAME** walaupun game tersebut bukan bagian file konfigurasi awal game.

*Gambar 6.5.2 Tampilan ketika ingin menghapus game yang ada di **QUEUE GAME***

```
Berikut adalah daftar game yang tersedia
1. RNG
2. DINER DASH
3. DINOSAUR IN EARTH
4. RISEWOMAN
5. EIFFEL TOWER
6. TOWER OF HANOI
7. MONOPOLI

Nomor Game yang mau ditambahkan ke antrian: 7
Game berhasil ditambahkan ke dalam daftar antrian.

Masukkan command: DELETE GAME
Berikut adalah daftar game yang tersedia
1. RNG
2. DINER DASH
3. DINOSAUR IN EARTH
4. RISEWOMAN
5. EIFFEL TOWER
6. TOWER OF HANOI
7. MONOPOLI
Masukkan nomor game yang akan dihapus: 7

Game gagal dihapus.
```

Pengguna program dapat menghapus game yang tidak ada pada queue game dan tidak ada pada file konfigurasi awal.

Gambar 6.5.3 Tampilan DELETE GAME yang berhasil

```
Masukkan command: DELETE GAME
Berikut adalah daftar game yang tersedia
1. RNG
2. DINER DASH
3. DINOSAUR IN EARTH
4. RISEWOMAN
5. EIFFEL TOWER
6. TOWER OF HANOI
7. MONOPOLI
Masukkan nomor game yang akan dihapus: 7
Game berhasil dihapus.
```

6.6 Data Test QUEUE GAME

Pengguna program dapat menggunakan QUEUE GAME untuk memasukkan game ke dalam antrian. Antrian ini berguna ketika ingin melakukan PLAY GAME sehingga game yang akan dimainkan adalah sesuai dengan antrian. Pengguna juga bisa memasukkan game yang sama kedalam antrian ini.

Gambar 6.6.1 Tampilan ketika berhasil memasukkan game ke dalam antrian

```
Masukkan command: QUEUE GAME
Berikut adalah daftar antrian game-mu
Antrian kosong

Berikut adalah daftar game yang tersedia
1. RNG
2. DINER DASH
3. DINOSAUR IN EARTH
4. RISEWOMAN
5. EIFFEL TOWER
6. TOWER OF HANOI

Nomor Game yang mau ditambahkan ke antrian: 2
Game berhasil ditambahkan ke dalam daftar antrian.

Masukkan command: QUEUE GAME
Berikut adalah daftar antrian game-mu
1. DINER DASH
```

6.7 Data Test SAVE

Pengguna program dapat menggunakan command SAVE untuk menyimpan file sesuai dengan apa yang dilakukan oleh pengguna program untuk dilanjutkan kembali ketika ingin menggunakan program kembali. SAVE dapat dilakukan dengan dua cara yaitu langsung menginput SAVE <filename> pada command atau SAVE saat melakukan input QUIT pada command program.

Gambar 6.7.1 Tampilan ketika berhasil melakukan save file

```
Masukkan command: SAVE testfile.txt
Save file berhasil disimpan

Masukkan command: QUIT
Apakah anda ingin melakukan SAVE sebelum keluar dari program? (Y/N): Y
Masukkan nama file: testfile2.txt
Save file berhasil disimpan

Anda keluar dari game BNMO
Bye bye ...
```


6.8 Data Test PLAY GAME

Pengguna program dapat menggunakan command PLAY GAME untuk memainkan game yang tersimpan dalam QUEUE GAME. Urutan dalam QUEUE GAME akan menentukan urutan game yang akan di mainkan dengan command PLAY GAME. Ketika user memilih RNG pertama kali dalam QUEUE GAME maka ketika user menjalankan command PLAY GAME, akan masuk ke RNG langsung.

Gambar 6.8.1 Tampilan ketika berhasil PLAY GAME game yang ada di QUEUE GAME

```
Masukkan command: PLAY GAME
Berikut adalah daftar antrian game-mu
1. RNG

Loading RNG . . .

RNG telah dimulai. Uji keberuntungan Anda dengan menebak angka X.
Tebakan: Lebih besar
Tebakan:
```

6.9 Data Test Command Lain

Pengguna program dapat memasukkan command apapun tetapi jika tidak sesuai dengan daftar command yang ada di HELP maka program akan memberikan pesan “Command tidak dikenali, silahkan masukkan command yang valid”.

Gambar 6.9.1 Tampilan ketika command yang dimasukan salah

```
Masukkan command: COMMAND ANEH
Command tidak dikenali, silahkan masukkan command yang valid.
```

6.10 Data Test Diner Dash

Pengguna program dapat memainkan game Diner Dash yang terdapat dalam program BNMO. Diner Dash merupakan sebuah permainan mengantar makanan namun secara terurut berdasarkan prioritas. Diner Dash memiliki spesifikasi game tersendiri. Spesifikasi tersebut meliputi terdapat 3 command yang dapat dilakukan dalam game, yaitu COOK, SERVE, dan SKIP. Dimana COOK merupakan command yang bertujuan untuk memasak makanan, SERVE merupakan command yang bertujuan untuk menyajikan makanan, dan SKIP yang merupakan command yang bertujuan untuk menyelesaikan 1 putaran tanpa melakukan apapun tetapi (durasi berkurang 1 serta ketahanan bertambah 1).

Saat memulai permainan, user akan ditampilkan dengan tiga buat pesanan yaitu M0, M1, M2 yang memiliki durasi memasak, ketahanan makanan, dan harga makanan. User dapat memasak pesanan apapun yang ada di tampilan pesanan dengan command “COOK Mx”. Setiap putaran pesanan yang masuk ke dalam tampilan masak akan berkurang waktu masaknya. Ketika sudah selesai waktu masaknya akan dimasukan ke dalam tampilan serve. User dapat “SERVE Mx” untuk menyajikan makanan dan menambah skor yang berupa saldo. SERVE hanya dapat dilakukan jika pesanan terdahulu sudah di serve.

Gambar 6.10.1 Tampilan DINER DASH ketika dimainkan dengan command PLAY GAME

```
Masukkan command: PLAY GAME
Berikut adalah daftar antrian game-mu
1. DINER DASH

Loading DINER DASH . . .

Selamat Datang di Diner Dash!

SALDO: 0

Daftar Pesanan
Makanan | Durasi memasak | Ketahanan | Harga
-----
M0      |      5         |      2    | 39209
M1      |      5         |      3    | 31317
M2      |      5         |      5    | 33198

Daftar Makanan yang sedang dimasak
Makanan | Sisa durasi memasak
-----
|

Daftar Makanan yang dapat disajikan
Makanan | Sisa ketahanan makanan
-----
|

COMMAND : _
```

Gambar 6.10.2 Tampilan DINER DASH ketika dimasukan command COOK M0

```
COMMAND : COOK M0

Berhasil memasak M0
=====

SALDO: 0

Daftar Pesanan
Makanan | Durasi memasak | Ketahanan | Harga
-----
M0      |      2         |      3    | 16334
M1      |      1         |      5    | 25724
M2      |      4         |      4    | 36962
M3      |      5         |      1    | 38145

Daftar Makanan yang sedang dimasak
Makanan | Sisa durasi memasak
-----
M0      |      2

Daftar Makanan yang dapat disajikan
Makanan | Sisa ketahanan makanan
-----
|

COMMAND : _
```

Gambar 6.10.3 Tampilan DINER DASH ketika dimasukan command SKIP

```

COMMAND : SKIP
SKIP 1 PUTARAN
=====

SALDO: 0

Daftar Pesanan
Makanan | Durasi memasak | Ketahanan | Harga
-----
M0      |      2         |      3    | 16334
M1      |      1         |      5    | 25724
M2      |      4         |      4    | 36962
M3      |      5         |      1    | 38145
M4      |      2         |      3    | 19961

Daftar Makanan yang sedang dimasak
Makanan | Sisa durasi memasak
-----
M0      |      1

Daftar Makanan yang dapat disajikan
Makanan | Sisa ketahanan makanan
-----
|

COMMAND : _

```

Gambar 6.10.4 Tampilan DINER DASH ketika dimasukan command COOK kembali

```

COMMAND : COOK M1
Makanan M0 telah selesai dimasak

Berhasil memasak M1
=====

SALDO: 0

Daftar Pesanan
Makanan | Durasi memasak | Ketahanan | Harga
-----
M0      |      2         |      3    | 16334
M1      |      1         |      5    | 25724
M2      |      4         |      4    | 36962
M3      |      5         |      1    | 38145
M4      |      2         |      3    | 19961
M5      |      2         |      1    | 21942

Daftar Makanan yang sedang dimasak
Makanan | Sisa durasi memasak
-----
M1      |      1

Daftar Makanan yang dapat disajikan
Makanan | Sisa ketahanan makanan
-----
M0      |      3

COMMAND :

```

Gambar 6.10.5 Tampilan DINER DASH ketika dimasukan command SERVE

```

COMMAND : SERVE M0

Makanan M1 telah selesai dimasak
=====

SALDO: 16334

Daftar Pesanan
Makanan | Durasi memasak | Ketahanan | Harga
-----
M1      | 1              | 5         | 25724
M2      | 4              | 4         | 36962
M3      | 5              | 1         | 38145
M4      | 2              | 3         | 19961
M5      | 2              | 1         | 21942
M6      | 3              | 2         | 42391

Daftar Makanan yang sedang dimasak
Makanan | Sisa durasi memasak
-----
      |

Daftar Makanan yang dapat disajikan
Makanan | Sisa ketahanan makanan
-----
M1      | 5

COMMAND : 

```

Gambar 6.10.6 Tampilan DINER DASH ketika daftar pesanan sudah maksimal

```

Daftar Pesanan
Makanan | Durasi memasak | Ketahanan | Harga
-----
M2      | 4              | 4         | 36962
M3      | 5              | 1         | 38145
M4      | 2              | 3         | 19961
M5      | 2              | 1         | 21942
M6      | 3              | 2         | 42391
M7      | 5              | 3         | 10153
M8      | 3              | 3         | 27421

Daftar Makanan yang sedang dimasak
Makanan | Sisa durasi memasak
-----
M2      | 4

Daftar Makanan yang dapat disajikan
Makanan | Sisa ketahanan makanan
-----
      |

COMMAND : COOK M3

Berhasil memasak M3
=====
Skor Anda : 42058
=====
- DINER DASH OVER -

```

6.11 Data Test SKIP GAME

Pengguna program dapat melewati sejumlah permainan pada daftar antrian game yang tidak ingin dimainkan menggunakan command SKIP GAME <jumlah permainan yang ingin dilewati>. Setelah permainan dilewati, game yang berada pada urutan nomor 1 pada antrian game akan langsung dimainkan. Input jumlah game yang ingin dilewati dapat melebihi jumlah permainan pada antrian game, tetapi command ini akan mengembalikan pesan error bila pemain memanggil command SKIP GAME pada kondisi daftar antrian game kosong.

Gambar 6.11.1 Tampilan daftar antrian game yang dimiliki pengguna program

```
Masukkan command: QUEUE GAME
Berikut adalah daftar antrian game-mu
1. RNG
2. DINER DASH
3. DINOSAUR IN EARTH
4. RISEWOMAN
5. EIFFEL TOWER
```

Gambar 6.11.2 Tampilan daftar antrian game yang dimiliki setelah melewati 2 dari 5 permainan dalam antrian game melalui command SKIP GAME

```
Masukkan command: SKIP GAME 2
Berikut adalah daftar antrian game-mu
1. DINOSAUR IN EARTH
2. RISEWOMAN
3. EIFFEL TOWER

Game DINOSAUR IN EARTH masih dalam maintenance, belum dapat dimainkan. Silahkan pilih game lain.
```

Gambar 6.11.3 Tampilan daftar antrian game yang dimiliki setelah melewati 5 dari 2 permainan dalam antrian game melalui command SKIP GAME

```
Masukkan command: SKIP GAME 5
Tidak ada permainan lagi di dalam daftar game-mu
```

Gambar 6.11.4 Tampilan ketika pengguna program memanggil command SKIP GAME pada kondisi antrian game kosong

```
Masukkan command: SKIP GAME 1
Tidak ada permainan di dalam daftar game-mu

Masukkan command: _
```

6.12 Data Test QUIT

Bila pengguna program memutuskan untuk berhenti bermain, pengguna program dapat memanggil command QUIT. Namun, sebelum keluar dari game BNMO, program akan menanyakan apakah pengguna program ingin melakukan penyimpanan daftar game yang dimiliki pada saat itu. Apabila pengguna program memilih untuk melakukan save sebelum keluar dari program, maka perubahan pada daftar game yang dimiliki akan disimpan pada suatu save file yang diinginkan pengguna. Apabila pengguna program memilih untuk tidak melakukan save sebelum keluar dari program, maka perubahan pada daftar game yang dimiliki tidak akan tersimpan.

Gambar 6.12.1 Tampilan ketika pengguna program mengubah kondisi daftar game dan menyimpan kondisi daftar game pada suatu save file setelah memanggil command QUIT

```
Masukkan command: LIST GAME
Berikut adalah daftar game yang tersedia
1. RNG
2. DINER DASH
3. DINOSAUR IN EARTH
4. RISEWOMAN
5. EIFFEL TOWER
6. TOWER OF HANOI

Masukkan command: CREATE GAME
Masukkan nama game yang ingin ditambahkan: JIGSAW
Game berhasil ditambahkan.

Masukkan command: QUIT
Apakah anda ingin melakukan SAVE sebelum keluar dari program? (Y/N): Y
Masukkan nama file: PERCOBAAN1.txt
Save file berhasil disimpan

Anda keluar dari game BNMO
Bye bye ...
```

Gambar 6.12.2 Tampilan kondisi daftar game dari save file setelah melakukan SAVE melalui penggunaan command QUIT

```
Masukkan command: LOAD PERCOBAAN1.txt
LOADNG FILE . . . . . [SUCCEEDED //]

Save file berhasil dibaca. BNMO berhasil dijalankan.

=====
+.*+.*+.*+.*+.*+ H E L P M E N U .+.*+.*+.*+.*+.*+
=====

      daftar dan cara kerja command:
1. LIST GAME      : menampilkan daftar game yang disediakan
2. QUEUE GAME     : mendaftarkan game ke dalam antrian game pribadi
3. PLAY GAME      : memainkan sebuah game
4. SKIP GAME <n>  : melewati game sebanyak yang diinginkan
5. CREATE GAME    : menambahkan game baru pada daftar game
6. DELETE GAME    : menghapus sebuah game dari daftar game
7. QUIT           : keluar dari program game ^^
8. SAVE <filename.txt> : menyimpan state game pemain saat ini
=====

Masukkan command: LIST GAME
Berikut adalah daftar game yang tersedia
1. RNG
2. DINER DASH
3. DINOSAUR IN EARTH
4. RISEWOMAN
5. EIFFEL TOWER
6. TOWER OF HANOI
7. JIGSAW

Masukkan command:
```

Gambar 6.12.3 Tampilan ketika pengguna program melakukan perubahan pada kondisi daftar game yang dimiliki dan tidak melakukan SAVE setelah memanggil command QUIT

```
Masukkan command: LIST GAME
Berikut adalah daftar game yang tersedia
1. RNG
2. DINER DASH
3. DINOSAUR IN EARTH
4. RISEWOMAN
5. EIFFEL TOWER
6. TOWER OF HANOI
7. JIGSAW

Masukkan command: CREATE GAME
Masukkan nama game yang ingin ditambahkan: SONIC
Game berhasil ditambahkan.

Masukkan command: QUIT
Apakah anda ingin melakukan SAVE sebelum keluar dari program? (Y/N): N

Anda keluar dari game BNMO
Bye bye ...
```

Gambar 6.12.4 Tampilan daftar game yang dimiliki pengguna bila tidak melakukan SAVE melalui penggunaan command QUIT

```
Masukkan command: LOAD PERCOBAAN1.txt
LOADNG FILE . . . . . [SUCCEEDED //]

Save file berhasil dibaca. BNMO berhasil dijalankan.

=====
+:+.+:*+.*+.*+.*+. H E L P M E N U .+:+.*+.*+.*+.*+.
=====

      daftar dan cara kerja command:
1. LIST GAME      : menampilkan daftar game yang disediakan
2. QUEUE GAME     : mendaftarkan game ke dalam antrian game pribadi
3. PLAY GAME      : memainkan sebuah game
4. SKIP GAME <n>  : melewati game sebanyak yang diinginkan
5. CREATE GAME    : menambahkan game baru pada daftar game
6. DELETE GAME    : menghapus sebuah game dari daftar game
7. QUIT           : keluar dari program game ^^
8. SAVE <filename.txt> : menyimpan state game pemain saat ini
=====

Masukkan command: LIST GAME
Berikut adalah daftar game yang tersedia
1. RNG
2. DINER DASH
3. DINOSAUR IN EARTH
4. RISEWOMAN
5. EIFFEL TOWER
6. TOWER OF HANOI
7. JIGSAW

Masukkan command:
```

6.13 Data Test Help

Pengguna program dapat menggunakan command HELP untuk menunjukkan daftar command yang dapat digunakan pada BNMO beserta fungsi nya. Ketika pengguna program baru saja melakukan START atau LOAD, command HELP akan dipanggil secara otomatis.

Gambar 6.13.1 Tampilan pemanggilan otomatis command HELP setelah pengguna program melakukan START

```
*****
c h o o s e << START >> o r << LOAD [file.txt] >>

Masukkan command: START
File konfigurasi sistem berhasil dibaca. BNMO berhasil dijalankan.

=====
+:*+.*+.*+.*+.*+. H E L P M E N U .+*+.*+.*+.*+.*+.
=====
daftar dan cara kerja command:
1. LIST GAME      : menampilkan daftar game yang disediakan
2. QUEUE GAME    : mendaftarkan game ke dalam antrian game pribadi
3. PLAY GAME     : memainkan sebuah game
4. SKIP GAME <n> : melewati game sebanyak yang diinginkan
5. CREATE GAME   : menambahkan game baru pada daftar game
6. DELETE GAME   : menghapus sebuah game dari daftar game
7. QUIT          : keluar dari program game ^^
8. SAVE <filename.txt> : menyimpan state game pemain saat ini
=====
Masukkan command:
```

Gambar 6.13.2 Tampilan pemanggilan command HELP

```
Masukkan command: HELP
=====
+:*+.*+.*+.*+.*+. H E L P M E N U .+*+.*+.*+.*+.*+.
=====
daftar dan cara kerja command:
1. LIST GAME      : menampilkan daftar game yang disediakan
2. QUEUE GAME    : mendaftarkan game ke dalam antrian game pribadi
3. PLAY GAME     : memainkan sebuah game
4. SKIP GAME <n> : melewati game sebanyak yang diinginkan
5. CREATE GAME   : menambahkan game baru pada daftar game
6. DELETE GAME   : menghapus sebuah game dari daftar game
7. QUIT          : keluar dari program game ^^
8. SAVE <filename.txt> : menyimpan state game pemain saat ini
=====
Masukkan command: _
```

6.14 Data Test RNG

RNG merupakan salah satu game yang dapat dimainkan pada BNMO. Bila game ini dimainkan, pengguna akan diberikan 8 kesempatan untuk menebak suatu angka random X pada rentang 1 sampai 500. Game kemudian akan memberikan petunjuk berupa apakah angka random X lebih besar atau lebih kecil dari tebakan pemain. Bila pemain gagal menebak sebanyak 8 kali, maka program akan memberi tahu nilai dari angka random X, berapa skor dari pengguna, dan game akan berakhir.

Gambar 16.14.1 Tampilan ketika pengguna program gagal menebak angka random X pada permainan RNG sebanyak 8 kali

```
Masukkan command: PLAY GAME
Berikut adalah daftar antrian game-mu
1. RNG

Loading RNG . . .

RNG telah dimulai. Uji keberuntungan Anda dengan menebak angka X.
Tebakan: 1
Lebih besar
Tebakan: 2
Lebih besar
Tebakan: 3
Lebih besar
Tebakan: 4
Lebih besar
Tebakan: 5
Lebih besar
Tebakan: 6
Lebih besar
Tebakan: 7
Lebih besar
Tebakan: 8
Lebih besar
Kesempatan Anda habis. X yang tepat adalah 239
Skor Anda: 0
Masukkan command:
```

Gambar 16.14.2 Tampilan ketika pengguna program berhasil menebak angka X pada kesempatan ke 8

```
Masukkan command: PLAY GAME
Berikut adalah daftar antrian game-mu
1. RNG

Loading RNG . . .

RNG telah dimulai. Uji keberuntungan Anda dengan menebak angka X.
Tebakan: 250
Lebih kecil
Tebakan: 125
Lebih kecil
Tebakan: 50
Lebih kecil
Tebakan: 30
Lebih kecil
Tebakan: 20
Lebih besar
Tebakan: 25
Lebih besar
Tebakan: 27
Lebih kecil
Tebakan: 26
Ya, X adalah 26
Skor Anda: 20
Masukkan command: _
```

6.15 Data Test **BONUS**

Kami membuat sebuah game yang bernama "Tower of Hanoi" yaitu salah satu game yang dapat dimainkan di dalam program ini. Game ini menjadi suatu spesifikasi bonus yang kami buat. Game ini dibuat dengan menerapkan ADT STACK. Pada awal permainan, pemain akan diminta untuk memasukkan jumlah cakram yang diinginkan, lalu game akan langsung dimulai. Pada saat memainkan, pengguna dapat melihat jumlah langkah yang telah dilakukan. Game akan berakhir jika tumpukan sudah tertumpuk secara terurut. Pada akhir permainan, game akan menampilkan *score* yang diperoleh oleh pemain.

Gambar 6.15.1 Tampilan awal game Tower of Hanoi

```
Masukkan command: PLAY GAME
Berikut adalah daftar antrian game-mu
1. TOWER OF HANOI

Loading TOWER OF HANOI . . .

=====
- TOWER OF HANOI -
=====

Masukkan jumlah cakram: 3

-----

Tower 1: [3, 2, 1]
Tower 2: []
Tower 3: []

      =
      ===
      =====
1      2      3
```

Gambar 6.15.2 Tampilan akhir game "Tower of Hanoi"

```
Memindahkan cakram dari tower (1 - 3): 1
Ke tower (1 - 3): 3

-----

Tower 1: []
Tower 2: []
Tower 3: [3, 2, 1]

      =
      ===
      =====
1      2      3

Jumlah langkah: 7

Selamat, Anda berhasil menyelesaikan Tower of Hanoi!
Skor Anda: 100
```

7 Test Script

No .	Fitur yang Dites	Tujuan Testing	Langkah-Langkah Testing	Input Data Test	Hasil yang Diharapkan	Hasil yang Keluar
1	Start	Memeriksa apakah <i>file</i> konfigurasi berhasil dibaca dan menampilkan tampilan ketika <i>file</i> konfigurasi berhasil dibuka	memberikan <i>command</i> START	START	Gambar 6.1.1 Gambar 6.1.2	Sesuai dengan hasil yang Diharapkan
2	Load	Memeriksa apakah <i>file</i> berhasil dibaca dan menampilkan tampilan ketika <i>file</i> berhasil dibuka	memberikan <i>command</i> LOAD diikuti dengan nama <i>file</i> yang <i>valid</i>	LOAD	Gambar 6.2.1 Gambar 6.2.2 Gambar 6.2.3 Gambar 6.2.4 Gambar 6.1.2	Sesuai dengan hasil yang Diharapkan
3	Create Game	Memeriksa apakah game bisa ditambahkan ke dalam list game	Mengetik CREATE GAME pada command setelah START atau LOAD lalu input nama Game yang ingin ditambahkan ke list	CREATE GAME	Gambar 6.3.1	Sesuai dengan hasil yang Diharapkan
4	List Game	Menampilkan game apa saja yang ada dalam program	Mengetik LIST GAME pada command setelah START atau LOAD	LIST GAME	Gambar 6.4.1 Gambar 6.4.2	Sesuai dengan hasil yang Diharapkan
5	Delete Game	Menghapus game dari list game yang ada	Mengetik DELETE GAME pada command setelah START atau LOAD lalu input nomor game sesuai list game untuk game yang ingin dihapus	DELETE GAME	Gambar 6.5.3	Sesuai dengan hasil yang Diharapkan
6	Queue Game	Memasukkan game ke dalam antrian	Mengetik QUEUE GAME pada command setelah START atau LOAD lalu input angka game yang ingin dimasukkan ke antrian	QUEUE GAME	Gambar 6.6.1	Sesuai dengan hasil yang Diharapkan
7	Save	Melakukan save file	Mengetik SAVE<filename> pada command setelah START atau LOAD	SAVE <filename>	Gambar 6.7.1	Sesuai dengan hasil yang Diharapkan
8	Play Game	Memainkan game yang ada telah masuk antrian	Mengetik PLAY GAME pada command setelah	PLAY GAME	Gambar 6.8.1	Sesuai dengan hasil yang Diharapkan

		dalam QUEUE GAME BNMO	menambahkan game dalam QUEUE GAME			
9	Command Lain	Melewatkan command yang tidak sesuai dengan daftar command yang ada dalam BNMO	Mengetik apapun yang bukan merupakan command dalam program BNMO	<command apapun selain command dalam HELP BNMO>	Gambar 6.9.1	Sesuai dengan hasil yang Diharapkan
10	Diner Dash	Menjalankan game Diner Dash dimana pengguna akan diberikan daftar pesanan berisi 3 menu pertama dan dapat di COOK dan SERVE. Serta memiliki skor akhir berupa saldo user.	Mengetik PLAY GAME setelah memasukkan game DINER DASH ke dalam QUEUE GAME dan game DINER DASH berada di urutan kedua QUEUE GAME	QUEUE GAME 2 dan PLAY GAME	Gambar 6.10.1 Gambar 6.10.2 Gambar 6.10.3 Gambar 6.10.4 Gambar 6.10.5 Gambar 6.10.6	Sesuai dengan hasil yang Diharapkan
11	Skip Game	Melewatkan sejumlah permainan pada daftar antrian game	Mengetik SKIP GAME <N> pada command, dengan N sebagai jumlah game yang ingin dilewati, setelah menambahkan game dalam QUEUE GAME	SKIP GAME <jumlah permainan yang ingin dilewati>	Gambar 6.11.2 Gambar 6.11.3 Gambar 6.11.4	Sesuai dengan hasil yang Diharapkan
12	Quit	Memberikan pilihan untuk melakukan save file dan keluar dari program BNMO	Mengetik QUIT pada command setelah START atau LOAD	QUIT	Gambar 6.12.1 Gambar 6.12.2 Gambar 6.12.3 Gambar 6.12.4	Sesuai dengan hasil yang Diharapkan
13	Help	Menampilkan daftar command yang dapat dijalankan pada program BNMO beserta fungsinya	Mengetik HELP pada command setelah START atau LOAD	HELP	Gambar 6.13.1 Gambar 6.13.2	Sesuai dengan hasil yang Diharapkan
14	RNG	Memberikan pengguna 8 kesempatan untuk menebak suatu angka random yang berbeda di setiap giliran bermain serta memberikan skor yang dimiliki oleh pengguna	Mengetik PLAY GAME setelah memasukkan game RNG ke dalam QUEUE GAME dan game RNG berada di urutan pertama QUEUE GAME	QUEUE GAME 1 dan PLAY GAME	Gambar 6.14.1 Gambar 6.14.2	Sesuai dengan hasil yang Diharapkan

15	Bonus	Menjalankan game bonus yaitu "Tower of Hanoi" yaitu menggunakan ADT STACK	Mengetik PLAY GAME setelah memasukkan game TOWER OF HANOI ke dalam QUEUE GAME dan game RNG berada di urutan pertama QUEUE GAME	QUEUE GAME 6 dan PLAY GAME	Gambar 6.15.1 Gambar 6.15.2	Sesuai dengan hasil yang Diharapkan
----	-------	---	--	----------------------------	--------------------------------	-------------------------------------

8 Pembagian Kerja dalam Kelompok

Tabel 8.1.1 Pembagian kerja dalam Kelompok

No.	Nama Anggota - NIM	Deskripsi Kontribusi
1.	Michael Sihotang – 18221054	ADT Mesin Kata dan Mesin Karakter, ADT Queue, ADT Stack, Create Game, Play Game, Driver ADT, Laporan
2.	Muhammad Dastin Fauzi – 18221062	Queue Game, Quit, Diner Dash, ADT Map, Driver ADT, Laporan
3.	Imanuel Raditya – 18221112	Main Program, ADT Array Dinamis, ADT Mesin Karakter dan Mesin Kata, ADT Queue , Command Load, List Game, Command Lain, Bonus (Tower of Hanoi)
4.	Miralistya Cahya Fatimah – 18221116	ADT Mesin Karakter dan Mesin Kata untuk pembacaan file, Start Menu, Command Help, Skip Game, Driver ADT, Laporan
5.	Seren Elizabeth Siahaan – 18221160	Game RNG, Command Save File, Delete Game, Laporan

9 Lampiran

9.1 Deskripsi Tugas Besar 1

BNMO (dibaca: Binomo) adalah sebuah robot video game console yang dimiliki oleh Indra dan Doni. Dua bulan yang lalu, ia mengalami kerusakan dan telah berhasil diperbaiki. Sayangnya, setelah diperbaiki ia justru mendapatkan lebih banyak bug dalam sistemnya. Oleh karena itu, Indra dan Doni mencari programmer lain yang lebih handal untuk ulang memprogram robot video game console kesayangannya.

Buatlah sebuah permainan berbasis CLI (command-line interface). Sistem 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. Library yang boleh digunakan hanya `stdio.h`, `stdlib.h`, `time.h` dan `math.h`

System Mechanics

1. About the System

BNMO merupakan suatu robot game console yang dapat menjalankan permainan. BNMO memiliki beberapa fitur utama, yaitu:

1. Memainkan game
2. Menambahkan game
3. Menghapus game
4. Mengurutkan game yang akan dimainkan

2. Main Menu

Ketika program pertama kali dijalankan, BNMO akan memperlihatkan main menu yang berisi welcome page dan beberapa menu pilihan yaitu START dan LOAD. Setelah itu, main menu akan menerima input commands yang akan dijelaskan pada bagian berikutnya.

3. Command

Pada setiap giliran, pemain dapat memasukkan command-command berikut:

- START : Salah satu command yang dimasukkan pertama kali oleh pemain ke BNMO. Setelah menekan Enter, dibaca file konfigurasi default yang berisi list game yang dapat dimainkan.
- LOAD <filename> : Salah satu command yang dimasukkan pertama kali oleh pemain ke BNMO. Memiliki satu argumen yaitu filename yang merepresentasikan suatu *save file* yang ingin dibuka. Setelah menekan Enter, akan dibaca save file <filename> yang berisi list game yang dapat dimainkan, histori dan scoreboard game.
- SAVE <filename> : command yang digunakan untuk menyimpan state game pemain saat ini ke dalam suatu file. Command SAVE memiliki satu argumen yang merepresentasikan nama file yang akan disimpan pada disk.
- CREATE GAME : command yang digunakan untuk menambahkan game baru pada daftar game.
- LIST GAME : command yang digunakan untuk menampilkan daftar game yang disediakan oleh sistem.
- DELETE GAME : command yang digunakan untuk menghapus sebuah game dari daftar game.
- QUEUE GAME : command yang digunakan untuk mendaftarkan permainan kedalam list.
- PLAY GAME : command yang digunakan untuk memainkan sebuah permainan.

- SKIP GAME <n> : command yang digunakan untuk melewati permainan sebanyak n.
- QUIT : Keluar dari program.
- HELP : Bantuan command-command yang disebutkan di atas. Tampilan dan kata-kata dibebaskan.
- COMMAND LAIN : Command-command lain selain yang disebutkan diatas tidak valid.

Spesifikasi Game

1. RNG

BNMO tidak selalu menikmati *game* yang sudah pasti *outcome*-nya. Karena itu, ia suka dengan *game* yang melibatkan RNG (*Random number generator*).

2. Diner Dash

Indra dan Doni juga suka permainan yang menegangkan. Oleh karena itu, ia ingin ada sebuah game Diner Dash dalam BNMO. Secara singkat, Diner Dash merupakan permainan mengantar makanan namun terurut berdasarkan prioritasnya.

3. Tower of Hanoi

Permainan matematika atau teka-teki. Ini terdiri dari n batang, dan sejumlah piringan dengan ukuran yang berbeda yang dapat berpindah ke batang lainnya. Teka-teki dimulai dengan tumpukan piringan dalam urutan ukuran pada satu batang, yang terkecil di bagian atas, sehingga membuat bentuk kerucut.







9.2 Notulen Rapat

**Form Asistensi Tugas Besar
IF2110/Algoritma dan Struktur Data
Sem. 1 2022/2023**







No. Kelompok/Kelas	: 8 / K2
Nama Kelompok	:
Anggota Kelompok (Nama/NIM)	: 1. Michael Sihotang (18221054) 2. Muhammad Dastin Fauzi (18221062) 3. Imanuel Raditya (18221112) 4. Miralistya Cahya Fatimah (18221116) 5. Seren Elizabeth Siahaan (18221160)

Asisten Pembimbing : Graciella Valeska Liander (18219075)

Asistensi I

Tanggal : 3 November 2022	Catatan Asistensi: <ul style="list-style-type: none"> - Pada command skip, apabila input n lebih besar dari jumlah game pada daftar antrian, maka semua game pada antrian akan terhapus, bukan hanya membatalkan <i>command SKIP</i>. - Keluaran dari <i>SKIP</i> diperbolehkan untuk mengeluarkan daftar game di antrian, karena ketika memanggil <i>command PLAY GAME</i> akan ditampilkan terlebih dahulu daftar antrian game.
Tempat : Zoom meetings	
Kehadiran Anggota Kelompok: <div style="text-align: center;"> <p>1 18221054</p>  <p>2 18221062</p>  <p>3 18221112</p>  <p>4 18221116</p>  <p>5 18221160</p>  </div>	
	Tanda Tangan Asisten: <div style="text-align: center;">  <p>graciellari</p> </div>

Asistensi II

Tanggal : 10 November 2022	
Tempat : Zoom meetings	
Kehadiran Anggota Kelompok: <div><div>1</div><div>18221054</div><div></div><div>2</div><div>18221062</div><div></div><div>3</div><div>18221112</div><div></div><div>4</div><div>18221116</div><div></div><div>5</div><div>18221160</div><div></div></div>	Catatan Asistensi: <ul style="list-style-type: none">- scoreboard dihilangkan, namun pada PLAY GAME, jika yang dimainkan adalah game hasil CREATE GAME oleh pengguna, maka tetap harus mengeluarkan score secara random dan menuliskan bahwa game over.- Ketika QUIT queue game akan terhapus, hal tersebut boleh dilakukan secara tersirat yaitu dengan pembuatan CreateEmptyQueue, tanpa harus secara eksplisit yaitu dengan dequeue satu satu.- Ketika code di run di vscode terkadang mengalami error glitch sesekali, seperti ketika melakukan createGame beberapa kali lalu ketika memanggil ListGame, tiba tiba exit. Namun ketika mengulangi langkah yang sama persis kembali, tidak terjadi error serupa. Ketika dicoba untuk di run pada cmd juga tidak mengalami error. Hal tersebut dapat diabaikan bila glitch hanya terjadi sesekali karena mungkin error dari vscode nya.- Driver pada ADT boleh hanya perlu mengaplikasikan fungsi dan prosedur original dari ADT tersebut.- Untuk Data Test pada laporan, berisikan SS dari program (hasil share screen sudah benar).- Pada Test Script. kolom 'Hasil yang Keluar' dapat hanya mengacu pada gambar atau dituliskan bahwa sesuai.- Pada Test Script, kolom 'Hasil Yang Diharapkan' dapat mengacu pada gambar Data Test.
	Tanda Tangan Asisten: <div> graciellari</div>

9.3 Log Activity Anggota Kelompok

NO	Tanggal	NIM	Nama	Aktivitas
1	30/10/2022	18221054	Michael Sihotang	Membuat repository github
2	31/10/2022	18221054	Michael Sihotang	Push ADT Array Dinamis, ADT Mesin Karakter, ADT Mesin Kata dan ADT Queue
3	01/11/2022	18221116	Miralistya Cahya F	Push command HELP dan merapikan folder
4	02/11/2022	18221112	Imanuel Raditya	Memperbaiki ADT Array Dinamis, ADT Mesin Karakter, ADT Mesin Kata, dan ADT Queue
5	02/11/2022	18221160	Seren Elizabeth S	Push game RNG
6	02/11/2022	18221116	Miralistya Cahya F	Push command START
7	02/11/2022	18221053	Michael Sihotang	Push CREATE GAME dan PLAY GAME
8	03/11/2022	18221116	Miralistya Cahya F	Push command SKIP GAME
9	03/11/2022	18221062	Muhammad Dastin F	Push command QUIT
10	03/11/2022	18221112	Imanuel Raditya	Edit Test dan Revisi CREATE GAME
11	04/11/2022	18221112	Imanuel Raditya	Revisi ADT Mesin Karakter dan Mesin Kata
12	04/11/2022	18221112	Imanuel Raditya	Push command LOAD
13	05/11/2022	18221112	Imanuel Raditya	Memodifikasi START GAME
14	06/11/2022	18221062	Muhammad Dastin F	Push QUEUE GAME
15	07/11/2022	18221062	Muhammad Dastin F	Push DINER DASH
16	07/11/2022	18221062	Muhammad Dastin F	Push ADT MAP dan ADT QueueDinerDash
17	07/11/2022	18221112	Imanuel Raditya	Membuat fungsi main.c
18	07/11/2022	18221054	Michael Sihotang	Memulai mengisi laporan

19	08/11/2022	18221160	Seren Elizabeth S	Push command DELETE GAME dan SAVE
20	08/11/2022	18221116	Miralistya Cahya F	Mengisi laporan bagian ringkasan dan Struktur Data
21	08/11/2022	18221054	Michael Sihotang	Mengisi laporan bagian penjelasan tambahan spesifikasi dan struktur data
22	09/11/2022	18221112	Imanuel Raditya	Revisi beberapa command dan main.c
23	09/11/2022	18221054	Michael Sihotang	Mengisi laporan bagian program utama, Test Data, Test Script, dan Algoritma yang menarik.
24	09/11/2022	18221116	Miralistya Cahya F	Mengisi laporan bagian Test Data dan Test Script
25	10/11/2022	18221160	Seren Elizabeth S	Revisi RNG
26	10/11/2022	18221116	Miralistya Cahya F	Push driver Mesin Kata dan Mesin Karakter
27	10//11/2022	18221054	Michael Sihotang	Push driver Queue, QueueDinerDash, dan arraydin
28	10//11/2022	18221054	Michael Sihotang	Push ADT STACK dan Driver ADT STACK
29	10/22/2022	18221112	Imanuel Raditya	Mengerjakan Bonus (Tower of Hanoi)