

LAPORAN TUGAS BESAR

IF2111 Algoritma dan Struktur Data

BNMO Game Console


Dipersiapkan oleh:

Kelompok 06 IF2111 K01

18221065	Josua Adriel Sinabutar
18221121	Rozan Ghosani
18221123	Abraham Megantoro Samudra
18221143	Lie, Kevin Sebastian S. T.
18221157	Cathleen Lauretta

Sekolah Teknik Elektro dan Informatika - Institut Teknologi Bandung

Jl. Ganesha 10, Bandung 40132

	Sekolah Teknik Elektro dan Informatika ITB	Nomor Dokumen		Halaman
		<i>IF2111-TB1-06</i>		<i>40</i>
		<i>Revisi</i>	<i>1</i>	<i>28 Oktober 2022</i>

Daftar Isi

1 Ringkasan	3
2 Penjelasan Tambahan Spesifikasi Tugas	4
2.1 Game ATC	4
3 Struktur Data (ADT)	5
3.1 ADT mesin_karakter	5
3.2 ADT mesin_kata	6
3.3 ADT array	7
3.4 ADT queue	8
3.5 ADT queuedash	9
4 Program Utama	9
5 Algoritma-Algoritma Menarik	10
5.1 Queue Dash	10
5.2 Penggunaan Parameter FILE pada Mesin Kata dan Karakter	10
6 Data Test	11
6.1 Welcome Page	11
6.2 START	12
6.3 LOAD <filename.txt>	12
6.4 SAVE <filename.txt>	12
6.5 CREATE GAME	13
6.6 LIST GAME	13
6.7 DELETE GAME	14
6.8 QUEUE GAME	16
6.9 PLAY GAME	17
6.10 SKIP GAME <n>	17
6.11 QUIT	18
6.12 HELP	18
6.13 COMMAND LAIN	19
6.14 RNG	19
6.15 DINER DASH	20
6.16 ATC GAME	23
7 Test Script	25
8 Pembagian Kerja dalam Kelompok	30

9 Lampiran	31
9.1 Deskripsi Tugas Besar 1	31
Mekanisme Sistem	31
Tentang Sistem	31
Main Menu	31
Command	31
9.2 Notulen Rapat	32
9.3 Log Activity Anggota Kelompok	35

1 Ringkasan

Indra dan Doni memiliki sebuah robot game console yang diberi nama BNMO. Sebelumnya, pada mata kuliah IF1210, BNMO sempat rusak karena dibanting oleh Indra yang stres karena berkuliah di ITB. Namun sayangnya, setelah diperbaiki, kini BNMO memiliki banyak bug di dalam sistemnya sehingga Indra dan Doni sedang mencari programmer yang lebih pandai dari yang sebelumnya untuk memperbaiki BNMO.

Sebagai robot game console, BNMO memiliki peran untuk menjalankan permainan yang terdapat di dalamnya. Terdapat beberapa fitur utama yang dapat dijalankan oleh BNMO, antara lain : memainkan game, menambahkan game, menghapus game, dan mengurutkan game yang akan dimainkan.

Dari tugas besar ini, dapat disimpulkan bahwa persoalan yang dimiliki oleh Indra dan Doni dapat diselesaikan dengan menggunakan ilmu pelajaran yang didapat dari mata kuliah IF2111 Algoritma dan Struktur Data.

2 Penjelasan Tambahan Spesifikasi Tugas

2.1 Game ATC

Permainan ATC (*Air Traffic Controller*) merupakan simulasi dari pekerjaan yang dilakukan ATC di dunia nyata. Pada permainan ini pengguna diberikan dua buah bandara. Pada saat tertentu akan ada pesawat yang datang dengan tipe berbeda bergantung pada jenis bandara yang merupakan pasangannya. Setiap pesawat memiliki jarak yang berbeda-beda terhadap bandaranya dan akan bergerak semakin dekat tiap putaran. Tugas pemain adalah mengatur pesawat mana yang bisa mendarat terlebih dahulu sehingga kecelakaan pesawat ketika ada beberapa pesawat memasuki bandara bersamaan dapat dihindari.

Fitur ini memiliki beberapa primitif

1. `createPesawat(Queue *Pesawat, int ID, char color)` untuk membuat elemen pesawat baru. Pesawat yang dibuat memiliki kondisi awal sembarang dan ketika primitif ini digunakan, pesawat yang dihasilkan akan memiliki panjang lintasan acak dan kode warna sesuai dengan input pengguna. Primitif ini dibuat untuk menanggulangi pembuatan pesawat yang repetitif.
2. `createBandara(Queue *Bandara)` untuk membuat bandara. Masukan awal bandara bersifat sembarang. Ketika selesai dibuat bandara akan kosong. Primitif ini dibuat untuk mempermudah pembuatan dua buah bandara.
3. `printLangit(Langit L)` untuk menampilkan ke layar array yang berisikan pesawat. Pesawat yang ada pada array akan ditampilkan dengan warna yang berbeda-beda bergantung pada jenis pesawat.
4. `printBandara(Queue B, char warna)` untuk menampilkan ke layar queue bandara yang berisikan landasan kosong atau berisikan pesawat.
5. `printGuide()` untuk menampilkan petunjuk permainan.
6. `printStatus(Queue M, Queue B, Langit L)` untuk menampilkan Bandara yang ada, dan kondisi langit terbaru.

7. insertPesawat(Langit *L, Queue Pesawat) untuk menambahkan pesawat baru ke dalam array langit.
8. updateStatus(Queue *M, Queue *B, Langit *L, TabKata *LandingM, TabKata *LandingB) yang digunakan untuk memperbarui kondisi langit dan bandara yang semakin maju setiap putarannya.
9. accPesawat(Langit *L, Word comm) yang digunakan ketika pengguna memasukan perintah untuk mempercepat laju pesawat. Ketika pengguna mempercepat laju pesawat, pesawat akan berada pada elemen pertama.
10. isCommVal(Word comm, Langit L) untuk memvalidasi masukan user. Masukan yang valid di antaranya adalah SKIP dan masukan berupa kode pesawat yang ada di langit.

Fitur ini dijalankan dengan memanggil prosedur playAtc() untuk menjalankan program. Program dimulai dengan menampilkan petunjuk permainan. Kemudian program akan membuat pesawat baru untuk dimasukkan ke langit. Kemudian kondisi terkini bandara dan pesawat akan dipanggil dengan menggunakan printStatus. Setelah state permainan ditampilkan, program akan meminta masukan perintah pengguna untuk mempercepat atau melewati putaran. Semakin lama program berjalan, akan ada semakin banyak pesawat baru yang berada di langit dan harus diatur agar tidak mendarat bertabrakan. Fitur ini diakhiri dengan menampilkan skor user.

3 Struktur Data (ADT)

3.1 ADT mesin_karakter

Struktur data pertama yang digunakan untuk menyelesaikan persoalan ini adalah Mesin Karakter yang diimplementasikan dalam bentuk ADT mesinkarakter. Sketsa struktur data yang ada pada ADT mesinkarakter adalah sebagai berikut :

- Memiliki beberapa primitif, yaitu :
 1. START(FILE *input) yang berfungsi untuk menerima nama file yang akan dibaca dan disimpan dalam bentuk pita yang berisi deret karakter. Karakter pertama yang berada pada pita akan berada pada jendela yang memiliki ukuran sebesar satu karakter. Pita karakter diakhiri dengan MARK, yang dalam persoalan tugas besar ini berupa newline (“\n”).
 2. ADV() yang membuat pita dimajukan sebanyak 1 karakter. Karakter yang terbaca pada jendela akan tersimpan ke dalam variabel currentChar yang memiliki tipe data character.
- State dari Mesin Karakter ditentukan oleh EOP (End of Process) dan currentChar, dimana EOP bertipe boolean yang akan bernilai true jika mesin karakter sudah mencapai MARK. Apabila mesin sudah mencapai EOP, maka mesin tidak lagi dioperasikan.
- Persoalan yang diselesaikan dengan menggunakan ADT ini adalah persoalan mengenai pembacaan karakter dari sebuah file sehingga console game yang dibuat dapat memanfaatkan file konfigurasi.
- Alasan mengapa ADT ini digunakan, selain karena wajib, ADT ini berperan sebagai ADT dasar, dimana ADT mampu membaca tipe data paling dasar yang dikenali oleh bahasa C, juga karena ADT ini digunakan pada implementasi ADT lain.

3.2 ADT mesin_kata

Struktur data yang digunakan selanjutnya adalah Mesin Kata yang bekerja untuk memproses tipe data Word berdasarkan ADT mesinkarakter yang sudah dibuat sebelumnya. Mesin kata yang digunakan diimplementasikan dalam bentuk ADT mesinkata. Adapun sketsa struktur data pada ADT mesinkata adalah sebagai berikut :

- Memiliki tipe data Word yang terdiri dari TabWord yang dapat menampung 50 elemen bertipe character dan integer berupa Length yang menyatakan banyaknya elemen pada TabWord. TabWord digunakan sebagai tempat menampung huruf yang akan dijadikan kata dan Length digunakan sebagai angka yang menyatakan panjang kata tersebut.
- Memiliki 5 primitif, yaitu :
 1. IgnoreBlanks() yang berfungsi untuk mengabaikan currentChar yang bernilai BLANK pada jendela karakter, yaitu spasi (" "), dan memproses ke karakter berikutnya dari pita karakter.
 2. STARTWORD(FILE *input) yang berfungsi untuk membaca file yang akan digunakan pada program.
 3. STARTINPUT() yang berfungsi untuk membaca input dari pengguna tanpa menggunakan scanf.
 4. ADVWORD() yang berfungsi untuk mengakuisisi kata berikutnya dan menyimpannya ke dalam variabel currentWord.
 5. CopyWord() yang berfungsi untuk menyimpan currentChar yang dibaca pada jendela karakter ke dalam variabel currentWord yang memiliki tipe data Word.
- Terdapat beberapa fungsi/prosedur tambahan yang diimplementasikan pada ADT untuk mempermudah proses penanganan persoalan, seperti stringLength(char *str) yang akan mengembalikan integer yang menyatakan jumlah huruf pada kata atau panjang kata tersebut, toKata(char *str) yang akan mengembalikan tipe data Word hasil transformasi array of character yang diterima dari parameter input fungsi, prosedur PrintWord(Word kata) yang akan mencetak variabel kata bertipe Word ke layar, WordCompare(Word currentWord, Word inputWord) yang akan mengembalikan boolean true/false untuk memeriksa apakah 2 buah kata dengan tipe Word merupakan kata yang sama atau bukan, dan lainnya.
- State dari mesin karakter ditentukan oleh EndWord dan currentWord, dimana EndWord bertipe boolean yang akan bernilai true jika mesin kata sudah selesai membaca serangkaian kata (ditandai oleh MARK berupa new line ("\n") atau BLANK berupa spasi (" ")). Kata yang didapat dari mesin kata akan disimpan ke dalam variabel currentWord bertipe Word.
- Persoalan yang diselesaikan dengan menggunakan ADT ini adalah persoalan mengenai pembacaan secara keseluruhan dari nama file dan input pengguna tanpa menggunakan scanf. Command yang diterima dari input user dapat ditransformasi dan diterima oleh program sehingga console game dapat dijalankan.
- Alasan mengapa ADT ini digunakan adalah karena ADT mesinkata dapat menerima input user untuk membaca command yang diterima oleh program sehingga mempermudah proses jalannya program dibandingkan menggunakan ADT mesinkarakter.

3.3 ADT array

Struktur data lain yang digunakan dalam persoalan ini adalah struktur data array. Sesuai definisinya, array yang diimplementasikan disini terdiri atas susunan elemen bertipe sama yaitu tipe data Word yang sudah diimplementasikan pada ADT mesinkata. Array yang digunakan merupakan array statik dengan asumsi game yang tersimpan pada BNMO tidak akan lebih dari 100. Takutnya kalau lebih dari 100 nanti BNMO-nya rusak lagi.. Sketsa struktur data array yang digunakan dapat dirincikan sebagai berikut :

- Memiliki tipe data TabKata yang terdiri dari memori tempat penyimpanan elemen ElType bertipe Word sebanyak 100 elemen dan integer berupa Neff yang menyatakan ada berapa banyak elemen efektif di dalam TabKata. TabKata digunakan sebagai tempat untuk menampung Word, baik yang didapat dari pembacaan file konfigurasi maupun input user.
- Memiliki beberapa fungsi/prosedur yang berperan sebagai konstruktor, selektor, dan operasi untuk elemen di dalamnya, yaitu :
 1. Konstruktor : MakeEmpty(TabKata *T) yang berfungsi untuk membuat array kosong. Array kosong merupakan array dengan 0 elemen efektif.
 2. Selektor : NbElmt(TabKata T) yang akan mengembalikan integer yang menyatakan ada berapa banyak elemen efektif i dalam array, MaxNbEl yang akan mengembalikan berapa banyak elemen yang dapat ditampung oleh array, GetFirstIdx dan GetLastIdx yang akan mengembalikan indeks elemen pertama, indeks elemen terakhir, serta GetElmt yang akan mengembalikan Word yang ditampung pada array di indeks ke-i. Terdapat selektor tambahan lain sebagai hasil ekspansi dari selektor sebelumnya yang berfungsi untuk mempermudah akses elemen dari array.
 3. IsEmpty(TabKata T) yang akan mengembalikan boolean bernilai true/false untuk memeriksa apakah array pada parameter adalah array kosong atau tidak.
 4. IsFull(TabKata T) yang akan mengembalikan boolean bernilai true/false untuk memeriksa apakah array pada parameter sudah penuh atau belum. Hal ini berguna untuk memastikan memori tidak overload karena array yang digunakan untuk menyelesaikan persoalan ini merupakan array statik.
 5. TulisIsi (TabKata T) yang berfungsi untuk mencetak isi dari array ke dalam layar. Karena tipe data yang tertampung di dalam array merupakan tipe data Word, cara akses yang digunakan juga tidak bisa menggunakan cara mencetak yang biasa.
- Persoalan yang diselesaikan menggunakan ADT ini adalah persoalan mengenai daftar game yang terdapat pada video console BNMO. Karena nama game yang terdaftar pada BNMO memiliki tipe data Word, butuh sebuah tempat untuk menampung nama-nama game tersebut sehingga dapat dioperasikan di dalam BNMO.
- Alasan mengapa ADT ini digunakan adalah karena elemen dalam ADT array akan tersimpan secara kontigu, dimana addressnya berurutan dan tidak ada bagian kosong di tengah-tengah array. Array yang digunakan juga array statik dengan asumsi bahwa BNMO tidak akan menampung game lebih dari 100 karena memori yang dimilikinya. Karena array-nya statik, maka tidak perlu dilakukan alokasi dan dealokasi memory sehingga program akan berjalan cukup efisien.

3.4 ADT queue

Struktur data yang digunakan selanjutnya adalah queue. Queue merupakan sederetan elemen yang dikenali berdasarkan elemen pertama (HEAD) dan elemen terakhirnya (TAIL). Konsep struktur data queue hampir mirip seperti array, namun terdapat beberapa aturan penyisipan dan penghapusan yang harus dipenuhi, seperti :

1. Penyisipan yang selalu dilakukan setelah elemen terakhir (TAIL).
2. Penghapusan yang selalu dilakukan pada elemen pertama (HEAD).

Dengan aturan demikian, maka Queue yang elemennya bertipe Word akan diimplementasikan dalam bentuk ADT queue dengan sketsa struktur data sebagai berikut :

- Memiliki tipe data Queue yang terdiri dari memori tempat penyimpanan elemen ElType bertipe Word sebanyak 100 elemen, serta dua integer berupa idxHead dan idxTail yang menyatakan indeks dari elemen pertama dan elemen terakhir Queue. Queue dapat digunakan sebagai “antrian” yang mengikuti aturan FIFO (First In First Out) sehingga elemen yang terdapat didalamnya dapat diakses secara berurutan sesuai dengan urutan masuk ke dalam Queue-nya.
- Memiliki beberapa fungsi/prosedur yang digunakan untuk melakukan operasi terhadap elemen di dalam Queue, yaitu :
 1. CreateQueue(Queue *q) yang berfungsi untuk membuat queue kosong. Queue disebut kosong jika indeks elemen pertama (HEAD) dan indeks elemen terakhirnya (TAIL) merupakan indeks yang tidak terdefinisi (IDX_UNDEF).
 2. isEmpty(Queue q) yang akan mengembalikan boolean bernilai true/false untuk memeriksa apakah queue pada parameter merupakan queue kosong atau bukan.
 3. isFull (Queue q) yang akan mengembalikan boolean bernilai true/false untuk memeriksa apakah queue pada parameter sudah penuh atau belum. Jika queue sudah penuh, maka elemen tidak dapat masuk ke dalam antrian dan harus menunggu elemen pertama keluar terlebih dahulu dari antrian.
 4. length(Queue q) yang akan mengembalikan integer yang menyatakan banyaknya elemen dalam queue.
 5. enqueue(Queue *q, ElType val) yang berfungsi untuk menambahkan val sebagai elemen terakhir dari Queue. Elemen val akan menjadi TAIL baru dari Queue.
 6. dequeue(Queue *q, ElType *val) yang berfungsi untuk menghapus elemen pertama pada Queue. Nilai dari elemen HEAD akan disimpan ada val, kemudian IDX_HEAD akan maju ke elemen setelah HEAD.
- Persoalan yang diselesaikan menggunakan ADT ini adalah persoalan mengenai antrian user untuk memainkan game. Dengan menggunakan ADT queue, susunan antrian akan didapatkan berdasarkan enqueue elemen dari input pemain sehingga ketika command PLAY GAME dipanggil, program hanya perlu melakukan dequeue dari antrian game dan memainkan game tersebut.
- Alasan mengapa ADT ini digunakan adalah karena konsep yang dimiliki oleh struktur data Queue. Karena elemennya tersusun secara FIFO (First In First Out), maka hal tersebut dapat

mempermudah program untuk mengakses nama game yang dimasukkan user ke dalam antrian game.

3.5 ADT *queuedash*

Struktur data yang digunakan selanjutnya adalah *queuedash* yang berguna di game Diner Dash. ADT ini berasal dari ADT *queue* yang dimodifikasi agar dapat mempermudah penyimpanan elemen pada game Diner Dash. Serta penambahan prosedur untuk menampilkan sesuai format pada spesifikasi game. Adapun struktur data pada ADT *queuedash* adalah sebagai berikut :

- Memiliki struktur data buffer yakni array yang berisi elemen dengan tipe *FoodType*, integer *idxHead* dan integer *idxTail*. Sementara itu *FoodType* merupakan struktur data yang berisi integer *foodID*, integer *cookDuration*, integer *sustain*, dan integer *price*
- Permasalahan yang diselesaikan dari ADT ini adalah mempermudah penyimpanan elemen dalam satu index dengan memodifikasi tipe elemen, serta primitif-primitif lain yang digunakan untuk mempermudah algoritma di program utama game DinerDash

4 Program Utama

Program utama BNMO diimplementasikan dalam file *main.c* yang mengandung seluruh file header dari ADT, Game, Driver, beserta file konfigurasi yang ada di dalamnya. Ketika program dipanggil, akan ditampilkan splash screen BNMO yang sedang melambaikan tangan untuk menyambut pemain. Setelah tampilan screen tersebut, program akan meminta user untuk memasukkan command pertama kali namun terbatas pada command *START* dan *LOAD* <filename.txt>. Jika input tidak valid, maka user akan terus diminta untuk memasukkan input sampai valid. Perbedaan *START* dan *LOAD* ada pada data file yang digunakan untuk bermain, dimana *START* akan menggunakan file *config.txt*, sedangkan *LOAD* <filename.txt> akan menggunakan file yang sesuai dengan input user.

Setelah berhasil masuk ke dalam Menu Utama, user akan diminta untuk memasukkan command lain yang terdefinisi di dalam BNMO. Command yang valid adalah command dengan huruf besar dan menggunakan spasi yang memisahkan antar katanya. Untuk melihat daftar perintah yang dapat dijalankan, user dapat memasukkan command “*HELP*” pada BNMO.

Selama user tidak memasukkan command “*QUIT*”, user dapat memainkan permainan di dalam BNMO dengan menggunakan perintah-perintah yang valid di dalamnya. Untuk melihat daftar permainan yang disimpan oleh BNMO, user dapat memasukkan perintah “*LIST GAME*”. Dari sana, user dapat memilih game yang ingin dimainkan dan memasukkan game tersebut ke dalam antrian dengan menggunakan “*QUEUE GAME*”. Jika user merasa bosan dengan game yang terdapat di dalam BNMO, user dapat menambahkan game sendiri dengan command “*CREATE GAME*”. Apabila game yang ingin ditambahkan sudah berada di dalam list game, maka program tidak akan menambahkan game tersebut dan akan meminta user untuk memasukkan command baru. User juga dapat menghapus game yang sudah ditambahkan dalam list menggunakan command “*DELETE GAME*”, tapi tidak dapat menghapus game yang sudah disediakan BNMO dari awal file konfigurasi dibaca.

Untuk memainkan sebuah game, user harus terlebih dahulu memastikan bahwa game yang ingin dimainkan sudah berada di dalam antrian game. Kemudian, user dapat menggunakan command “*PLAY GAME*” dan BNMO akan secara otomatis masuk ke dalam game yang berada

di antrian pertama. Apabila user sempat salah memasukkan game ke dalam antrian atau mendadak tidak ingin memainkan game tersebut, BNMO memiliki fitur skip game yang dapat melewati antrian game sebanyak n kali (tergantung jumlah antriannya) dengan command “SKIP GAME”.

Setelah selesai bermain, user dapat menyimpan kembali daftar game ke dalam BNMO dengan menggunakan command “SAVE <filename.txt>”. Namun, ketika SAVE dipanggil, BNMO tidak akan langsung ter-shut down. User harus memanggil command “QUIT” terlebih dahulu dan program akan menanyakan apakah progress permainan yang sudah dimainkan ingin di save atau tidak.

5 Algoritma-Algoritma Menarik

5.1 Queue Dash

Algoritma ini merupakan adaptasi ADT queue yang disesuaikan dengan struktur data Food dalam game diner dash. ADT ini memodifikasi ADT queue dibagian ElType menjadi ADT baru yang berisi 4 buah integer yakni ID Food, CookDuration, Sustain, dan Price. ADT ini dapat mempermudah dalam penyimpanan data dari suatu index, yakni bisa menyimpan 5 buah data. Modifikasi ini dapat membantu algoritma game diner dash serta menghemat queue yang digunakan.

```
/* Definisi elemen dan address */
typedef struct {
    int foodID;
    int cookDuration;
    int sustain;
    int price;
} FoodType;

typedef struct {
    FoodType buffer[CAPACITYDASH];
    int idxHead;
    int idxTail;
} QueueDash;
```

Gambar 1. Struktur tipe data pada QueueDash

5.2 Penggunaan Parameter FILE pada Mesin Kata dan Karakter

Algoritma ini merupakan modifikasi dari ADT mesin kata. Pada ADT ini, STARTWORD dimodifikasi sehingga menerima inputan berupa FILE* sehingga hal ini dapat menghemat penggunaan fungsi dalam membaca file atau membaca input dari user karena menggunakan fungsi yang sama pengimplementasiannya.

```

void STARTWORD(FILE *input) {
    START(input);
    IgnoreBlanks();
    if (currentChar == MARK)
    {
        EndWord = true;
    }
    else
    {
        EndWord = false;
        CopyWord();
    }
}

```

Gambar 2. Source code prosedur STARTWORD(FILE *input)

6 Data Test

6.1 Welcome Page

Test ini menguji fitur pembuka dari program BNMO. Pengecekan dilakukan dengan menjalankan perintah berikut di terminal.

```

gcc -o main main.c src/createGame.c src/deleteGame.c src/help.c src/listGame.c src/load.c
src/playGame.c src/queueGame.c src/quit.c src/save.c src/skipGame.c src/start.c
src/ADT/mesinkata.c src/ADT/mesinkarakter.c src/ADT/array.c src/ADT/queue.c
src/ADT/queuedash.c src/Driver/readTxtFile.c src/Driver/writeTxtFile.c
src/Games/dinnerdash.c src/Games/rng.c src/Games/atcGame.c

```

Perintah berikut akan menjalankan program dan menghasilkan tampilan berikut di layar.

```

Welcome to BNMO!

Ketik 'START' atau 'LOAD <filename.txt>' untuk mulai bermain
ENTER COMMAND: 

```

Gambar 3. Tampilan Pembuka

Dapat dilihat bahwa akan ditampilkan halaman pembuka dari program BNMO dan akan meminta perintah START atau LOAD.

6.2 START

Test ini menguji fitur START yang akan menjalankan program BNMO dan mengambil data game dari file txt konfigurasi *default*. Setelah memasukan perintah ini, pengguna dapat memasukan perintah-perintah lain untuk menjalankan program.

```
File konfigurasi sistem berhasil dibaca. BNMO berhasil dijalankan.  
ENTER COMMAND: █
```

Gambar 4. Tampilan START

6.3 LOAD <filename.txt>

Test ini menguji fitur LOAD yang akan menjalankan program BNMO dan mengambil data game dari nama file *txt* yang diinginkan. Setelah memasukan perintah ini, pengguna dapat menjalankan program dan memainkan game yang terdapat di dalam nama file yang dimasukan. Fitur ini bisa mengembalikan dua hasil, yakni file berhasil dibaca, atau file gagal dibaca (file tidak ditemukan) dan menampilkan perintah tidak dikenali .

```
ENTER COMMAND: LOAD percobaanSave.txt
```

Gambar 5. Perintah LOAD

Berikut ini merupakan hasil dari tampilan fitur yang mungkin didapatkan

```
Save file berhasil dibaca. BNMO berhasil dijalankan.  
ENTER COMMAND:
```

Gambar 6. Tampilan LOAD berhasil dijalankan

```
Perintah tidak dikenali  
ENTER COMMAND:
```

Gambar 7. Tampilan LOAD gagal dijalankan

6.4 SAVE <filename.txt>

Test ini menguji fitur SAVE yang akan menyimpan state game ke dalam nama file yang dimasukkan. Setelah memasukan perintah ini, state game akan tersimpan ke dalam file dengan nama sesuai masukan pengguna. File dapat menimpa file yang sudah ada atau menciptakan file baru dengan nama file yang dimasukan.

Untuk melakukan fitur save, dapat digunakan perintah “SAVE <nama file>.txt”. Berikut adalah contoh penggunaan perintah SAVE.

```
ENTER COMMAND: SAVE percobaanSave.txt
```

Gambar 8. Perintah SAVE.

Setelah perintah save, terdapat dua kemungkinan yang akan terjadi. Pertama, file berhasil disimpan. Kedua, file gagal disimpan. Kemungkinan file gagal disimpan sangatlah kecil mendekati nol dan ketika file gagal disimpan akan ditampilkan pesan “Save file gagal disimpan.”. Berikut ini merupakan tampilan ketika file berhasil disimpan dan hasil file yang telah disimpan.

```
Save file berhasil disimpan.  
ENTER COMMAND: _
```

Gambar 9. Tampilan perintah SAVE berhasil digunakan.

Name	Date modified	Type	
config	10/11/2022 13:16	Text Docum	6 RNG
percobaanSave	11/11/2022 14:16	Text Docum	Diner DASH DINOSAUR IN EARTH RISEWOMAN EIFFEL TOWER ATC GAME

Gambar 10. Hasil file yang tersimpan.

6.5 CREATE GAME

Test ini menguji fitur CREATE GAME yang akan menambahkan game baru sesuai dengan nama yang diinginkan pengguna. Setelah memasukan perintah ini, game yang ditambahkan akan tersimpan pada state list game yang ada. Game yang dimasukkan tidak bisa memiliki nama yang sama dengan game yang sudah ada.

```
Masukkan nama game yang akan ditambahkan:
```

Gambar 11. Tampilan CREATE GAME

6.6 LIST GAME

Test ini menguji fitur LIST GAME yang akan menampilkan game yang ada pada state program saat digunakan. Setelah memasukan perintah ini, game yang dimiliki akan ditampilkan ke layar.



Gambar 12. Tampilan LIST GAME

6.7 DELETE GAME

Test ini menguji fitur DELETE GAME yang akan menghapus game yang ada pada state program berdasarkan input nomor game dari pengguna. Setelah memasukan perintah ini, game yang dihapus akan hilang dari state list game yang ada.

ENTER COMMAND: DELETE GAME _

Gambar 13. Command DELETE GAME

Pada fitur ini terdapat beberapa batasan pada game yang bisa dihapus. Game yang dihapus haruslah game buatan pengguna, bukan game yang berada dalam file konfigurasi default. Jika pengguna mencoba menghapus game default, maka akan ditampilkan “Fitur gagal dihapus”. Jika pengguna memilih game buatan, maka game akan berhasil dihapus dan menampilkan pesan “Game berhasil dihapus”.

1. RNG
2. Diner DASH
3. DINOSAUR IN EARTH
4. RISEWOMAN
5. EIFFEL TOWER
6. ATC GAME

Masukkan nomor game yang akan dihapus: 5

Game gagal dihapus

ENTER COMMAND:

Gambar 14. Tampilan game di konfigurasi sistem gagal dihapus

1. RNG
2. Diner DASH
3. DINOSAUR IN EARTH
4. RISEWOMAN
5. EIFFEL TOWER
6. ATC GAME
7. KAKOI

Masukkan nomor game yang akan dihapus: 7

Game berhasil dihapus

ENTER COMMAND:

Gambar 15. Tampilan game di konfigurasi sistem berhasil dihapus

[illegible]

Gambar 16. Tampilan list game setelah game dihapus

6.8 QUEUE GAME

Test ini menguji fitur QUEUE GAME yang akan menambahkan game ke dalam antrian sesuai dengan input dari pengguna. Setelah memasukan perintah ini, game yang ditambahkan akan tersimpan pada antrian game yang ada. Pada fitur ini dapat terjadi dua hasil, yakni game yang dipilih tidak valid karena diluar dari pilihan game yang ada; dan game berhasil ditambahkan ke dalam antrian.

```
(\)(\)) (\)(\)
((((( ))) )
(\)(\)) (\)(\)

|_| |_| |_| |_| |_| |_| |_|

Belum ada daftar antrian.

Berikut adalah daftar game yang tersedia
1. RING
2. Diner DASH
3. DINOSAUR IN EARTH
4. RISEWOMAN
5. EIFFEL TOWER
6. ATC GAME

Nomor Game yang mau ditambahkan ke antrian: 7
Nomor permainan tidak valid, silahkan masukkan nomor game pada list.

|_| |_| |_| |_| |_| |_| |_|

ENTER COMMAND: |
```

Gambar 17. Tampilan antrian diluar jumlah daftar game yang tersedia

[illegible]

Gambar 18. Tampilan antrian didalam jumlah daftar game yang tersedia

```
Berikut adalah daftar antrian game-mu
1. ATC GAME
```

Gambar 19. Tampilan daftar antrian

6.9 PLAY GAME

Test ini menguji fitur PLAY GAME yang akan menjalankan game yang terdapat di dalam antrian paling atas. Setelah memasukan perintah ini, game akan dijalankan dan pengguna dapat bermain sesuai dengan game yang dipilih. Pada fitur ini, game nomor 3, 4, dan 5 tidak dapat dimainkan dan akan menampilkan pesan seperti yang ada di gambar berikut.

```
Berikut adalah daftar Game-mu
1. DINOSAUR IN EARTH

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

Gambar 20. Tampilan game tidak dapat dimainkan

```
Berikut adalah daftar Game-mu
1. Diner DASH

Loading Diner DASH ..
```

Gambar 21. Tampilan game berhasil dimainkan

6.10 SKIP GAME <n>

Test ini menguji fitur SKIP GAME yang akan melewati game sejumlah yang dimasukkan oleh pengguna. Setelah memasukan perintah ini, game akan dilewati sebanyak input pengguna dan game yang terdapat di antrian paling atas akan dijalankan. Pada fitur ini terdapat dua hasil yang mungkin keluar. Jika antrian game kosong karena dilewati hingga habis, maka akan menampilkan pesan “Tidak ada lagi dalam daftar game-mu”. Jika ternyata masih terdapat game yang tersisa pada antrian maka akan menjalankan game yang paling atas setelah dilewati.

```
Berikut adalah daftar Game-mu
Tidak ada permainan lagi dalam daftar game-mu.
ENTER COMMAND:
```

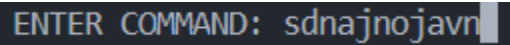
Gambar 22. Tampilan game habis setelah dilewati

```
Berikut adalah daftar Game-mu
1. ATC GAME
2. DINOSAUR IN EARTH
Game DINOSAUR IN EARTH masih dalam maintenance, belum dapat dimainkan. Silahkan pilih game lain.
```

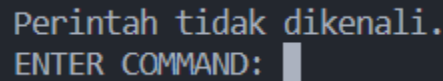
Gambar 23. Tampilan game berhasil dilewati

6.13 COMMAND LAIN

Test ini menguji fitur pada BNMO yang tidak dapat menerima command di luar apa yang sudah didefinisikan di dalam program. Setelah user memasukkan perintah yang tidak valid, maka BNMO akan mengeluarkan output bahwa perintah tidak dikenali dan akan meminta user untuk memasukkan perintah yang valid.



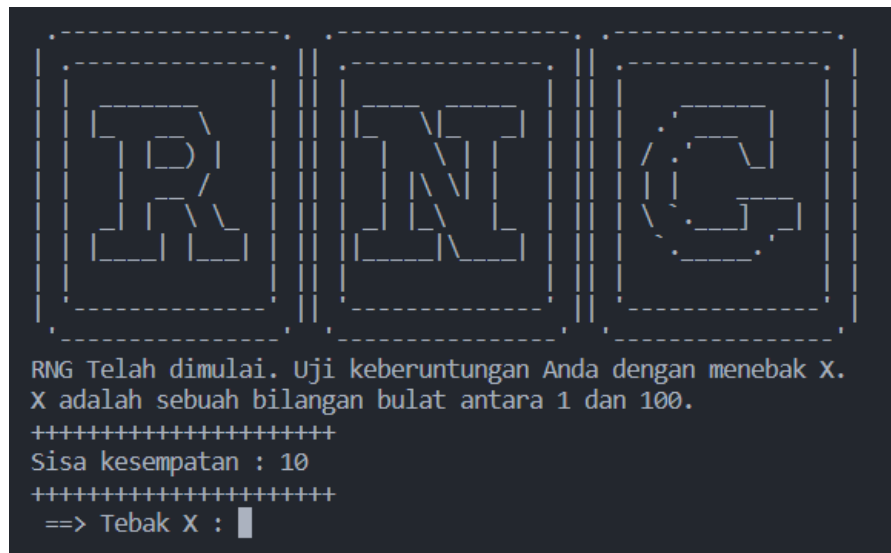
Gambar 28. Tampilan ketika user memasukkan command lain



Gambar 29. Tampilan pada program yang menolak command lain

6.14 RNG

Test ini menguji salah satu game yang terdapat dalam BNMO, yaitu RNG. Game ini meminta user untuk menebak angka X yang berada dalam rentang 1-100. User diberikan 10 kali kesempatan sebelum game over karena gagal menebak X.



Gambar 30. Tampilan Awal game RNG

Setelah user diminta untuk memasukkan angka dalam rentang 1-100, program akan memberitahu apakah X lebih besar atau lebih kecil dari input user. Setiap user menebak angka, maka kesempatan untuk menebak akan berkurang satu.

```

==> Tebak X : 36
~ LEBIH BESAR ~
+++++
Sisa kesempatan : 9
+++++
==> Tebak X : █

```

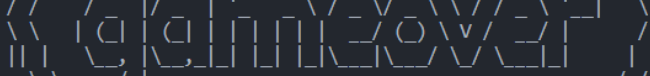
Gambar 31. Tampilan RNG ketika user sudah memasukkan angka random

```
==> Tebak X : 90
~ LEBIH KECIL ~
+++++
Sisa kesempatan : 7
+++++
==> Tebak X : 
```

Gambar 32. Tampilan RNG ketika user sudah memasukkan angka random

Ketika user dapat menebak X sebelum kesempatan menebak habis, maka user dianggap berhasil dan game over.

```
==> Tebak X : 87  
Selamat! Anda berhasil menebak X.  
  
=====
```



```
=====
```

SKOR KAMU : 4000

```
=====
```

ENTER COMMAND:

Gambar 33. Tampilan ketika user berhasil menebak X dan game over

6.15 DINER DASH

```

C:\Windows\System32\cmd.e  x  +  v

[===== Selamat Datang di =====]

[=====]

[=====]

SALDO : 0

Daftar Pesanan
Makanan | Durasi memasak | Ketahanan | Harga
-----
M0      | 1               | 1         | 24000
M1      | 2               | 2         | 30000
M2      | 3               | 2         | 14000

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

Daftar Makanan yang sedang dimasak
Makanan | Sisa ketahanan makanan
-----
|

MASUKKAN COMMAND :

```

Gambar 34. Tampilan Awal dari Game Diner Dash

```

[===== Selamat Datang di =====]

[=====]

[=====]

SALDO : 0

Daftar Pesanan
Makanan | Durasi memasak | Ketahanan | Harga
-----
M0      | 1               | 1         | 24000
M1      | 2               | 2         | 30000
M2      | 3               | 2         | 14000

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

Daftar Makanan yang sedang dimasak
Makanan | Sisa ketahanan makanan
-----
|

MASUKKAN COMMAND : COOK M0
Berhasil memasak M0
=====

```

Gambar 35. User memasukkan command COOK M0

```

[===== Selamat Datang di =====]

[=====]

[=====]

SALDO : 0

Daftar Pesanan
Makanan | Durasi memasak | Ketahanan | Harga
-----
M0      | 1               | 1         | 24000
M1      | 2               | 2         | 30000
M2      | 3               | 2         | 14000
M3      | 3               | 3         | 28000

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

Daftar Makanan yang sedang dimasak
Makanan | Sisa ketahanan makanan
-----
|

MASUKKAN COMMAND : COOK M1
Berhasil memasak M1
=====

```

Gambar 36. User memasukkan command COOK M1

```

[===== Selamat Datang di =====]

[=====]

[=====]

SALDO : 0

Daftar Pesanan
Makanan | Durasi memasak | Ketahanan | Harga
-----
M0      | 1               | 1         | 24000
M1      | 2               | 2         | 30000
M2      | 3               | 2         | 14000
M3      | 3               | 3         | 28000
M4      | 1               | 4         | 12000

Daftar Makanan yang sedang dimasak
Makanan | Sisa durasi memasak
-----
M1      | 2

Daftar Makanan yang sedang dimasak
Makanan | Sisa ketahanan makanan
-----
M0      | 1
MASUKKAN COMMAND : SERVE M0
Berhasil menyajikan M0
=====

```

[illegible][illegible][illegible]

Gambar 41. User memasukkan command SKIP 2x dan game over

Test ini menguji salah satu fitur permainan yang ada yakni Air Traffic Controller. Pada awal permainan ini akan ditampilkan panduan permainan sebelum permainan dimulai.

Gambar 42. Tampilan pertama ketika game dimuat

```
SCORE: 0

Bandara Merah
+-----+
| - - - - - |
+-----+

Bandara Biru
+-----+
| - - - - - |
+-----+

[ Langit ]

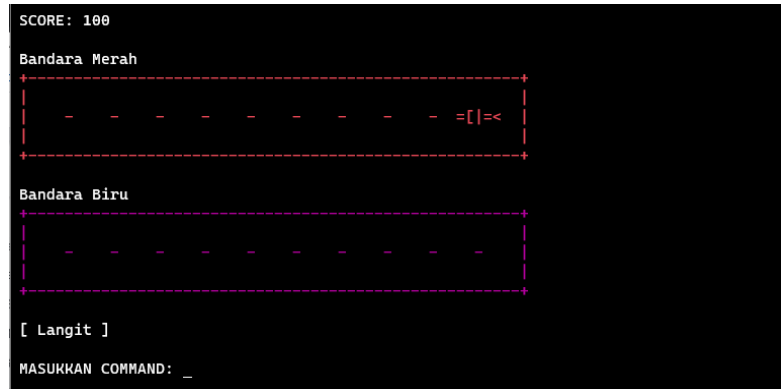
0- - - - - - - - - [ ] =< -o M0

MASUKKAN COMMAND:
```

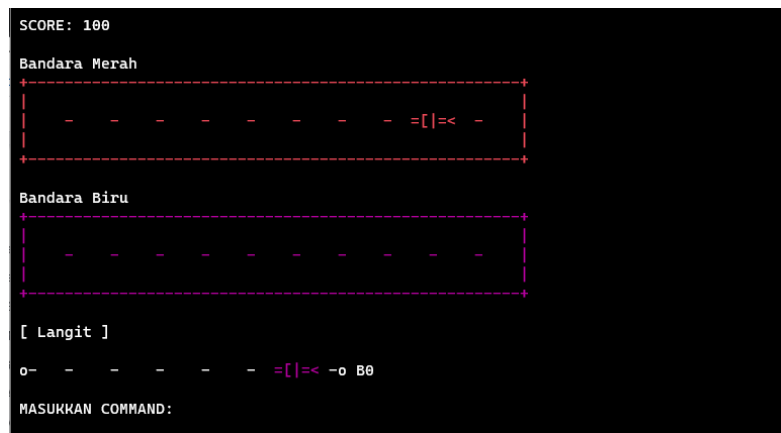
```
MASUKKAN COMMAND: ASDASD

Command tidak valid.
MASUKKAN COMMAND:
```

Gambar 44. Tampilan ketika perintah tidak valid

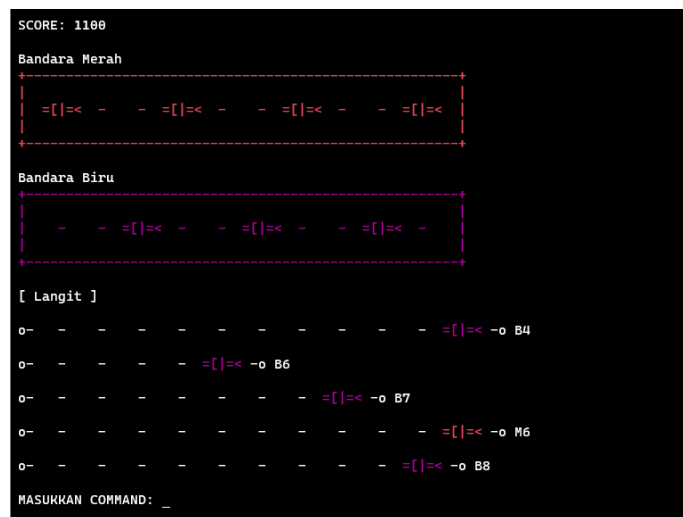


Gambar 45. Tampilan ketika perintah valid M0



Gambar 46. Tampilan ketika perintah SKIP

Setelah memainkan game beberapa saat, permainan akan berjalan semakin sulit. Banyaknya pesawat yang ada di langit akan semakin banyak seiring waktu. Berikut adalah tampilan ketika permainan sudah berjalan selama beberapa saat.



Gambar 47. Tampilan ketiga permainan sudah berjalan beberapa saat

Permainan akan berhenti ketika ada pesawat dengan jarak yang sama memasuki bandara. Berikut adalah tampilan ketika permainan berakhir.

```
0- - - - =[]=< -o B10
0- =[]=< -o B11
0- =[]=< -o M8
0- =[]=< -o B12
0- - - - - =[]=< -o M9
0- - =[]=< -o B13
```

Gambar 48. Kondisi pesawat B11 dan B12 bertabrakan

```
O- - - - - =[]=< -o M20  
MASUKKAN COMMAND: SKIP  
  
=====
```

Game over! Skor akhir 1200
Terjadi kecelakaan pada pesawat B11 B12 _

Gambar 49. Tampilan ketika permainan berakhir

7 Test Script

No.	Fitur yang Dites	Tujuan Testing	Langkah-Langkah Testing	Input Data Test	Hasil yang Diharapkan	Hasil yang Keluar
1	Fitur START	Memastikan bahwa game BNMO dapat dimulai, mengambil data game ke dalam array list game, dan menampilkan tampilan pesan yang sesuai.	1. Memasukkan command START	START	1. Program membaca file default config.txt 2. Program menampilkan pesan "File konfigurasi sistem berhasil dibaca. BNMO berhasil dijalankan."	Sesuai yang diharapkan
2	Fitur LOAD	Memastikan apakah file konfigurasi yang sudah di save sebelumnya dapat dibaca dan isi file disimpan di dalam array global	1. Memasukkan command LOAD diikuti nama file (LOAD <filename.txt>)	LOAD haha.txt	1. Program membaca file test.txt 2. Array global diisi nilai yang terdapat pada file test.txt	Sesuai yang diharapkan

3	Fitur SAVE	Memastikan state game tersimpan pada nama file yang diinginkan pengguna	1. Memasukkan perintah SAVE <filename.txt>	SAVE test.txt	1. Program menyimpan state game dari array global ke dalam folder data dengan nama file yang dimasukkan pengguna.	Sesuai yang diharapkan
4	Fitur CREATE GAME	Memastikan apakah game yang dimasukan oleh pengguna ditambahkan ke array global	1. Memasukkan perintah CREATE GAME 2. Memasukkan nama game 3. Memanggil perintah LIST GAME untuk memastikan game baru ada di dalam game	CREATE GAME game123 LIST GAME	1. Program menyimpan nama game baru ke dalam array global	Sesuai yang diharapkan
5	Fitur LIST GAME	Memastikan bahwa program menampilkan semua game yang ada di file.txt yang di gunakan.	1. Memasukan perintah LIST GAME	LIST GAME	1. Program menampilkan list game yang ada di dalam file.txt yang digunakan	Sesuai yang diharapkan
6	Fitur DELETE GAME	Menghapus game sesuai keinginan user yang terdapat pada array global	1. Memasukkan perintah DELETE GAME 2. Input nomor game yang ingin dihapus	DELETE GAME 7 LIST GAME	1. Game yang dihapus merupakan game diluar file konfigurasi awal sehingga game berhasil dihapus	Sesuai yang diharapkan
7	Fitur QUEUE GAME	Memastikan bahwa program memasukan game yang dipilih pengguna ke dalam antrian game yang siap dimainkan.	1. Memasukan perintah QUEUE GAME 2. Memasukan nomor game yang ingin dimasukan ke dalam antrian	QUEUE GAME 5	1. Program menampilkan pesan “Game berhasil dimasukkan ke dalam antrian”	Sesuai yang diharapkan
8	Fitur PLAY GAME	Memastikan bahwa program menjalankan game yang ingin dimainkan oleh pengguna	1. Memasukkan perintah PLAY GAME	PLAY GAME	1. Program menjalankan game yang berada di antrian paling atas	Sesuai yang diharapkan

		berdasarkan antrian game.				
9	Fitur SKIP GAME	Memastikan bahwa program melewati game sejumlah yang dimasukkan oleh pengguna dan memainkan game yang ada di antrian paling atas setelah dilewati.	1. Memasukkan perintah SKIP GAME (jumlah game yang ingin dilewati)	SKIP GAME 2	1. Program melewati 2 game dalam antrian 2. Program menjalankan game dalam antrian ke- 3	Sesuai yang diharapkan
10	Fitur QUIT	Memastikan bahwa program menanyakan pengguna apakah ingin menyimpan state game saat ini atau tidak, mengeksekusi masukan pengguna, dan keluar dari program BNMO.	1. Memasukkan perintah QUIT 2. Memasukkan pilihan apakah ingin menyimpan state game atau tidak	QUIT Y	1. Program meminta input Y/ N sebagai dasar eksekusi penyimpanan file 2. Jika Y maka program memanggil fungsi SAVE 3. Jika tidak, tidak melakukan apapun 4. Keluar dari game	Sesuai yang diharapkan
11	Fitur HELP	Memastikan bahwa program menampilkan pesan bantuan yang menjelaskan perintah yang dapat dimasukkan oleh user beserta kegunaannya	1. Memasukkan perintah HELP	HELP	1. Program menampilkan pesan bantuan yang menjelaskan perintah yang dapat dimasukkan oleh user beserta kegunaannya	Sesuai yang diharapkan
12	COMMAND LAIN	Memastikan bahwa program tidak menerima input diluar dari perintah-perintah yang sudah dibuat	1. Mencoba memasukkan input asal	test123	1. Program menampilkan pesan "Perintah tidak dikenali"	Sesuai yang diharapkan
13	DINER DASH COOK	Berguna untuk memasak orderan yang ada di Queue Order	1. Memasukkan perintah COOK <IDMakanan>	COOK M0	1. Program menampilkan "Memasak M0" jika M0 ada pada queue order	Sesuai yang diharapkan

					2. Program menampilkan gagal memasak jika M0 tidak terdapat di order 3. Program akan menambah antrian pada order serta memasukkan order yang bersangkutan ke dalam queue akanan yang sedang dimasak dan juga mengurangi durasi cook dan ketahanan makanan	
14	DINER DASH SERVE	Berguna untuk menyajikan makanan yang sudah siap dimasak yakni ketika cook duration makanan sudah mencapai 0 serta id makanan yang bersangkutan berada di HEAD queue order	1. Memasukkan perintah SERVE <IDMakanan>	SERVE M0	1. Program menampilkan “Berhasil menyajikan M0” jika M0 sudah selesai dimasak dan M0 berada di bagian HEAD queue order 2. Program menampilkan gagal menyajikan M0 jika M0 tidak terdapat di daftar makanan yang sudah bisa di serve atau M0 tidak berada di bagian HEAD order 3. Program akan menambah antrian pada order serta mendequeue queue order dan mendelete ID makanan pada daftar serve jika makanan berhasil di serve,	Sesuai yang diharapkan

					selain itu juga mengurangi durasi cook dan ketahanan makanan	
15	DINER DASH SKIP	Berguna untuk melewati 1 putaran dalam game, yakni mengurangi durasi memasak dan ketahanan serta menambah Beorder	1. Memasukkan command SKIP	SKIP	1. Program menampilkan “Berhasil melewati 1 putaran”, menambah orderan, serta mengurangi durasi memasak dan ketahanan makanan	
16	DINER DASH COMMAND LAIN	Berguna untuk mengatasi command yang tidak valid	1. Memasukkan command selain command yang valid diatas yaitu COOK SERVE SKIP	TUTUP TOKO	1. Program akan menampilkan bahwa command yang diinput tidak valid serta meminta inputan ulang dari user	Sesuai yang diharapkan
17	RNG INPUT TEBAKAN	Berguna untuk menerima inputan tebakkan user	1. Memasukkan angka random	45	1. Program akan menampilkan bahwa tebakkan lebih besar, lebih kecil atau sama dengan jawaban jika inputan yang valid yakni 1 sampai 100 2. Program akan menerima inputan kembali jika inputan sebelumnya diluar batas yakni diluar 1 sampai 100	Sesuai yang diharapkan
18	ATC INPUT SKIP	Berguna untuk melewati 1 putaran game.	1. Memasukkan command SKIP	SKIP	1. Program memperbarui state permainan, pesawat maju satu langkah.	Sesuai yang diharapkan
19	ATC INPUT B0/M0	Berguna untuk mempercepat laju pesawat ke bandara	1. Memasukkan command M0/B0	M0 / B0	1. Program memasukkan pesawat yang	Sesuai yang diharapkan

					dipilih ke bandara	
20	ATC INPUT COMMAN D LAIN	Berguna untuk mengatasi command yang tidak valid	1. Memasukkan command selain kode pesawat atau SKIP	HELLO	1. Program menampilkan bahwa input tidak valid dan meminta ulang perintah	Sesuai yang diharapkan

8 Pembagian Kerja dalam Kelompok

NIM	Nama	Deskripsi Tugas
18221065	Josua Adriel S.	<ol style="list-style-type: none"> 1. Membuat ADT array. 2. Membuat prosedur start, listGame, dan help. 3. Membuat driver ADT queue. 4. Membuat laporan bagian Data Test, Test Script, deskripsi tugas besar 5. Melakukan debugging dan revisi kode
18221121	Rozan Ghosani	<ol style="list-style-type: none"> 1. Membuat prosedur save, skipGame, dan playGame. 2. Membuat driver untuk membaca dan menulis txt file ke dalam bahasa C. 3. Membuat driver ADT array. 4. Membuat custom game bonus (Air Traffic Controller Game). 5. Membuat laporan bagian Spesifikasi Tambahan untuk custom game.
18221123	Abraham Megantoro S.	<ol style="list-style-type: none"> 1. Membuat repository bersama. 2. Membuat ADT mesinkarakter dan mesinkata. 3. Membuat prosedur load, deleteGame, dan queueGame. 4. Membuat main program dan debugging untuk seluruh fungsi/prosedur. 5. Merevisi kode 6. Membuat laporan bagian Data Test dan Test Script
18221143	Lie, Kevin Sebastian S.T.	<ol style="list-style-type: none"> 1. Membuat ADT queue 2. Membuat realisasi game RNG dan Diner Dash. 3. Memodifikasi mesin kata dan mesin karakter. 4. Membuat laporan bagian Struktur Data ADT queuedash, Test Script, Algoritma Menarik. 5. Membuat driver ADT QueueDash. 6. Melakukan debugging dan revisi kode.

18221157	Cathleen Lauretta	<ol style="list-style-type: none"> 1. Narahubung asisten dengan kelompok 2. Membuat prosedur createGame dan quit. 3. Membuat beberapa fungsi dari ADT mesinkata. 4. Membuat driver ADT mesinkarakter dan mesinkata. 5. Membuat laporan bagian cover, Struktur Data (mesinkarakter, mesinkata, array, queue), Program Utama, dan Data Test Diner Dash.
----------	-------------------	--

9 Lampiran

9.1 Deskripsi Tugas Besar 1

Latar Belakang

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.

Spesifikasi Umum

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

Mekanisme Sistem

Tentang Sistem

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

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.

Command

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

1. START
2. LOAD <filename>
3. SAVE <filename>

4. CREATE GAME
5. LIST GAME
6. DELETE GAME
7. QUEUE GAME
8. PLAY GAME
9. SKIP GAME <n>
10. QUIT
11. HELP
12. COMMAND LAIN


9.2 Notulen Rapat






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

No. Kelompok/Kelas : 06 / K01
 Nama Kelompok : -
 Anggota Kelompok (Nama/NIM) :
 1. 18221065 / Josua Adriel Sinabutar
 2. 18221121 / Rozan Ghosani
 3. 18221123 / Abraham Megantoro Samudra
 4. 18221143 / Lie, Kevin Sebastian S. T.
 5. 18221157 / Cathleen Lauretta

Asisten Pembimbing : 13519064 / Aditya Bimawan

Asistensi I






Tanggal : 4 November 2022	Catatan Asistensi: <ol style="list-style-type: none"> Implementasi mesin katanya ikutin yang awal aja, gapapa dibaca per row. Boleh modifikasi mesin kata yang udah ada. Command-command yang dibuat lebih baik berada di file terpisah karena lebih enak dibaca daripada digabung di satu file besar. Struktur ADT-nya(?) juga boleh dipecah-pecah lagi asal strukturnya jelas. Driver untuk setiap ADT jangan lupa dibuat (ada di spesifikasi, wajib) dan masuk nilai. Cara nge-testnya bebas, yang penting filenya ada dan test case-nya jangan sedikit. Jumlah file driver yang dibuat harus sesuai dengan jumlah ADT yang dibuat. Misal : Ada 4 ADT, maka ada 4 file driver. Github commit tidak terlalu dinilai karena pekerjaan satu kelompok bisa berada di satu device. Pengecekan kontribusi nanti akan dilihat dari Peer Assessment.
Tempat : Zoom Meeting	
Kehadiran Anggota Kelompok: No NIM Tanda tangan 1 18221065  2 18221121	


 3 18221123  4 18221143  5 18221157 	5. Sekali-kali cek spesifikasi , karena bisa saja ada revisi spesifikasi. 6. Fileheader “.h” bagusya ada komentar.
	Tanda Tangan Asisten: 

Asistensi II

Tanggal : 10 November 2022	Catatan Asistensi:
Tempat : Zoom Meeting & Selasar Hidden	

1. Pada Queue Game diperbolehkan game yang sama.
2. Perlu dibuat driver tambahan untuk ADT tambahan yang tidak masuk spesifikasi.
3. Jika menggunakan library tambahan selain yang ada di spesifikasi (bukan untuk algoritma utama), coba ditanyakan pada FAQ Tugas Besar.
4. Log Activity untuk membuat laporan juga dimasukkan saja.

<p>Kehadiran Anggota Kelompok:</p> <p>No NIM Tanda tangan</p> <p>1 18221065</p>  <p>2 18221121</p>  <p>3 18221123</p>  <p>4 18221143</p>  <p>5 18221157</p> 	<ol style="list-style-type: none"> 5. Coba diperbaiki Delete Game nya karena ketika dicoba saat asistensi sempat ada bug. 6. Tidak perlu menggunakan Ubuntu untuk medemonstrasikan program (Windows saja cukup). 7. Driver ADT yang dibuat tidak perlu didemokan. Cukup main programnya saja. 8. Terminal boleh diwarnai (menggunakan library stdlib.h).
	<p>Tanda Tangan Asisten:</p>

	
--	--

9.3 Log Activity Anggota Kelompok

Tanggal	NIM	Nama	Aktivitas
29/10/2022	18221123	Abraham Megantoro	Membuat repository di github dan mulai menambahkan anggota kelompok sebagai collaborator
31/10/2022	-	-	Melakukan pembagian tugas per spesifikasi fungsi/prosedur yang diperlukan
31/10/2022	18221143	Lie, Kevin Sebastian	Membuat ADT Queue
31/10/2022	18221065	Josua Adriel	Membuat ADT Array
31/10/2022	18221123	Abraham Megantoro	Membuat ADT Mesin Karakter dan Mesin Kata
31/10/2022	18221065	Josua Adriel	Melakukan revisi terhadap tipe data ElType (int jadi char*)
31/10/2022	18221143	Lie, Kevin Sebastian	Membuat sebagian besar dari game Diner Dash
01/11/2022	18221123	Abraham Megantoro	Membuat prosedur deleteGame, load, dan queueGame
01/11/2022	18221157	Cathleen Lauretta	Mengontak asisten untuk menjadwalkan asistensi pertama
01/11/2022	18221121	Rozan Ghosani	Memodifikasi ADT yang sudah ada (mengubah tipe data TabInt menjadi TabKata)
01/11/2022	18221121	Rozan Ghosani	Membuat fungsi untuk membaca file txt ke dalam program
02/11/2022	18221121	Rozan Ghosani	Membuat prosedur playGame, save, dan skipGame
02/11/2022	18221065	Josua Adriel	Membuat prosedur start dan listGame

02/11/2022	18221065	Josua Adriel	Melakukan revisi terhadap ADT (mengubah include .c menjadi .h)
02/11/2022	18221123	Abraham Megantoro	Melakukan merging terhadap fungsi dan prosedur yang sudah direalisasikan pada branch masing-masing anggota
02/11/2022	18221157	Cathleen Laurretta	Membuat prosedur createGame
03/11/2022	18221157	Cathleen Laurretta	Memodifikasi ADT mesinkarakter dan ADT mesinkata (tidak menggunakan scanf lagi, melainkan stdin) dan menambahkan beberapa fungsi tambahan pada ADT
03/11/2022	18221157	Cathleen Laurretta	Melakukan revisi terhadap prosedur createGame
03/11/2022	18221123	Abraham Megantoro	Melakukan merging ke main dari pull request
03/11/2022	18221123	Abraham Megantoro	Melakukan revisi terhadap ADT mesinkata serta prosedur deleteGame dan listGame
03/11/2022	18221123	Abraham Megantoro	Mengubah directory dari data dan readme file
03/11/2022	18221121	Rozan Ghosani	Melakukan revisi terhadap input dari prosedur save
03/11/2022	18221123	Abraham Megantoro	Melakukan merging ke main dari pull request
03/11/2022	18221121	Rozan Ghosani	Melakukan revisi terhadap fungsi/prosedur skipGame dan playGame
03/11/2022	18221121	Rozan Ghosani	Menambahkan ADT mesinkarakter dan ADT mesinkata khusus untuk pita file
03/11/2022	18221065	Josua Adriel	Melakukan merging main ke branch dan merging lagi ke main

03/11/2022	18221123	Abraham Megantoro	Melakukan merging ke main dari pull request
04/11/2022	18221143	Lie, Kevin Sebastian	Melakukan modifikasi terhadap ADT mesinkarakter dan ADT mesinkata (supaya dapat menerima tipe data Kata yang tidak mengandung spasi di dalamnya)
04/11/2022	18221143	Lie, Kevin Sebastian	Mengembalikan bentuk ADT ke sebelum modifikasi
04/11/2022	18221121	Rozan Ghosani	Melakukan revisi terhadap fungsi/prosedur skipGame dan save (sehingga bisa menyimpan langsung di data)
04/11/2022	18221065	Josua Adriel	Membuat prosedur help
04/11/2022	18221123	Abraham Megantoro	Melakukan merging ke main dari pull request
04/11/2022	18221157	Cathleen Lauretta	Membuat laporan bagian cover dan ringkasan
04/11/2022	18221123	Abraham Megantoro	Melakukan revisi terhadap ADT-ADT yang ada dan membuat main program
04/11/2022	18221143	Lie, Kevin Sebastian	Melakukan modifikasi terhadap ADT mesinkarakter dan ADT mesinkata dan menambahkan prosedur CLOSE pita
04/11/2022	18221123	Abraham Megantoro	Melakukan merging ke main dari pull request
04/11/2022	-	-	Melakukan asistensi pertama tugas besar
08/11/2022	18221157	Cathleen Lauretta	Membuat prosedur quit dan driver untuk ADT mesinkata
08/11/2022	18221065	Josua Adriel	Melakukan merging main ke branch dan merging lagi ke main

08/11/2022	18221123	Abraham Megantoro	Melakukan update terhadap directory dari driver ADT dan main program
08/11/2022	18221121	Rozan Ghosani	Melakukan revisi terhadap fungsi/prosedur txt reader, playGame, skipGame, save, load, dan WordToStr
08/11/2022	18221123	Abraham Megantoro	Melakukan merging ke main dari pull request
08/11/2022	18221123	Abraham Megantoro	Melakukan revisi terhadap fungsi/prosedur deleteGame, queueGame, main program dan merapikan ADT array
08/11/2022	18221143	Lie, Kevin Sebastian	Melakukan finalisasi terhadap game Diner Dash dan RNG
08/11/2022	18221123	Abraham Megantoro	Melakukan merging ke main dari pull request
09/11/2022	18221123	Abraham Megantoro	Melakukan modifikasi terhadap game dan ADT Queue (mengubah Queue menjadi QueueDash)
09/11/2022	18221065	Josua Adriel	Melakukan perbaikan terhadap bug yang ada pada fungsi ketika dijalankan di main program
09/11/2022	18221065	Josua Adriel	Membuat driver ADT Queue
09/11/2022	18221157	Cathleen Lauretta	Membuat Log Activity anggota kelompok
09/11/2022	18221123	Abraham Megantoro	Melakukan revisi terhadap fungsi txt reader, load, dan start serta merapikan fungsi/prosedur displayQueueGame, listGame, dan skipGame
09/11/2022	18221123	Abraham Megantoro	Merapikan struktur main program
09/11/2022	18221121	Rozan Ghosani	Memperbaiki bug yang ada pada fungsi load
09/11/2022	18221143	Lie, Kevin Sebastian	Melakukan update terhadap game dan save ketika memanggil prosedur quit

09/11/2022	18221121	Rozan Ghosani	Membuat game custom (Air Traffic Control) dan driver ADT array
09/11/2022	18221123	Abraham Megantoro	Melakukan merging ke main dari pull request
09/11/2022	18221065	Josua Adriel	Melakukan revisi terhadap driver ADT queue
09/11/2022	18221143	Lie, Kevin Sebastian	Menambahkan interface untuk game yang sudah dibuat dan membuat driver ADT Queue Dash
09/11/2022	18221123	Abraham Megantoro	Melakukan merging ke main dari pull request
10/11/2022	18221157	Cathleen Laurretta	Membuat laporan bagian Struktur Data Mesin Karakter
10/11/2022	18221123	Abraham Megantoro	Melakukan merging ke main dari pull request
10/11/2022	18221143	Lie, Kevin Sebastian	Memperbaiki bug yang ada pada main program
10/11/2022	18221123	Abraham Megantoro	Melakukan revisi terhadap prosedur playGame dan skipGame
10/11/2022	-	-	Melakukan asistensi kedua tugas besar dan melakukan demo untuk mengecek bug yang ada pada main program
10/11/2022	-	-	Memperbaiki bug yang terdapat pada main program (prosedur deleteGame)
10/11/2022	18221123	Abraham Megantoro	Melakukan merging ke main dari pull request
10/11/2022	18221157	Cathleen Laurretta	Memperbaiki bug yang terdapat pada prosedur createGame
10/11/2022	18221157	Cathleen Laurretta	Membuat laporan bagian Struktur Data Mesin Kata dan Array
10/11/2022	18221143	Lie, Kevin Sebastian	Membuat laporan bagian Struktur Data Queue Dash

10/11/2022	18221121	Rozan Ghosani	Membuat laporan bagian Spesifikasi Tambahan untuk Custom Game
11/11/2022	18221123	Abraham Megantoro	Melakukan merging ke main dari pull request
11/11/2022	18221157	Cathleen Lauretta	Memperbaiki input command untuk prosedur skipGame
11/11/2022	18221157	Cathleen Lauretta	Membuat driver ADT mesinkarakter dan memperbaiki driver ADT mesinkata
11/11/2022	18221157	Cathleen Lauretta	Membuat laporan bagian Struktur Data Queue dan Program Utama
11/11/2022	18221065	Josua Adriel	Membuat laporan bagian Data Test
11/11/2022	18221123	Abraham Megantoro	Membuat laporan bagian Data Test
11/11/2022	18221121	Rozan Ghosani	Membuat laporan bagian Data Test dan Algoritma Menarik
11/11/2022	18221143	Lie, Kevin Sebastian	Membuat laporan bagian Test Script dan Data Test Diner Dash