

LAPORAN TUGAS BESAR

IF2111 Algoritma dan Struktur Data


Bimono

Dipersiapkan oleh:

Kelompok 4

I Dewa Made Manu Pradnyana	18221047
Laurentia Kayleen Christopher	18221053
Felisa Aidadora Darmawan	18221137
Ananda Abdul Hafizh	18221167
Hans Stephano Edbert N	18221171

Sekolah Teknik Elektro dan Informatika - Institut Teknologi Bandung
Jl. Ganesha 10, Bandung 40132

	Sekolah Teknik Elektro dan Informatika ITB	Nomor Dokumen		Halaman
		<i>IF2111-TB1-04</i>		<i>34</i>
		<i>Revisi</i>	<i>0</i>	<i>11 November 2022</i>

Daftar Isi

1 Ringkasan	3
2 Penjelasan Tambahan Spesifikasi Tugas	
2.1 Spesifikasi Game Bonus Marvel Snap	4
3 Struktur Data	
3.1 ADT array	6
3.2 ADT mesin karakter	6
3.3 ADT mesin kata	7
3.4 ADT queue	7
4 Program Utama	
4.1 Main Menu	9
4.2 Inisialisasi dan File Konfigurasi	9
4.3 Command START	10
4.4 Command LOAD <filename>	10
4.5 Command SAVE <filename>	11
4.6 Command CREATEGAME	11
4.7 Command LISTGAME	11
4.8 Command DELETGAME	12
4.9 Command QUEUEGAME	12
4.10 Command PLAYGAME	12
4.11 Command SKIP GAME <n>	12
4.12 Command QUIT	13
4.13 Command HELP	13
4.14 Game RNG	12
4.15 Game Diner Dash	1
5 Data Test	
6.1 START	16
6.2 LOAD <filename>	16
6.3 CREATEGAME	17
6.4 LISTGAME	17
6.5 DELETGAME	18
6.6 QUEUEGAME	20
6.7 PLAYGAME	21
6.8 SKIPGAME <n>	22
6.9 HELP	24
6.10 SAVE <namafilename>	24
6.11 QUIT	25
6 Test Script	26
7 Pembagian Kerja dalam Kelompok	28
8 Lampiran	
8.1 Deskripsi Tugas Besar 2	30
8.2 Notulen Rapat	30
8.3 Log Activity Anggota Kelompok	34

1 Ringkasan

Tugas ini melibatkan sebuah robot video game console bernama BNMO. Karena beberapa hal, BNMO sempat rusak, namun sudah diperbaiki—tentu saja, ada beberapa hal yang kurang sejak perbaikan tersebut. Fitur-fitur yang ada pada BNMO harus dikembalikan lagi. Oleh karena itu, kelompok kami berusaha untuk melengkapi kembali fitur-fitur BNMO agar bisa dimainkan lagi.

Laporan ini terdiri atas beberapa bagian utama. Setelah ringkasan ini, akan ada penjelasan tambahan mengenai spesifikasi tugas yang perlu diketahui oleh pembaca, penjelasan struktur data atau ADT yang digunakan, penjelasan program utama yang kami buat, hasil dari *data test* yang dilakukan pada main program kami, skenario *test* yang kami gunakan seperti ekspektasi output dari setiap command di program, pembagian kerja kelompok kami untuk melihat kontribusi dan siapa saja yang melakukannya, serta lampiran penting yang dibutuhkan untuk pengertian lebih lanjut mengenai laporan ini.

Sebagai kesimpulan, BNMO dibuat dengan menggunakan bahasa C dengan menggunakan beberapa ADT yang sudah dipelajari di mata kuliah IF2111 Algoritma dan Struktur Data. ADT yang digunakan antara lain adalah ADT array, ADT mesin karakter, ADT mesin kata, dan ADT queue.

2 Penjelasan Tambahan Spesifikasi Tugas

Pada tugas bonus, kelompok kami membuat sebuah game dengan judul MARVEL SNAP. Game ini terinspirasi dari game MARVEL SNAP yang baru saja release secara global pada tanggal 18 Oktober 2022. Untuk aturan dari game ini bisa dilihat dibawah ini :

- a. Game ini dimainkan dengan dua player dimana player 1 memulai turn terlebih dahulu
- b. Game ini dimainkan dalam enam turn dan pada masing turn, player mendapatkan energy sebesar jumlah turn
- c. Di awal Game setiap player mendapat tiga kartu dengan setiap player pasti mendapatkan kartu dengan cost 1, serta pada awal turn, setiap pemain mengambil 1 kartu random
- d. Terdapat tiga arena dimana pada masing - masing arena maksimal dimasukkan empat kartu untuk masing - masing player
- e. Pada setiap giliran, player dapat memasukkan kartu kedalam arena atau melakukan SKIP
- f. Pemain dapat memasukkan kartu lebih dari satu pada setiap turn asalkan energy masih cukup
- g. Board akan di update setelah kedua player menyelesaikan turn
- h. Pada akhir turn 6, setiap arena akan diadu power totalnya. Untuk setiap arena dengan total power lebih banyak, player yang memenangkan arena tersebut mendapatkan skor 1
- i. Player yang menang adalah player dengan skor lebih banyak
- j. Jika skor seri, player yang menang adalah player dengan total power pada seluruh arena lebih besar

Pada game ini, terdapat beberapa fitur tambahan yang diterapkan untuk membuat game ini, diantaranya :

STEI- ITB	IF2111-TB1-04	Halaman 3 dari 35 halaman
Template dokumen ini dan informasi yang dimilikinya adalah milik Sekolah Teknik Elektro dan Informatika ITB dan bersifat rahasia. Dilarang me-reproduksi dokumen ini tanpa diketahui oleh Sekolah Teknik Elektro dan Informatika ITB.		

2.1 Spesifikasi Game Bonus “Marvel Snap”

Pada game “Marvel Snap” terdapat beberapa spesifikasi dan fitur tambahan untuk menjalankan program tersebut, yaitu :

a. Fitur randomCard

Fitur ini berfungsi untuk mengenerate kartu secara random dengan skala energy dari 1 - 6 dan skala power berdasarkan energy kartu :

- Energy card 1 --> Power antara 1 - 2
- Energy card 2 --> Power antara 2 - 4
- Energy card 3 --> Power antara 3 - 6
- Energy card 4 --> Power antara 4 - 8
- Energy card 5 --> Power antara 5 - 10
- Energy card 6 --> Power antara 6 - 12

b. Fitur PrintArena

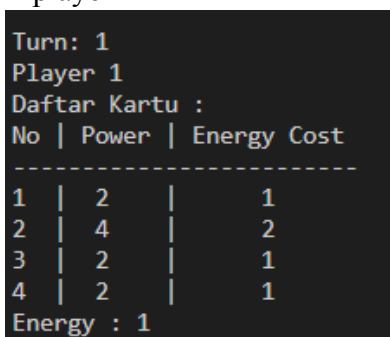
Fitur ini berfungsi untuk print arena yang sedang berlangsung. Berikut adalah contoh gambar arena permainan



Gambar 2.1

c. Fitur PrintCard

Fitur ini berfungsi untuk print kartu yang dimiliki oleh player. Berikut adalah contoh gambar kartu yang dimiliki player



Gambar 2.2

d. Fitur PlayerTurn

Fitur ini berfungsi dalam mengatur alur giliran dari setiap pemainnya. Terdapat beberapa alur yang dilewati diantaranya :

- Fase Memilih Kartu
Pada fase ini player dapat memilih kartu yang ingin dimasukkan ke dalam arena
- Fase Memilih Arena
Pada fase ini player dapat memilih arena yang ingin dimasukkan kartu. Pilihan arena yang valid adalah 1 - 3
- Fase Validasi Input
Pada fase ini, player akan ditanyakan apakah sudah yakin untuk memilih kartu dan arena tersebut. Jika “Yes” maka kartu yang dipilih akan dimasukkan ke dalam arena yang dipilih. Jika “No” maka akan balik kembali ke fase memilih Kartu
- Fase Giliran Tambahan
Pada fase ini, player dapat memilih untuk memasang kartu kembali jika masih terdapat sisa energy, Jika “Yes” maka kembali ke fase memilih kartu dengan kartu pilihan sebelumnya berhasil dipasang ke arena pilihan player. Jika “No” maka giliran akan berpindah ke player berikutnya

```

Turn: 1
Player 2
Daftar Kartu :
No | Power | Energy Cost
-----
1 | 1 | 1
2 | 4 | 2
3 | 2 | 2
4 | 7 | 6
Energy : 1
Masukan nomor kartu yang ingin dimainkan: 1
Masukan arena yang ingin dimasukkan kartu(1/2/3): 1
Apakah Anda yakin?(YES/NO) YES

```

Gambar 2.3

3 Struktur Data (ADT)

3.1 ADT array

ADT array pada file `array.h` digunakan untuk menyelesaikan berbagai persoalan mengenai elemen-elemen yang ada di *games* tertentu. Suatu `TabWord` akan memiliki elemen `ElType`, kapasitas `TabWord`, dan `Neff` atau jumlah elemen yang terdapat di `TabWord`. Setiap elemen `TabWord` yang bertipe `ElType` merupakan elemen dengan tipe `Word`, yang akan diperjelas pada ADT mesinkata. `TabWord` menjadi sebuah tempat penampungan elemen *games*.

Pada ADT array, terdapat fungsi *MakeTabWord* yang akan membentuk `TabWord` kosong, *DeallocateTabWord* yang akan men-dealokasi `TabWord`, *isEmpty* untuk memeriksa `TabWord` kosong atau tidak, *Length* yang mengembalikan jumlah elemen pada array, *Get* untuk mengembalikan elemen array pada indeks tertentu, *GetCapacity* untuk mendapatkan jumlah kapasitas tersedia, *InsertAt*, *InsertLast*, *InsertFirst*, untuk menambahkan suatu elemen bertipe `ElType` pada lokasi `TabWord` tertentu, *DeleteAt*, *DeleteLast*, *DeleteFirst*, untuk menghapus suatu elemen bertipe `ElType` pada lokasi array tertentu, *PrintTabWord* untuk menampilkan isi dari `TabWord`, *CopyTabWord* untuk menyalin suatu `TabWord`, dan *SearchTabWord* untuk mencari indeks dimana sebuah elemen bertipe `ElType` pada `TabWord` disimpan. Alasan pemilihan ADT array adalah dengan tujuan dapat mengakses suatu *game* dan elemennya secara terstruktur sehingga mempermudah untuk *command* lainnya. ADT array dideklarasikan di `array.h`, dan diimplementasikan pada file `array.c`.

3.2 ADT mesin karakter

ADT mesin karakter pada file `mesinkarakter.h` digunakan untuk menyelesaikan persoalan terkait membaca pita karakter. Suatu elemen dengan tipe `char` bernama `CC` adalah *current character* atau karakter yang saat ini sedang dibacakan oleh program. Ada sebuah elemen bernama `EOP` yang menandakan bahwa pita sudah berada pada akhirnya dan sudah saatnya berhenti membaca pita tersebut. Ada tiga prosedur di ADT mesin karakter ini, yaitu *loadstart*, *cmdstart*, dan *adv*. *Loadstart* adalah prosedur yang akan menerima sebuah *input* berupa pita karakter dan tanda bahwa mesin sudah siap dioperasikan. *Cmdstart* adalah prosedur yang membacakan *command* dari program. *Adv* adalah prosedur yang akan memajukan pembacaan ke karakter berikutnya supaya bisa dilanjutkan hingga nanti mencapai akhir dari pita tersebut.

ADT mesin karakter dideklarasikan di `mesinkarakter.h`, dan diimplementasikan pada file `mesinkarakter.c`.

3.3 ADT mesin kata

ADT mesin kata pada file mesinkata.h digunakan untuk menyelesaikan berbagai persoalan terkait kata melalui input user. Suatu elemen dengan tipe Word akan memiliki akses untuk mendapatkan karakter ke sekian dari kata tersebut, dan jumlah kata tersebut. Dalam ADT mesin kata, terdapat fungsi *STARTWORD* yang akan membuka file dan menerima kata, *ADVWORD* yang akan berpindah ke kata berikutnya, *COPYWORD* yang akan mengakuisisi kata dan menyimpan kata tersebut dalam *currentWord*, *IgnoreBlanks* yang akan mengabaikan satu atau beberapa BLANK, *STARTCMD* yang akan menerima kata melalui input user, *ADVCMD* yang akan berpindah ke kata berikutnya, *COPYCMD* yang akan mengakuisisi command dan menyimpan kata dalam *currentCommand*, *IgnoreBlanksCMD* yang akan mengabaikan satu atau beberapa BLANK, *toWord* yang akan mengubah suatu elemen bertipe string ke elemen bertipe Word, *toString* yang akan mengubah suatu elemen bertipe Word menjadi elemen bertipe string, *toInt* yang akan mengubah suatu elemen bertipe Word menjadi elemen bertipe integer, *compareWord* yang membandingkan sebuah elemen bertipe Word dengan sebuah string dan mengembalikan true jika string dan kata sama, false jika berbeda, *compare2Word* yang membandingkan dua buah elemen bertipe Word, dan mengembalikan true jika kedua elemen sama, *strLength* yang mengembalikan panjang suatu string kata, dan *printWord* yang akan menampilkan isi kata pada layar. Alasan pemilihan ADT mesin kata adalah dengan tujuan dapat mendapat alternatif untuk membaca input dari user tanpa menggunakan fungsi *scanf*, dan mempermudah berjalannya fungsi *start*, *load*, *create game*, *delete game*, dan *quit*, dengan memanfaatkan berbagai fungsi dalam ADT mesin kata. ADT mesin kata dideklarasikan di *mesinkata.h* dan diimplementasikan pada file *mesinkata.c*.

3.4 ADT queue

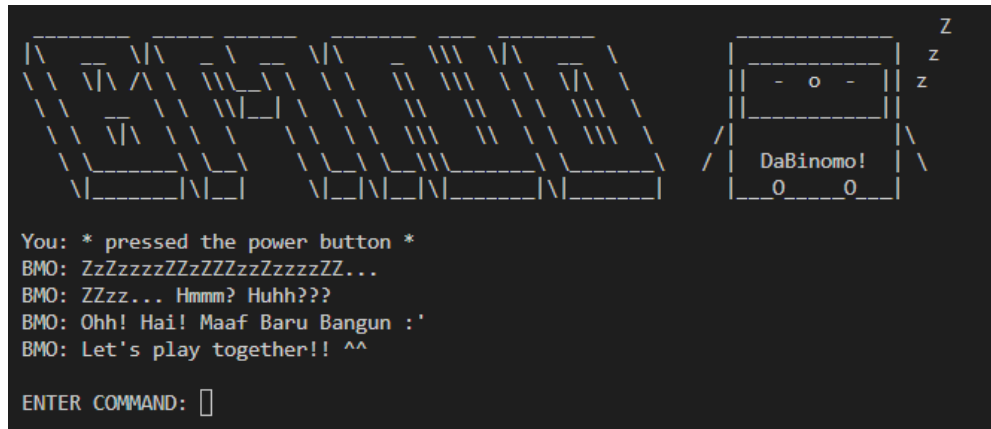
ADT queue pada file *queue.h* digunakan untuk menyelesaikan persoalan terkait dengan antrian game yang akan ditampilkan serta diedit (memainkan game di antrian, menambahkan game ke antrian, dll) sesuai kebutuhan program. Kami mendefinisikan ADT Queue dengan representasi array secara eksplisit dan alokasi statik. Dalam definisi elemen dan address, sebuah elemen *ElType* ber-tipe Word, sesuai dengan ADT mesin kata, yang menjadi penampung “kata” dalam Queue. Struktur dari Queue itu sendiri terdiri dari array bertipe *ElType*, indeks Head, dan indeks Tail. Dalam ADT, beberapa fungsi yang bisa digunakan untuk fungsi-fungsi dalam program adalah *CreateQueue* yang akan membuat queue kosong, *isEmpty* dan *isFull* untuk mengecek apakah kosong/penuh, *length* untuk mengetahui panjang queue, *enqueue* dan *dequeue* untuk memasukkan/mengeluarkan elemen dari queue, *displayQueue* untuk mencetak atau menampilkan isi dari queue, dan *isInQueue* yang akan mengecek apakah elemen yang sedang kita cari atau kita butuhkan ada di dalam queue tersebut. Alasan dipilihnya ADT queue adalah untuk mempermudah menyelesaikan persoalan pada queue game, play game, dan skip game,

yaitu keharusan untuk menampilkan “antrian” game yang akan lebih efektif dan efisien implementasinya jika menggunakan ADT queue. ADT queue dideklarasikan di queue.h dan implementasinya ada pada file queue.c.

4 Program Utama

4.1 Main Menu

Ketika program dijalankan pertama kali, BNMO akan memperlihatkan main menu yang berisi welcome page dan beberapa menu pilihan yaitu START dan LOAD. Setelah itu, pengguna diminta untuk memasukkan command dan program akan menjalankan fungsi sesuai input pengguna.



Gambar 4.1

4.2 Inisialisasi dan File Konfigurasi

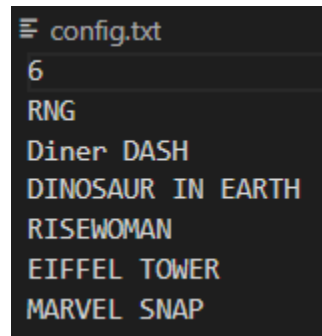
Setelah program mulai dijalankan, program akan membuat `listGame` bertipe `TabWord` untuk menyimpan list game yang dimiliki user, `queueGame` bertipe `Queue` untuk menyimpan antrian game, serta `listCommand` bertipe `TabWord` untuk menyimpan command yang diinput oleh user kata per kata. `TabWord` command akan didealokasi setiap setelah user menginput command dan dialokasi setiap sebelum user menginput command. Program juga menyimpan boolean `active` untuk memeriksa apakah program masih berjalan serta boolean `loaded` untuk memeriksa apakah program sudah melakukan konfigurasi.

```
Queue queueGame; CreateQueue(&queueGame);
TabWord listGame; MakeTabWord(&listGame);
TabWord listCommand; MakeTabWord(&listCommand);

boolean active = true;
boolean loaded = false;
```

Gambar 4.2

Adapun file konfigurasi yang kami gunakan berjumlah enam game dengan struktur file konfigurasi sebagai berikut.



```
config.txt
6
RNG
Diner DASH
DINOSAUR IN EARTH
RISEWOMAN
EIFFEL TOWER
MARVEL SNAP
```

Gambar 4.3

4.3 Command START

Ketika command START dipanggil, program akan langsung membaca konfigurasi file (config.txt) dan memasukkannya ke dalam array listgame. Prosedur START dijalankan dengan memanfaatkan ADT mesinkata dan array dimana ADT mesinkata digunakan untuk membaca konfigurasi file, sedangkan ADT array digunakan sebagai tempat untuk menyimpan konfigurasi file yang dibaca menggunakan mesinkata. Pada prosedur START diminta parameter filename bertipe char*, listgame bertipe TabWord, serta loaded bertipe boolean (untuk mengetahui apakah file konfigurasi berhasil dibaca atau tidak). Command START hanya dapat dipanggil sekali ketika user belum melakukan START atau LOAD <filename>, sehingga jika user ingin melakukan START kembali, user harus melakukan QUIT terlebih dahulu dan menjalankan program kembali.

4.4 Command LOAD <filename>

Ketika command LOAD <filename> dipanggil, program akan langsung membaca file <filename> dan memasukkannya ke dalam array listgame. Jika <filename> yang dimasukkan tidak terdapat dalam folder data, maka LOAD gagal dilakukan. Prosedur LOAD dijalankan dengan memanfaatkan ADT mesinkata dan array dimana ADT mesinkata digunakan untuk membaca konfigurasi file, sedangkan ADT array digunakan sebagai tempat untuk menyimpan konfigurasi file yang dibaca menggunakan mesinkata. Pada prosedur LOAD diminta parameter filename bertipe char*, listgame bertipe TabWord, serta loaded bertipe boolean (untuk mengetahui apakah file <filename> berhasil dibaca atau tidak). Command LOAD hanya dapat dipanggil sekali ketika user belum melakukan START atau LOAD <filename>, sehingga jika user ingin melakukan LOAD <filename> kembali, user harus melakukan QUIT terlebih dahulu dan menjalankan program kembali.

4.5 Command *SAVE <filename>*

Ketika command *SAVE <filename>* dipanggil, program akan menjalankan fungsi *SAVE* dan membaca masukan *filename* yang dipanggil bersama-sama dengan command *SAVE*. *Filename* berisikan nama file yang akan digunakan untuk penyimpanan konfigurasi pengguna yang berisikan Game yang ada ketika command *SAVE* dipanggil. Jika file tidak ditemukan dalam folder *data* maka akan dibuat file teks (.txt) baru. Bila pengguna tidak memasukkan nama file setelah command *SAVE*, maka akan ditampilkan pesan *error* “Mohon masukkan dengan format yang benar!”. Jika masukan valid, ketika proses penyimpanan berhasil dilakukan, program akan menampilkan “Berhasil menyimpan progress.”, sementara ketika proses penyimpanan gagal, program akan menampilkan “Gagal membuka / memuat file. Silahkan coba lagi!”

4.6 Command *CREATE GAME*

Ketika command *creategame* dipanggil, pengguna akan diminta untuk memasukkan nama game yang akan ditambahkan, lalu menambahkan game tersebut ke dalam TabWord *listGame*. Game baru yang ditambahkan pengguna dapat dihapus, namun game-game yang terdapat dalam file *config.txt*, yakni RNG, Diner Dash, DINOSAUR IN EARTH, RISEWOMAN, EIFFEL TOWER, dan MARVEL SNAP, tidak dapat dihapus. Keberhasilan command *CREATE GAME* ditentukan juga oleh keberadaan suatu game dalam LIST GAME. Jika tidak terdapat nama game yang sama dalam LIST GAME, program akan mengeksekusi *CREATE GAME* dan menampilkan “Game berhasil ditambahkan”. Sementara itu, jika sudah terdapat nama game yang sama dalam LIST GAME, program akan memberikan keluaran “Game sudah terdaftar dalam "LIST GAME"!”. Pada kasus ketika pengguna memasukkan *string* kosong, program akan menampilkan pesan *error* “Mohon masukkan masukan yang valid!”.

4.7 Command *LIST GAME*

Ketika command *listgame* dipanggil, akan ditampilkan daftar game yang tersedia. Command ini memanfaatkan fungsi primitif ADT array, yaitu *PrintTabWord*.

4.8 Command **DELETE GAME**

Ketika command `deletgame` dipanggil, akan ditampilkan daftar game yang dimiliki sistem. Kemudian, pengguna akan diminta untuk memasukan input nomor game yang ingin dihapus, dan jika memenuhi syarat bahwa game dapat dihapus, yaitu game yang ingin dihapus bukan RNG, Diner Dash, DINOSAUR IN EARTH, RISEWOMAN, atau EIFFEL TOWER, maka game akan dihapus dari TabWord `listgame`. Jika game berhasil dihapus, program akan menampilkan pesan “Game berhasil dihapus!”, namun jika game tidak berhasil dihapus, program akan menampilkan pesan “Game gagal dihapus!”.

4.9 Command **QUEUE GAME**

Ketika command `queuegame` dipanggil, pertama akan ditampilkan antrian game atau ‘`queuegame`’ yang dimiliki oleh user dengan fungsi `displayQueue` yang dapat digunakan dengan memanggil file ‘`queue.h`’. Kemudian, ditampilkan juga daftar game dalam bentuk TabWord ‘`listgame`’ yang ada pada program dengan fungsi `TulisIsi` dari file ‘`array.h`’. Dari situ, akan diminta sebuah input dari user untuk menambahkan game ke dalam antrian. Ketika sudah diterima, program akan mengecek sesuai dengan list game. Apabila input yang diterima adalah ‘2’, maka program akan mengecek game manakah yang memiliki ‘`id = 2`’. Lalu, informasi yang telah didapat itulah yang akan dimasukkan ke dalam queue dengan fungsi `enqueue` yang dapat digunakan dengan memanggil ‘`queue.h`’.

4.10 Command **PLAY GAME**

Ketika command `playgame` dipanggil, antrian game seperti yang ada pada command `queuegame` akan kembali ditampilkan. Setelah itu, secara otomatis game pada antrian paling pertama akan dimainkan atau dieksekusi oleh program. Hal ini juga akan otomatis mengeluarkan game tersebut dari antrian, karena dianggap sudah bukan termasuk antrian lagi dengan cara *di-dequeue* (fungsi diambil dari `queue.h`). Akan tetapi, hanya RNG, Diner DASH, dan MARVEL SNAP yang bisa dimainkan. Oleh karena itu, salah satu fitur

4.11 Command **SKIP GAME <n>**

Ketika command `skip game` dipanggil, antrian game seperti yang ada pada command `queuegame` juga akan kembali ditampilkan. Bersama dengan command `skip game` itu sendiri akan dimasukkan juga sebuah input “n” yang berisi angka. Angka tersebut adalah jumlah game yang mau di-skip atau tidak dimainkan dari antrian yang ada. Misal, apabila $n = 2$, maka jika di antrian game ada tiga game, dua game pertama akan di-skip atau *di-dequeue* tanpa dimainkan, dan game terakhir tetap akan *di-dequeue*, tapi dimainkan. Apabila n lebih dari antrian di game, maka program akan menampilkan kata-kata “Tidak ada permainan lagi dalam daftar game-mu.”

4.12 Command *QUIT*

Command *QUIT* merupakan command yang dipanggil ketika pengguna ingin keluar dari program. Pemanggilan command ini dipengaruhi oleh keadaan konfigurasi BNMO. Jika BNMO belum terkonfigurasi, maka command *QUIT* akan menampilkan “pesan perpisahan”. Sementara itu, dalam keadaan sudah terkonfigurasi, maka command *QUIT* akan menawarkan pengguna untuk melakukan *SAVE* atau tidak. Jika ‘Ya’, maka pengguna akan memasukkan nama file tujuan dan memanggil fungsi *SAVE*. Jika ‘Tidak’, maka program akan menampilkan “pesan perpisahan”.

4.13 Command *HELP*

Command *HELP* akan menampilkan daftar *command* (perintah) yang valid dan penjelasan singkat untuk mempermudah pengguna untuk memahami fungsi dari setiap *command*. Command *HELP* dipengaruhi oleh keadaan konfigurasi BNMO. Jika BNMO belum terkonfigurasi, maka command yang akan ditampilkan command *HELP* terbatas pada *START*, *LOAD <filename.txt>*, *HELP*, dan *QUIT*. Jika BNMO sudah terkonfigurasi, maka keseluruhan command akan ditampilkan kepada pengguna.

4.14 Game *RNG*

File *RNG.c* ini berisi dua fungsi. Fungsi pertama adalah fungsi untuk mengacak sebuah bilangan, sedangkan fungsi kedua kita adalah fungsi utama (*int main*). Dua fungsi ini bekerja secara linear dalam satu kali panggil file *RNG.c*.

Fungsi acak yang digunakan adalah *acakAngka* tidak menerima input variabel, namun menghasilkan sebuah input integer. Fungsi ini memanggil library *math.h*, *stdio.h*, *stdlib.h*, dan *time.h*, ADT Mesin Karakter, dan file ‘*boolean.h*’ yang akan mendefinisikan tipe boolean. Fungsi *acakAngka* ini membatasi angka yang di acak. Angka akan diacak dan dihasilkan output integer dengan range bilangan diantara (0 sampai 100).

Selanjutnya untuk fungsi utama itu sendiri akan diberikan skor awal (maksimal) dengan nilai sebesar 100 point. Point sempurna akan diberikan bila mana user berhasil menebak angka di kesempatan pertama. Untuk alur berpikir dari fungsi utama ini adalah program akan mengacak sebuah bilangan dengan memanggil fungsi *acakAngka*. Selanjutnya user akan diminta untuk memasukkan input ke dalam fungsi. Bila input yang masuk adalah bilangan dengan nilai yang lebih kecil daripada jawaban yang telah ditetapkan, maka akan muncul sebuah tampilan berupa “lebih besar”. Sedangkan bila input yang masuk adalah bilangan dengan nilai yang lebih besar, maka akan muncul tampilan string berupa “lebih kecil”. Setiap kesalahan user dalam menebak

angka (jawaban) akan memberikan sebuah sangsi kecil berupa pengurangan skor sebanyak -5 (minus lima) point. User akan dianggap kalah jika skor akhir dari permainan < 0 point.

4.15 Game Diner Dash

File Diner Dash ini berisi library `stdio.h`, `stdlib.h`, `time.h`, file `boolean.h`, dan `dinerdash.h`. Terdapat tiga buah tipe data tambahan. Pertama, tipe data Pesanan (berisi Makanan, DurasiMasak, Ketahanan, Harga). Kedua, tipe QueuePesanan (berisi buffer, IdxHead, dan IdxTail). Dan terakhir, tipe ArrayPesanan (berisi buffer, Capacity, Neff).

Terdapat 16 prosedur dan 6 fungsi pada `dinerdash.c` ini. Prosedur - prosedur yang dimaksud adalah `CreateArrayPesanan`, `DeallocateArrayPesanan`, `InsertAPesananAt`, `DeleteAPesananAt`, `CreateQueuePesanan`, `TabelPesanan`, `TabelMasakan`, `TabelSajian`, `Inisialisasi`, `enqueuePesanan`, `Memasak`, `Ketahanan`, `dequeuePesanan`, `Cook`, `Serve`, dan `dinerDASH`. Sedangkan, untuk fungsinya itu sendiri ada `isEmptyAPesanan`, `SearchArrayPesanan`, `isEmptyQPesanan`, `MakeRandomPesanan`, `QueuePesananLength`, dan `GetIdx`.

`CreateQueuePesanan` adalah sebuah prosedur satu variabel input dengan tipe pointer to `QueuePesanan`. Prosedur ini dibuat dengan output akhir yang diminta adalah sebuah `QueuePesanan` dengan `IdxHead` dan `IdxTail` yang bernilai `IDX_UNDEF`.

`CreateArrayPesanan` adalah sebuah prosedur satu variabel input dengan tipe pointer to `ArrayPesanan`. Prosedur ini dibuat dengan output akhir yang diminta adalah sebuah `ArrayPesanan` dengan buffer dinamis, `Capacity` yang bernilai kapasitas, dan `Neff` bernilai 0.

`isEmptyAPesanan` merupakan sebuah fungsi satu variabel input dengan tipe `ArrayPesanan`. Fungsi ini memberikan return boolean dari `array.Neff` yang bernilai 0. Jika `array.Neff = 0`, maka fungsi ini akan memberikan return true, begitu juga sebaliknya.

`DeallocateArrayPesanan` adalah sebuah prosedur satu variabel input dengan tipe pointer to `ArrayPesanan`. Prosedur ini dibuat dengan output akhir yang diminta adalah sebuah array yang dibebaskan, nilai `array.Neff = 0`, dan `array.Capacity = 0`.

`InsertAPesananAt` adalah sebuah prosedur tiga variabel input dengan tipe pointer to `ArrayPesanan`, `Pesanan`, dan integer. Initial State yang diterima hanya array dengan jumlah `Neff` sama dengan `Capacity`. Final State yang diberikan dari prosedur ini adalah sebuah array dengan pesanan yang telah terselip di urutan yang diminta (berdasarkan input variabel bertipe integer) dan jumlah `Array.Neff` yang bertambah 1.

`DeleteAPesananAt` adalah sebuah prosedur tiga variabel input dengan tipe pointer to `ArrayPesanan`, `Pesanan`, dan integer. Initial State yang diterima pada prosedur ini adalah array yang tidak kosong. Adapun untuk Final State yang akan dikeluarkan dari prosedur ini adalah sebuah array dengan `Neff-1` dan memiliki buffer yang dihapus pada urutan yang telah ditentukan.

isEmptyQPesanan adalah sebuah fungsi satu variabel input dengan tipe QueuePesanan. Fungsi ini akan mengembalikan sebuah boolean dari IdxHead dan IdxTail yang bernilai IDX_UNDEF.

TabelPesanan adalah sebuah prosedur satu variabel input dengan tipe QueuePesanan. Initial State yang diterima untuk prosedur ini adalah sebuah QueuePesanan yang kosong maupun berisi. Final State yang diberikan adalah sebuah printf dengan bentukan tabel yang telah disesuaikan dengan kebutuhan spek.

TabelMasakan adalah sebuah prosedur satu variabel input dengan tipe ArrayPesanan. Initial State yang diterima untuk prosedur ini adalah sebuah ArrayPesanan yang kosong maupun berisi. Final State yang diberikan adalah sebuah printf dengan bentukan tabel yang telah disesuaikan dengan kebutuhan spek.

TabelSajian adalah sebuah prosedur satu variabel input dengan tipe ArrayPesanan. Initial State yang diterima untuk prosedur ini adalah sebuah ArrayPesanan yang kosong maupun berisi. Final State yang diberikan adalah sebuah printf dengan bentukan tabel yang telah disesuaikan dengan kebutuhan spek.

RandomPesanan adalah sebuah fungsi satu variabel input dengan tipe data integer. Fungsi ini membagi case pengacakan nomor berdasarkan urutan antrian (antrian dibawah 10 atau tidak). Output akhir yang dihasilkan dari fungsi ini adalah sebuah Pesanan dengan Makanan, DurasiMasak, Ketahanan, dan Harga yang telah diacak dengan batasan angka - angka tertentu (Durasi bernilai integer dari 1 sampai 5, Ketahanan bernilai integer dari 1 sampai 5, dan Harga bernilai 10.000 sampai 50.000).

Inisialisasi adalah sebuah prosedur satu variabel input dengan tipe pointer QueuePesanan. Initial State yang diterima adalah sebuah QueuePesanan kosong ataupun berisi. Final State yang diberikan adalah tiga buah Pesanan acak dengan Makanan, DurasiMasak, Ketahanan, dan Harga yang telah ditetapkan. Tiga buah Pesanan acak yang telah dibuat tadi, selanjutnya akan dimasukkan ke QueuePesanan.

enqueuePesanan adalah sebuah prosedur dua variabel input dengan tipe pointer to QueuePesanan dan Pesanan. Initial State yang diterima untuk prosedur ini adalah sebuah QueuePesanan yang kosong maupun berisi. Final State yang diberikan adalah sebuah QueuePesanan dengan Pesanan yang telah dimasukan ke dalam QueuePesanan dengan Pesanan sebagai IdxTail.

dequeuePesanan adalah sebuah prosedur dua variabel input dengan tipe pointer to QueuePesanan dan pointer to Pesanan. Initial State yang diterima untuk prosedur ini adalah hanya sebuah ArrayPesanan yang berisi (entah itu berisi satu ataupun lebih dari satu). Final State yang diberikan adalah sebuah QueuePesanan dengan Pesanan yang telah dihapus dari QueuePesanan. Untuk kondisi QueuePesanan setelah prosedur ini berlangsung adalah QueuePesanan yang kosong atau berisi.

GetIdx adalah sebuah fungsi dua variabel input dengan tipe data QueuePesanan dan Kata. Fungsi ini akan mengecek terlebih dahulu apakah input yang dimasukan apakah Pesanan.Makanan merupakan sebuah Pesanan yang berada di dalam QueuePesanan atau tidak. Jika Pesanan tidak ada di dalam QueuePesanan tidak ada, maka akan diberikan return integer berupa IDX_UNDEF. Namun, apabila Pesanan berada di dalam QueuePesanan, maka return yang akan diberikan adalah ASCII dari urutan Pesanan.Makanan.

Cook adalah sebuah prosedur empat variabel input dengan tipe pointer to ArrayPesanan, QueuePesanan, Kata, dan boolean. Initial State yang diterima untuk prosedur ini adalah Makanan dengan index yang tidak IDX_UNDEF. Adapun untuk Final State yang diberikan dari prosedur ini adalah printf kondisi masakan yang berhasil dimasukkan kedalam ArrayPesanan ataupun tidak.

Serve adalah sebuah prosedur lima variabel input dengan tipe pointer to ArrayPesanan, pointer to QueuePesanan, Kata, dan pointer to integer, dan pointer to boolean. Initial State yang diterima untuk prosedur ini adalah Makanan dengan index yang tidak IDX_UNDEF. Adapun untuk Final State yang diberikan dari prosedur ini adalah printf kondisi masakan yang berhasil didelete dari ArrayPesanan ataupun tidak dan kondisi saldo yang bertambah ataupun tidak.

SearchArrayPesanan adalah sebuah fungsi dua variabel input dengan tipe data ArrayPesanan dan Kata. Input yang diterima pada fungsi terhitung bisa apa saja (selama kita tetap memasukkan ArrayPesanan dan Kata). Untuk return dari fungsi ini adalah sebuah integer berupa indeks dari ArrayPesanan yang buffernya sesuai dengan Kata yang diberikan .

Prosedur Ketahanan dan Memasak sama sama memberikan hasil akhir berupa decrement durasi dari ketahanan dan memasak itu sendiri

Adapun untuk prosedur dinerDASH itu sendiri merupakan sebuah prosedur utama yang memanggil semua prosedur dan fungsi penyusun lainnya. Tampilan yang akan muncul pada prosedur ini adalah tabel pesanan, tabel sajian, tabel masakan, saldo, komando dari user, dan respon dari komando yang diberikan.

5 Data Test

5.1 START

Command START akan membaca file konfigurasi (config.txt) dan dimasukkan ke dalam array listgame. Command START hanya dapat dilakukan sekali ketika user belum berhasil melakukan START atau LOAD <filename>. Secara default, command START pasti berhasil dilakukan karena file config.txt pasti berada di dalam folder data. Namun, jika terjadi testcase dimana config.txt tidak terdapat pada folder data, maka command START gagal dilakukan.

```
ENTER COMMAND: START
BMO: Mengenali perintah...
KONFIGURASI AWAL BERHASIL!
```

Gambar 5.1

Jika user sudah melakukan START / LOAD <filename> namun ingin melakukan START kembali, maka program akan mengirimkan pesan sebagai berikut :

```
ENTER COMMAND: START
BMO: Mengenali perintah...
BMO: Sistem sudah terkonfigurasi.
** Hint: Untuk memulai konfigurasi baru, silahkan "QUIT" terlebih dahulu **
```

Gambar 5.2

5.2 LOAD <namafile>

Command LOAD <filename> akan membaca file <filename> dan dimasukkan ke dalam array listgame. Command LOAD <filename> hanya dapat dilakukan sekali ketika user belum berhasil melakukan START atau LOAD <filename>. Jika <filename> yang dimasukkan tidak terdapat dalam folder data, maka LOAD gagal dilakukan.

```
ENTER COMMAND: LOAD save.txt
BMO: Mengenali perintah...
FILE save.txt TIDAK DITEMUKAN!
```

Gambar 5.3

Sebaliknya jika <filename> yang dimasukkan terdapat pada folder data, maka LOAD berhasil dilakukan.

```

ENTER COMMAND: LOAD save1.txt
BMO: Mengenali perintah...
FILE save1.txt BERHASIL DIMUAT!

```

Gambar 5.4

Jika user sudah melakukan START / LOAD <filename> namun ingin melakukan LOAD <filename> kembali, maka program akan mengirimkan pesan sebagai berikut :

```

ENTER COMMAND: LOAD save1.txt
BMO: Mengenali perintah...
BMO: Sistem sudah terkonfigurasi.
** Hint: Untuk memulai konfigurasi baru, silahkan "QUIT" terlebih dahulu **

```

Gambar 5.5

5.3 CREATE GAME

Command CREATE GAME akan menambahkan game baru dengan nama yang didapatkan melalui input pengguna pada program BNMO, dan akan disimpan ke dalam listgame.

```

ENTER COMMAND: CREATE GAME
BMO: Mengenali perintah...
Masukkan nama game yang akan ditambahkan: BINOMO
Game berhasil ditambahkan

```

Gambar 5.6

5.4 LIST GAME

Command LIST GAME akan menampilkan daftar game yang dimiliki oleh sistem, termasuk dengan game baru yang ditambah melalui command create game (jika ada).

```

ENTER COMMAND: LIST GAME
BMO: Mengenali perintah...
Berikut adalah daftar game yang tersedia
1. RNG
2. Diner DASH
3. DINOSAUR IN EARTH
4. RISEWOMAN
5. EIFFEL TOWER
6. MARVEL SNAP
7. BINOMO

```

Gambar 5.7

Penjelasan: MARVEL SNAP merupakan nama game bonus dan pada test case sebelumnya telah ditambahkan game BIMONO

5.5 DELETE GAME

Command DELETE GAME akan menghapus game sesuai nomor yang dimasukkan pengguna. Game yang bisa dihapus hanya game yang baru ditambahkan melalui command *CREATE GAME*.

```
ENTER COMMAND: DELETE GAME
BMO: Mengenali perintah...
Berikut adalah daftar game yang tersedia
1. RNG
2. Diner DASH
3. DINOSAUR IN EARTH
4. RISEWOMAN
5. EIFFEL TOWER
6. MARVEL SNAP
7. BINOMO

** Hint: Game yang dapat dihapus adalah Game dengan nomor urut > 6 **
Masukkan nomor game yang akan dihapus: 7

Game Berhasil Dihapus!
```

Gambar 5.8

Sementara itu, game seperti RNG, Diner Dash, DINOSAUR IN EARTH, RISEWOMAN, EIFFEL TOWER, dan MARVEL SNAP merupakan game yang tidak dapat dihapus.

```
ENTER COMMAND: DELETE GAME
BMO: Mengenali perintah...
Berikut adalah daftar game yang tersedia
1. RNG
2. Diner DASH
3. DINOSAUR IN EARTH
4. RISEWOMAN
5. EIFFEL TOWER
6. MARVEL SNAP

** Hint: Game yang dapat dihapus adalah Game dengan nomor urut > 6 **
Masukkan nomor game yang akan dihapus: 6

Gagal Menghapus Game: Game terdapat dalam konfigurasi awal!
```

Gambar 5.9

Adapun kondisi jika game yang ingin dihapus masuk ke dalam *queueGame*, nomor urut yang dimasukkan lebih besar dari jumlah game yang dimiliki pengguna, atau input bukan angka maka Game akan gagal dihapus.

```

ENTER COMMAND: DELETE GAME
BMO: Mengenali perintah...
Berikut adalah daftar game yang tersedia
1. RNG
2. Diner DASH
3. DINOSAUR IN EARTH
4. RISEWOMAN
5. EIFFEL TOWER
6. MARVEL SNAP
7. BISMILLAH IP 4

** Hint: Game yang dapat dihapus adalah Game dengan nomor urut > 6 **
Masukkan nomor game yang akan dihapus: 7

Gagal Menghapus Game: Game terdapat dalam antrian!

```

Gambar 5.10

```

ENTER COMMAND: DELETE GAME
BMO: Mengenali perintah...
Berikut adalah daftar game yang tersedia
1. RNG
2. Diner DASH
3. DINOSAUR IN EARTH
4. RISEWOMAN
5. EIFFEL TOWER
6. MARVEL SNAP
7. BISMILLAH IP 4

** Hint: Game yang dapat dihapus adalah Game dengan nomor urut > 6 **
Masukkan nomor game yang akan dihapus: sjdkal
Mohon masukkan masukan yang valid!

```

Gambar 5.11

```

ENTER COMMAND: DELETE GAME
BMO: Mengenali perintah...
Berikut adalah daftar game yang tersedia
1. RNG
2. Diner DASH
3. DINOSAUR IN EARTH
4. RISEWOMAN
5. EIFFEL TOWER
6. MARVEL SNAP

** Hint: Game yang dapat dihapus adalah Game dengan nomor urut > 6 **
Masukkan nomor game yang akan dihapus: 10

Gagal Menghapus Game: Masukan berada di luar rentang!

```

Gambar 5.12

5.6 QUEUE GAME

Command QUEUE GAME akan menampilkan daftar antrian game yang dimiliki user serta daftar game yang dimiliki oleh user. Command ini akan meminta input nomor game dari daftar game yang ingin dimasukkan ke antrian game. Jika input valid (nomor game \leq banyak game yang dimiliki user), maka game akan dimasukkan kedalam daftar antrian game.

```
ENTER COMMAND: QUEUE GAME
BMO: Mengenali perintah...
Antrian game Anda kosong.
Berikut adalah daftar game yang tersedia
1. RNG
2. Diner DASH
3. DINOSAUR IN EARTH
4. RISEWOMAN
5. EIFFEL TOWER
6. MARVEL SNAP
Nomor Game yang mau ditambahkan ke antrian : 1
Game berhasil ditambahkan ke dalam daftar antrian.
```

Gambar 5.13

Adapun input yang dianggap tidak valid seperti nomor game yang dimasukkan lebih besar dari banyak game yang dimiliki *user* atau input bukan angka, maka program akan mengirimkan pesan sebagai berikut.

```
ENTER COMMAND: QUEUE GAME
BMO: Mengenali perintah...
Berikut adalah antrian game Anda :
1. RNG

Berikut adalah daftar game yang tersedia
1. RNG
2. Diner DASH
3. DINOSAUR IN EARTH
4. RISEWOMAN
5. EIFFEL TOWER
6. MARVEL SNAP
Nomor Game yang mau ditambahkan ke antrian : jakjl
Mohon masukkan masukan yang valid!
```

Gambar 5.14

```
ENTER COMMAND: QUEUE GAME
BMO: Mengenali perintah...
Berikut adalah antrian game Anda :
1. RNG

Berikut adalah daftar game yang tersedia
1. RNG
2. Diner DASH
3. DINOSAUR IN EARTH
4. RISEWOMAN
5. EIFFEL TOWER
6. MARVEL SNAP
Nomor Game yang mau ditambahkan ke antrian : 10
Masukan berada di luar rentang "LIST GAME"
Penambahan Game pada daftar antrian dibatalkan.
```

Gambar 5.15

5.7 PLAY GAME

Command Play Game akan memainkan game yang paling pertama ada di antrian (antrianatas), jika user memainkan game yang berada pada konfigurasi awal, maka yang dapatdimainkan hanya RNG, Diner DASH, dan MARVEL SNAP. Sedangkan untuk gamekonfigurasi lainnya, akan mengeluarkan pesan “game sedang dalam maintenance”.

```
ENTER COMMAND: PLAY GAME
BMO: Mengenali perintah...
Berikut adalah antrian game Anda :
1. RNG
Loading RNG...
```

Gambar 5.16

```
ENTER COMMAND: PLAY GAME
BMO: Mengenali perintah...
Berikut adalah antrian game Anda :
1. DINOSAUR IN EARTH
Loading DINOSAUR IN EARTH...
DINOSAUR IN EARTH masih dalam maintenance.
Silahkan pilih game lainnya!
```

Gambar 5.17

Adapun case jika game yang dimainkan merupakan game buatan user, maka permainan akan langsung berakhir dengan skor random.

```
ENTER COMMAND: PLAY GAME
BMO: Mengenali perintah...
Berikut adalah antrian game Anda :
1. BISMILLAH IP 4
Loading BISMILLAH IP 4...
[ GAME OVER ]
[ SCORE: 6334 ]
```

Gambar 5.18

Selain itu, untuk case antrian kosong, program akan mengeluarkan pesan “Antrian game Anda kosong.”

```

ENTER COMMAND: PLAY GAME
BMO: Mengenali perintah...
Antrian game Anda kosong.
BMO: Antrian sama Dompot kamu kok mirip yaa? ^^
** Hint: Tambahkan Game dalam antrian menggunakan "QUEUE GAME" **

```

Gambar 5.19

5.8 SKIPGAME <n>

Command SKIPGAME <n> akan terlebih dahulu mendequeue antrian sebanyak <n>, lalu program akan memainkan game dengan antrian paling pertama. jika user memainkan game yang berada pada konfigurasi awal, maka yang dapat dimainkan hanya RNG, Diner DASH, dan MARVEL SNAP. Sedangkan untuk game konfigurasi lainnya, akan mengeluarkan pesan “game sedang dalam maintenance”.

```

ENTER COMMAND: SKIP GAME 1
BMO: Mengenali perintah...
Berikut adalah antrian game Anda :
1. Diner DASH
2. RNG

Skipping 1 games...
Loading RNG..

```

Gambar 5.20

```

ENTER COMMAND: SKIP GAME 1
BMO: Mengenali perintah...
Berikut adalah antrian game Anda :
1. RNG
2. RISEWOMAN

Skipping 1 games...
Loading RISEWOMAN...
RISEWOMAN masih dalam maintenance.
Silahkan pilih game lainnya!

```

Gambar 5.21

Adapun case jika game yang dimainkan merupakan game buatan user, maka permainan akan langsung berakhir dengan skor random.

```

ENTER COMMAND: SKIP GAME 1
BMO: Mengenali perintah...
Berikut adalah antrian game Anda :
1. RNG
2. BISMILLAH IP 4

Skipping 1 games...
Loading BISMILLAH IP 4...
[ GAME OVER ]
[ SCORE: 26500 ]

```

Gambar 5.22

Adapun test case jika setelah antrian didequeue, antrian game menjadi kosong, maka program akan mengeluarkan pesan “antrian kosong”, dan jika <n> bukan angka maka input akan dianggap invalid.

```

ENTER COMMAND: SKIP GAME 1
BMO: Mengenali perintah...
Berikut adalah antrian game Anda :
1. Diner DASH

Skipping 1 games...
BMO: Antrian sama Dompot kamu kok mirip yaa? ^^
** Hint: Tambahkan Game dalam antrian menggunakan "QUEUE GAME" **

```

Gambar 5.23

```

ENTER COMMAND: SKIP GAME 1
BMO: Mengenali perintah...
Antrian game Anda kosong.

```

Gambar 5.24

```

ENTER COMMAND: SKIP GAME jddklasjdlak
BMO: Mengenali perintah...
Mohon masukkan dengan format yang benar!
** Hint: Ketik "HELP" untuk melihat perintah **

```

Gambar 5.25

5.9 HELP

Command `help` akan menampilkan daftar *command* dan penjelasan singkat untuk mempermudah pengguna untuk memahami fungsi dari setiap *command*. Berikut merupakan tampilan pemanggilan fungsi *HELP* ketika *START* atau *LOAD* belum berhasil.

```
ENTER COMMAND: HELP
BMO: Mengenali perintah...
DAFTAR COMMAND:
1. START - Melakukan konfigurasi sistem baru
2. LOAD <namafile.txt> - Memuat konfigurasi pada SAVEFILE yang dipilih
3. HELP - Menampilkan daftar command (perintah) yang dapat dipanggil
4. QUIT - Mengakhiri dan keluar dari sistem
```

Gambar 5.26

Sementara itu, tampilan pemanggilan fungsi *HELP* ketika *START* atau *LOAD* sudah berhasil dieksekusi ada pada gambar berikut.

```
ENTER COMMAND: HELP
BMO: Mengenali perintah...
DAFTAR COMMAND:
1. START - Melakukan konfigurasi sistem baru
2. LOAD <namafile.txt> - Memuat konfigurasi pada SAVEFILE yang dipilih
3. HELP - Menampilkan daftar command (perintah) yang dapat dipanggil
4. QUIT - Mengakhiri dan keluar dari sistem
5. SAVE <namafile.txt> - Melakukan penyimpanan konfigurasi pada file tertentu
6. CREATE GAME - Membuat dan menambahkan game baru ke dalam daftar
7. LIST GAME - Menampilkan daftar game yang dapat dimainkan
8. DELETE GAME - Menghapus suatu game dari dalam daftar
9. QUEUE GAME - Menambahkan game tertentu ke dalam antrian permainan
10. PLAY GAME - Memulai permainan berdasarkan antrian teratas
11. SKIP GAME <n> - Melewati n banyak game dari dalam antrian dan memainkan game berikutnya.
```

Gambar 5.27

5.10 SAVE <namafile>

Command `SAVE <namafile>` menyimpan *listgame* ke dalam sebuah file <namafile>. Berikut tampilan pemanggilan *SAVE* pada kasus *string kosong*, *filename valid*, dan *filename invalid*.

```
ENTER COMMAND: SAVE
BMO: Mengenali perintah...
Mohon masukkan dengan format yang benar!
** Hint: Ketik "HELP" untuk melihat perintah **

ENTER COMMAND: SAVE save1.txt
BMO: Mengenali perintah...

-----
Berhasil menyimpan progress. ^^
-----

ENTER COMMAND: SAVE save1.txt\
BMO: Mengenali perintah...

-----
Gagal membuka / membuat file. Silahkan coba lagi!
-----
```

Gambar 5.28

5.11 QUIT

Command *QUIT* merupakan command yang dipanggil ketika pengguna ingin keluar dari program. Pemanggilan command ini dipengaruhi oleh keadaan konfigurasi BNMO. Berikut tampilan pemanggilan *QUIT* bila program sudah terkonfigurasi.

Jika user ingin menyimpan:

```
ENTER COMMAND: QUIT
BMO: Mengenali perintah...
BMO: Apakah anda ingin menyimpan progress anda?
** Hint: Ketika 'Ya' jika ingin menyimpan atau 'Tidak' jika ingin membuang **
You: Ya

BMO: Masukkan nama save file anda!
Nama file: save1.txt

-----
Berhasil menyimpan progress. ^^
-----

BMO: BMO sangat senang bisa bermain denganmu! ^^
BMO: Let's play again sometimes, okay? ^^
BMO: Battery low. Shutdown...
```

Gambar 5.29

Jika user tidak ingin menyimpan:

```
ENTER COMMAND: QUIT
BMO: Mengenali perintah...
BMO: Apakah anda ingin menyimpan progress anda?
** Hint: Ketika 'Ya' jika ingin menyimpan atau 'Tidak' jika ingin membuang **
You: Tidak

BMO: Progress anda tidak disimpan.

BMO: BMO sangat senang bisa bermain denganmu! ^^
BMO: Let's play again sometimes, okay? ^^
BMO: Battery low. Shutdown...
```

Gambar 5.30

Berikut tampilan pemanggilan *QUIT* bila program belum terkonfigurasi.

```
ENTER COMMAND: QUIT
BMO: Mengenali perintah...

BMO: BMO sangat senang bisa bermain denganmu! ^^
BMO: Let's play again sometimes, okay? ^^
BMO: Battery low. Shutdown...
```

Gambar 5.31

6 Test Script

No.	Fitur yang Dites	Tujuan Testing	Langkah-Langkah Testing	Input Data Test	Hasil yang Diharapkan	Hasil yang Keluar
1	START	Untuk memeriksa berjalannya program dengan baik setelah pemanggilan fungsi.	START	Dapat dilihat pada Data Test 5.1	Dapat dilihat pada Gambar 5.1 dan Gambar 5.2	Dapat dilihat pada Gambar 5.1 dan Gambar 5.2
2	LOAD	Untuk mengetahui apakah ketika command load dipanggil, nama file yang di-load akan terbaca.	LOAD	Dapat dilihat pada Data Test 5.2	Dapat dilihat pada Gambar 5.3, Gambar 5.4, Gambar 5.5	Dapat dilihat pada Gambar 5.3, Gambar 5.4, Gambar 5.5
3	CREATE GAME	Untuk menambahkan game baru ke dalam daftar game	CREATE GAME Masukkan nama game yang ingin ditambahkan	Dapat dilihat pada Data Test 5.3	Dapat dilihat pada Gambar 5.6	Dapat dilihat pada Gambar 5.6
4	LIST GAME	Untuk memeriksa apakah data game yang dimiliki terdaftar secara lengkap dan terbaru.	LIST GAME	Dapat dilihat pada Data Test 5.4	Dapat dilihat pada Gambar 5.7	Dapat dilihat pada Gambar 5.7
5	DELETE GAME	Untuk memeriksa apakah program dapat menghapus game yang bukan game konfigurasi awal, dan apakah program mengenali game konfigurasi awal.	DELETE GAME Masukkan nomor game yang akan dihapus	Dapat dilihat pada Data Test 5.5	Dapat dilihat pada Gambar 5.8 , Gambar 5.9, Gambar 5.10, Gambar 5.11, Gambar 5.12	Dapat dilihat pada Gambar 5.8 , Gambar 5.9, Gambar 5.10, Gambar 5.11, Gambar 5.12
6	QUEUEGAME	Untuk memastikan daftar antrian game sesuai dengan keinginan pengguna.	QUEUE GAME Masukkan nomor game yang ingin ditambah ke antrian	Dapat dilihat pada Data Test 5.6	Dapat dilihat pada Gambar 5.13, Gambar 5.14 dan Gambar 5.15	Dapat dilihat pada Gambar 5.13, Gambar 5.14 dan Gambar 5.15
7	PLAYGAME	Untuk memainkan permainan.	PLAY GAME	Dapat dilihat	Dapat dilihat pada Gambar 5.16 , Gambar	Dapat dilihat pada Gambar 5.16 , Gambar

		Game yang dipilih adalah permainan dengan urutan tertatas di antrian game.		pada Data Test 5.7	5.17, Gambar 5.18, Gambar 5.19	5.17, Gambar 5.18, Gambar 5.19
8	SKIP GAME <n>	Untuk memeriksa apakah program dapat melewati game sesuai permintaan pengguna.	SKIP GAME <n>	Dapat dilihat pada Data Test 5.8	Dapat dilihat pada Gambar 5.20, Gambar 5.21, Gambar 5.22, Gambar 5.23, Gambar 5.24, Gambar 5.25	Dapat dilihat pada Gambar 5.20, Gambar 5.21, Gambar 5.22, Gambar 5.23, Gambar 5.24, Gambar 5.25
9	HELP	Untuk memberikan bantuan command	HELP	Dapat dilihat pada Data Test 5.9	Dapat dilihat pada Gambar 5.26 dan Gambar 5.27	Dapat dilihat pada Gambar 5.26 dan Gambar 5.27
10	SAVE	Untuk memeriksa apakah program dapat menyimpan file.	SAVE <namafile>	Dapat dilihat pada Data Test 5.10	Dapat dilihat pada Gambar 5.28	Dapat dilihat pada Gambar 5.28
11	QUIT	Untuk memeriksa apakah pengguna dapat keluar dari program.	QUIT	Dapat dilihat pada Data Test 5.11	Dapat dilihat pada Gambar 5.29, Gambar 5.30, Gambar 5.31	Dapat dilihat pada Gambar 5.29, Gambar 5.30, Gambar 5.31

7 Pembagian Kerja dalam Kelompok

No	Fitur/ADT/Fungsi	NIM Coder	NIM Tester
1	ADT Array	18221047, 18221171	18221047, 18221171
2	ADT Mesin Karakter	18221047, 18221171	18221047, 18221171
3	ADT Mesin Kata	18221047, 18221171	18221047, 18221171
4	ADT Queue	18221053	18221053, 18221171
5	Main	18221171	18221171
6	Console	18221047, 18221053, 18221137, 18221167, 18221171	18221047, 18221053, 18221137, 18221167, 18221171
7	Fungsi Program	18221053, 18221137, 18221167	18221053, 18221137, 18221167, 18221171

NIM	Nama	Tugas
18221047	I Dewa Made Manu Pradnyana	<ol style="list-style-type: none"> 1. Mengerjakan fungsi START 2. Mengerjakan fungsi LOAD 3. Membantu mengerjakan fungsi Diner Dash 4. Membantu mengerjakan RNG 5. Mengerjakan Game Bonus
18221053	Laurentia Kayleen Christopher	<ol style="list-style-type: none"> 1. Mengerjakan fungsi QUEUEGAME 2. Mengerjakan fungsi PLAYGAME 3. Mengerjakan fungsi SKIPGAME 4. Mengerjakan laporan bagian fungsi
18221137	Felisa Aidadora Darmawan	<ol style="list-style-type: none"> 1. Mengerjakan fungsi CREATEGAME 2. Mengerjakan fungsi LISTGAME 3. Mengerjakan fungsi DELETGAME 4. Mengerjakan MoM asistensi
18221167	Ananda Abdul Hafizh	<ol style="list-style-type: none"> 1. Mengerjakan fungsi Diner Dash 2. Mengerjakan fungsi RNG 3. Mengerjakan driver array

18221171	Hans Stephano Edbert N	<ol style="list-style-type: none"> 1. Mengerjakan fungsi SAVE 2. Mengerjakan fungsi QUIT 3. Mengerjakan fungsi HELP 4. Mengerjakan MAIN
----------	------------------------	---

8 Lampiran

8.1 Deskripsi Tugas Besar 1

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


8.2 Notulen Rapat

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





No. Kelompok/Kelas : Kelompok-04/K-01
Nama Kelompok : KakCRBaikHatiDanTidakSombong
Anggota Kelompok (Nama/NIM) :
1. I Dewa Made Manu Pradnyana / 18221047
2. Laurentia Kayleen Christopher / 18221053
3. Felisa Aidadora Darmawan / 18221137
4. Ananda Abdul Hafizh / 18221167
5. Hans Stephano Edbert N / 18221171


Asisten Pembimbing : Cynthia Rusadi

Asistensi I





Tanggal : 4 November 2022	Catatan Asistensi: Kak Cynthia : command queuegame pertama dimasukin sesuai angka 1-5. sbg saran dikasih set id. nanti input pakai mesin kata, nanti baru dibaca dan disesuaikan sama id nya. karena di tiap game uda di set id nya berapa nanti dicari tau ini game yang mana. sbg saran juga buat adt untuk nentuin id sama nama game. nanti udah tau nama game nya apa aja, nanti dimasukin ke queuegame. Yang dimasukin itu terserah, boleh id game, nama game, atau dimasukin adt dari game nya sekalian juga bole. Kayleen : bedanya daftar antrian game-mu dan daftar antrian yang tersedia ? - Kak Cynthia : itu ada daftar game yang bisa dimainin. Di spek ada 2 game yang uda pasti.
Tempat : https://itb-ac-id.zoom.us/j/95191254003?pwd=UzByb2sxd1JabmVUQVYyWXM0dExzQT09	
Kehadiran Anggota Kelompok: 1 18221047 I Dewa Made Manu Pradnyana 	



STEI- ITB	IF2111-TB1-04	Halaman 31 dari 35 halaman
Template dokumen ini dan informasi yang dimilikinya adalah milik Sekolah Teknik Elektro dan Informatika ITB dan bersifat rahasia. Dilarang me-reproduksi dokumen ini tanpa diketahui oleh Sekolah Teknik Elektro dan Informatika ITB.		

<p>2 18221053 Laurentia Kayleen Christopher</p>  <p>3 18221137 Felisa Aidadora Darmawan</p>  <p>4 18221167 Ananda Abdul Hafizh</p>  <p>5 18221171 Hans Stephano Edbert N</p> 	<p>Daftar game yang tersedia itu game apa aja yang tersedia dan bisa dimainkan berkali-kali. Hans : di file configuration awalnya pake 2 game fix ? - Kak Cynthia : iya, kalo belum ditambahin game sendiri. Kak Cynthia : untuk scanf dan fgets di dalam mesin kata boleh. Kayleen : untuk playgame, untuk enter command playgame pasti mainin daftar pertama di antrian game ya ka ? untuk skipgame, kalau ada 3 game tapi skip 2 berarti kita terima input nilai 2 itu ? di playgame kalau udah mainin game pertama, berarti game tersebut hilang dari daftar game untuk dimainkan ya Kak Cynthia , kecuali ada 2x di daftar ? - Kak Cynthia : iya - Kak Cynthia : Iya tapi gaboleh pakai scanf. Untuk skipgame kalau n lebih dari jumlah game di daftar, daftar nya jadi kosong. - Kak Cynthia : iya bener dia hilang Apis : untuk diner dash dia pasti baca file ya ? - Kak Cynthia : itu udah ga ngebaca file lagi, itu pakai input dari user tanpa scanf. Hafizh : dalam diner dash ada memasak dan serve, bisa ga Kak Cynthia kita cook makanan 2x sekaligus ? Misalnya M0 nya 2x. Ada batas maksimal gosong gak ? Kalau gosong harus masak dari awal, ketahanan dan durasi pakai RNG lagi atau enggak ? - Kak Cynthia : karena pakai ID makanan, itu increment jadi m0 1 m0 2 m0 3 jadi ga ada yang sama. Untuk durasi dan ketahanan pakai RNG. Kadek : pas jalanin BNMO di awal user nya harus command start atau load dulu atau bisa yang lain ? - Kak Cynthia : iya, jadi kalau command yang lain berarti ga valid input nya. Hans : Start dan load harus dipanggil dulu di awal. Kalau diawal ditambahin command exit, cmdlist boleh ga ? - Kak Cynthia : boleh Hans : Berarti gaperlu sama persis kayak spek, boleh dimodif dikit ? - Kak Cynthia : boleh aja selama tujuannya masi sama. Hans : Ketika loading perlu dibuat delay ga ? - Kak Cynthia : gaperlu Hans : di dalam file configuration isinya sekarang Cuma jumlah game sama game yang dibuat pakai creategame ya ? - Kak Cynthia : iya</p> <p>Tanda Tangan Asisten:</p>
--	---

	
--	---

Asistensi II

Tanggal : 11 November 2022	Catatan Asistensi:
Tempat : https://itb-ac-id.zoom.us/j/91690822741?pwd=WGdLeFRtdGN0dis2cWRkaDEyank4QT09	Hans : udah masukin ke main pake binomain nama filenya, ada satu bug yang udah lama tp gabisa ketemu, tentang save dan delete kadang-kadang error kadang-kadang engga. Kalo gamenya cmn 5 gabisa save gitu error.
Kehadiran Anggota Kelompok: <div style="text-align: center;"> 1 18221047 I Dewa Made Manu Pradnyana  2 18221053 Laurentia Kayleen Christopher  3 18221137 Felisa Aidadora Darmawan  4 18221167 Ananda Abdul Hafizh  </div>	Kak Cynthia : mungkin implementasi di savenya ada yang beda, tp yang penting skrg udah jalan jd gpp Kak Cynthia : perlu bikin driver buat masing-masing adt dasar Kayleen : Algoritma menarik di laporan perlu ngga sih Kak? Kak Cynthia : optional aja sih Kayleen : mengurangi nilai ga kak kalo ngga ada? Kak Cynthia : engga kok Kayleen : di laporan data test, perlu masukin apa aja sesuai deskripsi aja, untuk fungsi start inputnya gimana, expected output gimana Kadek : kalo misalnya mau delete game atau queue game, inputnya ngga valid, harus di while sampe inputnya valid atau terserah kita aja? Kak Cynthia : terserah kalian Kadek : di fungsi perlu tambahkan komen ga Kak Cynthia : gaperlu yang spesifik banget yang penting jelasin fungsinya ngapain Kadek : termasuk kriteria penilaian ga Kak? Kak Cynthia : iya Kayleen : ringkasan perlu panjang atau ringkas aja? Kak Cynthia : gausah panjang-panjang aja, deskripsi sama persoalan, laporan ini tentang apa, kesimpulan tugas besar apa. Kak Cynthia : pengumpulan di olympia, salah satu aja yang kumpulin Kadek : kalo ada yang error setelah kumpulin bisa resubmit ga Kak Cynthia : kasih tau aja ke aku, paling nanti anggota lainnya yang submit soalnya satu orang cuman bisa submit sekali

<p>5 18221171 Hans Stephano Edbert N</p> 	
<p>Tanda Tangan Asisten:</p> 	<p>Dokumentasi :</p>

8.3 Log Activity Anggota Kelompok

Timeline Kelompok

No	Waktu	Keterangan
1.	Minggu, 30 Oktober 2022	Diskusi pembagian kerja dalam kelompok dan metode pengerjaan secara kelompok
2.	Jumat, 4 November 2022	<ul style="list-style-type: none"> - Asistensi 1 - Diskusi kelompok cara pengerjaan dan pola pikir pengerjaan
3.	Sabtu, 5 November 2022	Diskusi kelompok mengenai kesulitan yang dihadapi dan saling memberikan solusi
4.	Rabu, 9 November 2022	<i>Debugging</i> code, penyatuan program utama
5.	Jumat, 11 November 2022	<ul style="list-style-type: none"> - Asistensi 2 - Penyusunan laporan