

# LAPORAN TUGAS BESAR

## IF2111 Algoritma dan Struktur Data

### BNMO


Dipersiapkan oleh:

Kelompok 10

Attariq Muhammad Azhar	18221043
Mochamad Syahrial Alzaidan	18221055
Iskandar Muda Rizky Parlambang	18221109
Devina Ar'raudah	18221113
Arifuddin Achmad Subagja	18221127

Sekolah Teknik Elektro dan Informatika - Institut Teknologi Bandung

Jl. Ganesha 10, Bandung 40132

	Sekolah Teknik Elektro dan Informatika ITB	Nomor Dokumen		Halaman
		IF2111-TB1-10		39
		Revisi	-	11 November 2022

# Daftar Isi

<b>1 Ringkasan</b>	<b>3</b>
<b>2 Penjelasan Tambahan Spesifikasi Tugas</b>	<b>4</b>
2.1 Game Tebak Kata	5
<b>3 Struktur Data (ADT)</b>	<b>5</b>
3.1 ADT Array	5
3.2 ADT Mesin Karakter dan Mesin Kata	6
3.3 ADT Queue	7
3.4 ADT Map	8
<b>4 Program Utama</b>	<b>9</b>
<b>5 Algoritma-Algoritma Menarik</b>	<b>9</b>
5.1 CONCAT Function	10
5.2 WordToInt Function	10
<b>6 Data Test</b>	<b>11</b>
6.1 Welcome Page	11
6.2 START	11
6.3 LOAD <filename>	12
6.4 SAVE <filename>	12
6.5 CREATE GAME	13
6.6 LIST GAME	14
6.7 DELETE GAME	14
6.8 QUEUE GAME	16
6.9 PLAY GAME	17
6.10 SKIP GAME <n>	19
6.11 QUIT	20
6.12 HELP	21
6.13 COMMAND LAIN	21
6.15 Diner Dash	23
6.16 ADT Array	27
6.17 ADT Mesin Karakter dan Mesin Kata	28
6.18 ADT Map	29
6.19 ADT Queue	30
<b>7 Test Script</b>	<b>31</b>

<b>8</b>	<b>Pembagian Kerja dalam Kelompok</b>	<b>33</b>
<b>9</b>	<b>Lampiran</b>	<b>34</b>
9.1	Deskripsi Tugas Besar 1	34
9.2	Notulen Rapat	36
9.3	Log Activity Anggota Kelompok	39

# 1 Ringkasan

Lima orang mahasiswa STI diminta tolong untuk mengurus BNMO (dibaca: Binomo). BNMO adalah sebuah robot video *game console* yang dimiliki oleh Indra dan Doni. BNMO ini merupakan video game console terancang di zamannya. BNMO juga tidak bisa dimiliki oleh sembarang orang sebab merupakan suatu *product elite* dari perusahaan TESLA. Barang ini hanya bisa dipesan dengan menggunakan sistem *pre-order* dan ada mekanisme seleksi untuk siapa yang berhak membelinya.

BNMO merupakan *game console* yang dapat memainkan beberapa permainan seru di dalamnya. BNMO diprogram dengan bahasa C dan berbasis CLI (Command Line Interface). Ada permainan RNG, Dinner Dash, dan masih banyak lainnya. Permainan apa saja yang bisa dimainkan di sini tergantung pada pengguna. Artinya, BNMO ini juga memiliki fitur Create Game untuk membuat game baru. Beberapa fitur lain BNMO yang menunjang mesin seru ini, antara lain Start, Load, Save, List Game, Delete Game, Queue Game, Play Game, Skip Game, Quit, dan Help. Semua fitur-fitur ini disusun dengan menggunakan struktur data terkait *array*, mesin kata, mesin karakter, dan *queue*.

Alur penggunaan dari BNMO ini dimulai dengan cara menekan tombol START nya. Sebelum memainkan game kita harus melakukan Load Game terhadap *file* yang pernah kita simpan sebelumnya. Tujuannya adalah agar pencapaian permainan kita sebelumnya dan dapat diteruskan, alias tidak memulai dari nol lagi. Setelah itu, kita bisa memainkan game yang kita inginkan dengan menambahkannya terlebih dahulu dalam semacam *playlist* dengan fitur Queue Game. Kita hanya bisa memainkan *game* sesuai urutannya. Namun, jika kita bosan dan sudah terlanjur memasukkannya dalam *playlist* tadi, kita bisa menggunakan Skip Game untuk segera menuju *game* yang kita inginkan. Jika kita bingung ingin memilih *game* apa, cukup gunakan List Game. Namun, jika masih belum menemukan *game* yang diinginkan, kita bisa memakai fitur Create Game untuk memasukkan *game* yang kita inginkan. Jika kita bosan dengan *game* yang ada, kita juga bisa menghapusnya dari BNMO dengan Delete Game. Jangan lupa gunakan fitur Help jika masih bingung dengan cara memakai BNMO dan tekan Quit jika sudah tidak memakainya lagi agar hemat energi.

Akan tetapi, sejak dua bulan yang lalu *game console* elit ini mengalami kerusakan dan telah diperbaiki. Sayangnya, setelah diperbaiki ia justru mendapatkan lebih banyak bug dalam sistemnya. Oleh karena itu, Indra dan Doni meminta tolong 5 mahasiswa STI (dengan asumsi bahwa mereka adalah programmer handal) untuk memprogram ulang robot video *game console* kesayangannya. Dengan adanya mega proyek ini, 5 mahasiswa tersebut tidak hanya membantu Indra dan Doni untuk kembali bahagia, tetapi juga media pembelajaran yang baik untuk kami mempelajari bahasa C dengan lebih baik lagi. Selama pengerjaan BNMO ini, mahasiswa tersebut dapat memahami serta mempelajari bagaimana membuat program yang cukup kompleks yang dibantu dengan materi struktur data terkait yang telah dipelajari oleh mereka. Mega proyek ini juga membantu 5 mahasiswa ini menemukan ide dan alur dari program *game console* ini.

## 2 Penjelasan Tambahan Spesifikasi Tugas

Pada video game console BNMO kami menyediakan game RNG, Diner Dash. Selain itu, terdapat juga game tambahan yang dapat dimainkan oleh user. Game tambahan tersebut adalah game tebak kata.

## 2.1 Game Tebak Kata

*Game* Tebak Kata merupakan sebuah *game* yang menantang pemainnya untuk menebak kata yang ada. Berikut adalah spesifikasi *game* ini:

1. Kata diambil secara *random* dari kosa kata yang terdapat pada program.
2. Pemain diberikan kesempatan 10 kali untuk mencoba menebak kata tersebut.
3. Kata dapat berupa nama orang, tempat ataupun negara.
4. Kata yang ditebak hanya terdiri atas satu buah kata.
5. Game akan berakhir apabila pemain dapat menebak kata dengan benar ataupun gagal menebak setelah 10 kali mencoba.

## 3 Struktur Data (ADT)

BNMO merupakan *game console* yang cukup menarik karena terdapat berbagai fitur di dalamnya. Selain fitur pengelolaan *game*, BNMO juga bisa memainkan langsung *game* tersebut. Namun, dalam pemrogramannya tentu membutuhkan beberapa bantuan ADT (*Abstract Data Type*). Dalam hal ini, kami menggunakan ADT Array, ADT Mesin Karakter dan Mesin Kata, serta ADT Queue.

### 3.1 ADT Array

#### A. Sketsa Struktur Data

- `ArrayDin MakeArrayDin();`  
Primitif ini merupakan primitif untuk membuat *array* kosong.
- `void DeallocateArrayDin(ArrayDin *array);`  
Primitif ini digunakan untuk mendealokasikan sebuah *array*.
- `boolean IsEmpty(ArrayDin array);`  
Primitif ini merupakan sebuah fungsi yang mengirimkan *true* apabila *array* kosong.
- `int Length (ArrayDin array);`  
Primitif ini merupakan fungsi yang mengembalikan jumlah elemen yang ada pada suatu *array*.
- `ElType Get(ArrayDin array, IdxType i);`  
Primitif ini digunakan untuk mendapatkan elemen indeks ke-i dari *array*.
- `int GetCapacity(ArrayDin array);`  
Primitif ini digunakan untuk mendapatkan kapasitas yang tersedia pada *array*.
- `void InsertAt(ArrayDin *array, ElType el, IdxType i);`  
Primitif ini digunakan untuk menambahkan elemen baru el di indeks ke-i pada *array*.
- `void InsertLast(ArrayDin *array, ElType el);`  
Primitif ini digunakan untuk menambahkan elemen el di akhir *array*.
- `void InsertFirst(ArrayDin *array, ElType el);`  
Primitif ini digunakan untuk menambahkan elemen pada awal *array*.
- `void DeleteAt(ArrayDin *array, IdxType i);`

Primitif ini digunakan untuk menghapus elemen pada indeks ke-i pada suatu *array*.

- `void DeleteLast(ArrayDin *array);`  
Primitif ini digunakan untuk menghapus elemen terakhir pada suatu *array*.
- `void DeleteFirst(ArrayDin *array);`  
Primitif ini digunakan untuk menghapus elemen pertama pada suatu *array*.
- `void PrintArrayDin(ArrayDin array);`  
Primitif ini digunakan untuk melakukan *print* suatu *array*.
- `ArrayDin CopyArrayDin(ArrayDin array);`  
Primitif ini merupakan fungsi yang mengembalikan *array* baru dengan elemen sama dengan *array* masukannya.
- `IdxType SearchArrayDin(ArrayDin array, ElType el);`  
Primitif ini merupakan fungsi yang mengembalikan indeks dimana *el* ditemukan, dan apabila tidak ditemukan akan mengembalikan nilai -1.

B. Persoalan yang Diselesaikan

ADT ini digunakan untuk menangani fungsi-fungsi, seperti List Game, Delete Game, dan Create Game.

C. Alasan Pemilihan

ADT ini dipilih karena dalam hal manajemen list dari *game* apa yang disimpan dalam BNMO tentu lebih mudah dengan ADT Array. Penggunaannya cukup simpel dan fleksibel. Saat kita ingin menambahkan *game*, kita bisa menaruhnya pada indeks paling akhir. Jika kita ingin menghapus *game*, kita bisa menghapus *game* pada indeks ke-i dengan memajukan indeks elemen lainnya setelah *game* tersebut dan mengurangi nilai efektifnya. Saat mengakses pun, kita hanya perlu mengakses indeksnya.

D. Implementasi

ADT Array diimplementasikan sebagai ADT Array dengan nama *file* header *arraydin.h*.

### 3.2 ADT Mesin Karakter dan Mesin Kata

A. Sketsa Struktur Data

1. Mesin Karakter

- `void LoadPita(char* filename, boolean isF);`  
Membaca pita sesuai dengan tipe yang diinginkan (*file* atau masukan).
- `void StopLoadPita();`  
Primitif ini digunakan untuk menghentikan pembacaan *file*.
- `void START();`  
Primitif ini digunakan untuk memulai pita karakter dan menyimpan *current character* ke CC.
- `void ADV();`  
Primitif ini digunakan untuk memajukan pita satu karakter.
- `char GetCC();`  
Primitif ini digunakan untuk mengirimkan *current character* pada pita.
- `boolean IsEOP();`

Primitif ini merupakan sebuah fungsi yang mengirimkan true apabila pita telah habis.

## 2. Mesin Kata

- `void IgnoreBlanks();`

Primitif ini merupakan prosedur untuk mengabaikan satu atau beberapa *blank*.

- `void STARTWORD();`

Primitif ini merupakan prosedur untuk memulai pita dan menyimpan kata ke *current word*.

- `void ADVWORD();`

Primitif ini merupakan prosedur untuk melanjutkan pita ke kata selanjutnya.

- `void CopyWord();`

Primitif ini merupakan prosedur untuk mengakuisisi kata dan menyimpannya dalam *currentWord*.

- `int WordToInt(Word W);`

Primitif ini merupakan fungsi yang merubah kata menjadi sebuah bilangan bulat.

- `char *WordToString(Word W);`

Primitif ini merupakan fungsi yang merubah Word menjadi bentuk String tanpa mengubah *state* awal dari W.

- `boolean isWordSame(Word W1, Word W2);`

Primitif ini merupakan fungsi yang mengecek apakah kata W1 dan W2 merupakan kata yang sama.

- `Word stringToWorld(char *s);`

Primitif ini digunakan untuk mengubah string s menjadi Word, tanpa mengubah *state* awal dari s.

## B. Persoalan yang Diselesaikan

ADT ini digunakan untuk menangani fungsi-fungsi, seperti Load Game, Save Game, dan Start Game.

## C. Alasan Pemilihan

Penyesuaian dengan aturan tugas besar yang tidak boleh menggunakan fungsi “scanf()” membuat ADT ini sangat dibutuhkan untuk menentukan fungsi apa yang dipanggil. Selain itu, ADT ini juga digunakan untuk menentukan nama *file* apa yang ingin kita masukkan dalam BNMO.

## D. Implementasi

ADT ini diimplementasikan dengan mesinkarakter.h dan mesinkata.h.

## 3.3 ADT Queue

### A. Sketsa Struktur Data

- `void CreateQueue(Queue * q);`

Primitif yang berfungsi untuk membuat *queue* kosong dengan menetapkan `IDX_HEAD` dan `IDX_TAIL` sebagai `IDX_UNDEF`

- `boolean isEmpty(Queue q);`

Fungsi yang mengirimkan *true* apabila *queue* kosong.

- `boolean isFull(Queue q);`  
Primitif ini mengirimkan value *true* apabila seluruh elemen *queue* telah terisi.
- `int length(Queue q);`  
Primitif ini mengembalikan jumlah elemen yang ada di dalam *queue*.
- `void enqueue(Queue *q, ElType val);`  
Primitif ini berfungsi menambahkan *val* pada *q* dengan aturan FIFO.
- `void dequeue (Queue *q, ElType *val);`  
Primitif ini berfungsi menghapus *val* pada *q* dengan aturan FIFO.
- `void displayQueue(Queue q);`  
Primitif ini berfungsi menuliskan isi *queue* secara traversal, *queue* ditulis di antara kurung siku; antara dua elemen dipisahkan dengan separator "koma", tanpa tambahan karakter di depan, di tengah, atau di belakang, termasuk spasi dan enter

B. Persoalan yang Diselesaikan

ADT ini digunakan untuk menyelesaikan fungsi-fungsi, seperti Queue Game, Play Game, dan Skip Game.

C. Alasan Pemilihan

Dalam skema untuk memilih dan memainkan *game* yang diinginkan, kita bisa memodelkannya dalam struktur data Queue. Dengan model data *queue*, maka *game* yang ingin kita mainkan akan kita masukkan dalam semacam *playlist* untuk menunggu giliran dimainkan. Saat kita ingin memainkan, cukup gunakan Play Game dan BNMO akan otomatis memainkan *game* yang ada di urutan paling depan. Jika kita tidak ingin memainkan *game* yang telah terlanjur masuk antrian, cukup kita Skip Game *playlist* tersebut. Selain pada fungsi itu, ADT Queue sangat berguna dalam implementasi *game* Dinner Dash.

D. Implementasi

ADT ini diimplementasikan dengan `queue.h`.

### 3.4 ADT Map

A. Sketsa Struktur Data

- `void CreateEmpty(Map *M);`  
Primitif ini berfungsi membuat map *M* menjadi kosong.
- `boolean IsMapEmpty (Map M);`  
Primitif ini mengirimkan *true* apabila map kosong.
- `boolean IsFull (Map M);`  
Primitif ini mengirimkan *true* apabila map penuh.
- `valuetype Value (Map M, keytype k);`  
Primitif ini mengembalikan nilai value dengan key *k* dari map *M*.
- `void Insert (Map *M, keytype k, valuetype v);`  
Primitif ini merupakan prosedur untuk menambahkan elemen dengan key *k* dan value *v* ke map *M*.
- `void Delete (Map *M, keytype k);`  
Primitif ini menghapus elemen dengan key *k* dari map *M*.



- B. Persoalan yang Diselesaikan  
ADT ini digunakan untuk menangani permasalahan yang ada di game bonus “Tebak Kata”.
- C. Alasan Pemilihan  
ADT ini dipilih karena dapat menyimpan kunci jawaban dan *clue* dari game Tebak Kata. Sebagai contoh, kita dapat menyimpan jawaban “gibeh” pada key dan “dinosaur” sebagai *clue* pada value di suatu Map sehingga pengaksesan dapat dijalankan dengan lebih mudah.
- D. Implementasi  
ADT Array diimplementasikan sebagai ADT Map dengan nama *file* header map.h.

## 4 Program Utama

Program utama dalam robot video game console BNMO dimulai dengan memanggil fungsi “*bnmo\_pic*” untuk menampilkan rangkaian karakter yang membentuk gambar dan tulisan dari BNMO serta welcome page dalam game console tersebut. Kemudian, program menampilkan pilihan menu yang ada dalam game console tersebut. Dalam prosedur ini, pemain akan diminta untuk memberikan sebuah masukan yang menentukan apakah ingin START untuk langsung memulai permainan dari awal tanpa memuat data permainan sebelumnya atau LOAD untuk memuat data permainan selanjutnya. Program akan memeriksa apakah inputan tersebut valid atau tidak. Jika inputan tidak valid, akan ditampilkan kalimat yang menunjukkan bahwa command tersebut tidak dikenali. Selanjutnya, pengguna akan diminta untuk memberikan inputan yang valid.

Ketika command atau inputan yang diterima adalah LOAD maka akan dilanjutkan dengan argumen lain yaitu filename yang merepresentasikan suatu *save file* yang ingin dibuka. Akan tetapi, jika command yang diterima adalah START, BNMO akan memulai games tersebut. Berikutnya, ketika games tidak kosong dan command yang dimasukkan adalah LOAD, maka *save file* berhasil dibaca dan BNMO berhasil dijalankan. Namun, jika command yang dimasukkan adalah START, *file* konfigurasi sistem akan berhasil dibaca dan BNMO berhasil dijalankan. Akan tetapi, jika games kosong, BNMO akan gagal dijalankan.

Program akan dimulai ketika Games tidak kosong. Lalu, BNMO akan menampilkan main menunya. Kemudian, pengguna akan diminta untuk menginputkan command yang diinginkan. Ketika command yang dimasukkan valid, program akan memanggil fungsi sesuai command yang dimasukkan. Ketika command yang di input adalah START and LOAD, akan ditampilkan bahwa sistem telah membaca *file*, sehingga pengguna dapat memulai ulang sistem untuk membaca *file* lain. Namun, jika yang diinputkan command tidak sesuai, akan ditampilkan kalimat bahwa command tidak dikenali dan pengguna akan diminta untuk memasukkan kembali command yang valid. Jika program keluar, maka akan otomatis menyimpan data dengan SAVE. Kemudian, user menerima ucapan *bye-bye* untuk yang terakhir.

## 5 Algoritma-Algoritma Menarik

Dalam pembuatan sistem BNMO ini, terdapat dua buah algoritma yang dikategorikan sebagai menarik, yaitu *Concat Function* dan *WordToInt Function*. Kedua fungsi tersebut dianggap memiliki algoritma yang menarik karena dalam menjalankan sistem BNMO ini kedua

fungsi tersebut digunakan untuk mengubah masukan *user* menjadi suatu nilai yang dapat diproses oleh sistem. Fungsi Concat diterapkan pada *file* create\_game.c untuk menggabungkan setiap masukan kata oleh *user* sedangkan fungsi WordToInt diterapkan pada *file* mesinkata.c untuk mengubah masukan dengan tipe data Word menjadi sebuah *integer*.

## 5.1 CONCAT Function

```
char* concat(char* s1, char* s2){
/* Mengembalikan hasil konkatenasi s1 dan s2 */
    int i = 0, j = 0;
    int lenS1 = strlen(s1), lenS2 = strlen(s2);
    char* str = (char*) malloc(100*sizeof(char));
    while (i < lenS1){
        str[i] = s1[i];
        i++;
    }
    str[i] = ' ';
    i++;
    while (j < lenS2){
        str[i] = s2[j];
        i++;
        j++;
    }
    str[i] = '\0';
    return str;
}
```

Fungsi concat adalah fungsi yang mengembalikan hasil konkatenasi antara dua *string*. Algoritma tersebut dianggap menarik karena fungsi ini merupakan implementasi fungsi primitif list, yaitu concat. Hal yang membedakan fungsi ini dengan fungsi concat pada primitif list adalah adanya spasi (" ") antara dua buah string. Fungsi concat ini dipakai pada program Create Game untuk menggabungkan setiap kata yang dimasukkan oleh *user*.

## 5.2 WordToInt Function

```
int WordToInt(Word W) {
/* Fungsi mengubah Word ke integer */
    int i;
    int temp = 0;
    for (i = 0; i < W.Length; i++){
        if (W.TabWord[i] != BLANK) {
            temp = temp * 10 + (W.TabWord[i] - '0');
        }
    }
    return temp;
}
```

}

Fungsi WordToInt digunakan untuk mengembalikan angka yang masih berbentuk *string* menjadi sebuah angka bertipe *integer*. Algoritma pada fungsi ini dianggap menarik karena di dalam sistem, mesin kata akan membaca masukan *user* sebagai masukan bertipe *Word* dan ketika sistem membutuhkan suatu masukan bertipe *integer* dari *user*, fungsi WordToInt digunakan sebagai *converter* masukan *Word* menjadi *integer*. Fungsi ini dipakai di beberapa program menu. Di menu load, mesin kata akan membaca baris pertama pada *file* yang merupakan jumlah game yang tersedia di daftar *game* lalu fungsi WordToInt digunakan untuk mengembalikan angka dengan nilai yang sesuai dengan baris yang dibaca oleh mesin kata. Fungsi ini juga dipakai di *game* RNG untuk mengembalikan nilai *integer* yang sesuai dengan masukan *user*. Nilai *integer* tersebut kemudian akan dijadikan pembandingan jawaban *user* dengan jawaban yang benar.

## 6 Data Test

Fitur-fitur yang disediakan oleh sistem akan dilakukan pengetesan dengan memberikan masukan oleh *user* dan membandingkan kesamaan keluaran dari sistem dengan yang ada pada spesifikasi tugas besar.

### 6.1 Welcome Page

Deskripsi
<p><i>Test</i> ini dilakukan untuk memastikan bahwa program sudah dapat berjalan serta dapat menampilkan tampilan judul game serta main menu. Cara melakukan kompilasi program adalah menggunakan perintah gcc:</p> <pre>“gcc bnmo_pic.c program/help.c program/skipgame.c program/playgame.c program/queuegame.c program/delete_game.c program/list_game.c program/create_game.c program/save.c program/load.c program/start.c program/ADT/queue/queue.c program/ADT/mesinkarkata/mesinkar.c program/ADT/mesinkarkata/mesinkata.c program/ADT/arraydin/arraydin.c main.c game/dinnerdash.c game/rng.c game/tebakkata.c game/queuedinnerdash.c program/ADT/Map/map.c -o main”</pre> <p>Pada terminal yang ingin digunakan, lalu jalankan executable code main tersebut.</p>
Data Test



harus menginputkan argumen filename yang menggambarkan suatu *save file* yang ingin dibuka.

#### Hasil yang seharusnya diberikan

ENTER COMMAND: **LOAD savefile1.txt**  
Save file berhasil dibaca. BNMO berhasil dijalankan.

#### Data Test

```
PILIHAN MENU:
=> START
=> LOAD (filename.txt)
ENTER COMMAND: LOAD savefile1.txt

Save file berhasil dibaca. BNMO berhasil dijalankan ^^
```

Gambar 6.3 Tampilan BNMO saat user memanggil *command* LOAD beserta nama *save file*-nya

## 6.4 SAVE <filename>

#### Deskripsi

*Test* ini digunakan untuk memastikan input SAVE akan menyimpan state game pengguna saat ini ke dalam suatu *file*. Pengguna juga harus menginputkan argumen filename yang menggambarkan suatu *file* yang akan disimpan pada disk.

#### Hasil yang seharusnya diberikan

ENTER COMMAND: **SAVE savefile1.txt**  
Save file berhasil disimpan.

#### Data Test

```
ENTER COMMAND: SAVE savefile2.txt
Save file berhasil disimpan ^^

ENTER COMMAND: █
```

Gambar 6.4 Tampilan BNMO saat user memanggil *command* SAVE beserta nama *save file*-nya

## 6.5 CREATE GAME

#### Deskripsi

*Test* ini digunakan untuk memastikan input CREATE GAME akan menambahkan game baru di daftar game. Akan tetapi, saat game tersebut telah tersedia maka akan ditampilkan

bahwa game tersebut telah tersedia, dan user akan diminta untuk memasukkan nama game lainnya.

### Hasil yang seharusnya diberikan

ENTER COMMAND: **CREATE GAME**

Masukkan nama game yang akan ditambahkan: **EXTRA1**

Game berhasil ditambahkan

### Data Test

```
ENTER COMMAND: CREATE GAME
Masukkan nama game yang akan ditambahkan: CALL OF DUTY

  +---+
  |   |
  |   |
  | + o |   Game CALL OF DUTY berhasil ditambahkan ^^
  | , o |
  |   |
  +---+

ENTER COMMAND: LIST GAME

Berikut adalah daftar game yang tersedia:
1. RNG
2. Diner DASH
3. DINOSAUR IN EARTH
4. RISEWOMAN
5. EIFFEL TOWER
6. CALL OF DUTY

ENTER COMMAND: █
```

Gambar 6.5 Tampilan BNMO saat user memanggil *command* CREATE GAME serta memasukkan nama *game* yang ingin ditambahkan.

## 6.6 LIST GAME

### Deskripsi

*Test* ini digunakan untuk memastikan input LIST GAME akan menampilkan daftar game yang disediakan oleh sistem.

### Hasil yang seharusnya diberikan

ENTER COMMAND: **LIST GAME**

Berikut adalah daftar game yang tersedia

1. RNG
2. LUNCH SLOW
3. DINOSAUR IN EARTH
4. RISEWOMAN
5. EIFFEL TOWER

### Data Test

ENTER COMMAND: LIST GAME

Berikut adalah daftar game yang tersedia:

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

Gambar 6.6 Tampilan BNMO saat user memanggil *command* LIST GAME

## 6.7 DELETE GAME

Deskripsi
<p><i>Test</i> ini digunakan untuk memastikan input DELETE GAME dapat menghapus sebuah game dari daftar game. Akan tetapi, game yang terdapat di <i>queue game</i> dan 5 game pertama dalam <i>file</i> konfigurasi tidak dapat dihapus.</p>
Hasil yang seharusnya diberikan
<p>ENTER COMMAND: <b>DELETE GAME</b></p> <p>Berikut adalah daftar game yang tersedia</p> <ol style="list-style-type: none"><li>1. RNG</li><li>2. LUNCH SLOW</li><li>3. DINOSAUR IN EARTH</li><li>4. RISEWOMAN</li><li>5. EIFFEL TOWER</li><li>6. CUSTOM GAME 1</li></ol> <p>Masukkan nomor game yang akan dihapus: 6</p> <p>Game berhasil dihapus</p>
<p>ENTER COMMAND: <b>DELETE GAME</b></p> <p>Berikut adalah daftar game yang tersedia</p> <ol style="list-style-type: none"><li>1. RNG</li><li>2. LUNCH SLOW</li><li>3. DINOSAUR IN EARTH</li><li>4. RISEWOMAN</li><li>5. EIFFEL TOWER</li></ol> <p>Masukkan nomor game yang akan dihapus: 1</p> <p>Game gagal dihapus</p>
Data Test

```

ENTER COMMAND: DELETE GAME

Berikut adalah daftar game yang tersedia:
1. RNG
2. Diner DASH
3. DINOSAUR IN EARTH
4. RISEWOMAN
5. EIFFEL TOWER
6. CALL OF DUTY

Masukkan nomor game yang akan dihapus: 6
Game yang akan dihapus: CALL OF DUTY
Game berhasil dihapus
ENTER COMMAND: █

```

Gambar 6.7 Tampilan BNMO saat user memanggil *command* DELETE GAME dan memasukkan nomor urutan *game* yang ingin dihapus.

```

Masukkan nomor game yang akan dihapus: 6
Game yang akan dihapus: CALL OF DUTY
Game berhasil dihapus
ENTER COMMAND: LIST GAME

Berikut adalah daftar game yang tersedia:
1. RNG
2. Diner DASH
3. DINOSAUR IN EARTH
4. RISEWOMAN
5. EIFFEL TOWER

```

Gambar 6.8 Tampilan daftar *game* pada BNMO setelah user melakukan penghapusan *game* dari daftar *game*

```

Berikut adalah daftar game yang tersedia:
1. RNG
2. Diner DASH
3. DINOSAUR IN EARTH
4. RISEWOMAN
5. EIFFEL TOWER

Masukkan nomor game yang akan dihapus: 1
Game yang akan dihapus: RNG
Game gagal dihapus
ENTER COMMAND: █

```

Gambar 6.9 Tampilan BNMO saat user ingin melakukan penghapusan *game* yang terdaftar pada *file* konfigurasi *default*.

## 6.8 QUEUE GAME

Deskripsi
Test ini digunakan untuk memastikan input QUEUE GAME untuk dimasukkan ke dalam list permainan. Namun, saat user menjalankan QUIT maka, list queue tersebut akan hilang.
Hasil yang seharusnya diberikan
ENTER COMMAND: <b>QUEUE GAME</b> Berikut adalah daftar antrian game-mu 1. EIFFEL TOWER 2. RISEWOMAN 3. LUNCH SLOW



Berikut adalah daftar game yang tersedia

1. RNG
2. LUNCH SLOW
3. DINOSAUR IN EARTH
4. RISEWOMAN
5. EIFFEL TOWER

Nomor Game yang mau ditambahkan ke antrian: 2

Game berhasil ditambahkan ke dalam daftar antrian.

ENTER COMMAND: **QUEUE GAME**

Berikut adalah daftar antrian game-mu

1. EIFFEL TOWER
2. RISEWOMAN
3. LUNCH SLOW

Berikut adalah daftar game yang tersedia

1. RNG
2. LUNCH SLOW
3. DINOSAUR IN EARTH
4. RISEWOMAN
5. EIFFEL TOWER

Nomor Game yang mau ditambahkan ke antrian: 9

Nomor permainan tidak valid, silahkan masukkan nomor game pada list.

### Data Test

```
ENTER COMMAND: QUEUE GAME
Berikut adalah daftar antrian game-mu:
1. Diner DASH
2. RNG
3. RISEWOMAN

Berikut adalah daftar game yang tersedia:
1. RNG
2. Diner DASH
3. DINOSAUR IN EARTH
4. RISEWOMAN
5. EIFFEL TOWER

Nomor Game yang mau ditambahkan ke antrian: 1

Game berhasil ditambahkan kedalam daftar antrian.

ENTER COMMAND: █
```

*Gambar 6.10* Tampilan BNMO saat user memanggil *command* QUEUE GAME dan memasukkan nomor urutan *game* pada daftar *game* yang ingin ditambahkan ke *queue*.

```

ENTER COMMAND: QUEUE GAME
Berikut adalah daftar antrian game-mu:
1. Diner DASH
2. RNG
3. RISEWOMAN
4. RNG

Berikut adalah daftar game yang tersedia:
1. RNG
2. Diner DASH
3. DINOSAUR IN EARTH
4. RISEWOMAN
5. EIFFEL TOWER

Nomor Game yang mau ditambahkan ke antrian: 7

Nomor permainan tidak valid, silahkan masukkan nomor game pada list.
ENTER COMMAND: █

```

Gambar 6.11 Tampilan BNMO saat user memasukkan nomor urutan game di luar daftar game.

## 6.9 PLAY GAME

Deskripsi
Test ini digunakan untuk menguji command PLAY GAME. Command ini akan memainkan game berdasarkan antrian game. Akan tetapi, game yang bisa dimainkan hanya game yang terdapat pada spesifikasi. Selain itu, game tidak dapat dimainkan.
Hasil yang seharusnya diberikan
<p>ENTER COMMAND: <b>PLAY GAME</b></p> <p>Berikut adalah daftar Game-mu</p> <ol style="list-style-type: none"> <li>1. EIFFEL TOWER</li> <li>2. RISEWOMAN</li> <li>3. LUNCH SLOW</li> </ol> <p>Loading EIFFEL TOWER ...</p>
<p>ENTER COMMAND: <b>PLAY GAME</b></p> <p>Berikut adalah daftar Game-mu</p> <ol style="list-style-type: none"> <li>1. RISEWOMAN</li> <li>2. RISEWOMAN</li> <li>3. LUNCH SLOW</li> </ol> <p>Game RISEWOMAN masih dalam maintenance, belum dapat dimainkan. Silahkan pilih game lain.</p>
Data Test

```

Diner Dash

Selamat Datang di Diner Dash!

=====
Saldo: 0

Daftar Pesanan
Makanan      | Durasi memasak | Ketahanan | Harga
-----
M0            | 1              | 1         | 40526
M1            | 3              | 1         | 18016
M2            | 3              | 5         | 35773

Daftar Makanan yang sedang dimasak
Makanan      | Sisa durasi memasak
-----
M0            | 1
M1            | 3
M2            | 3

Daftar Makanan yang dapat disajikan
Makanan      | Sisa ketahanan makanan
-----
M0            | 1
M1            | 3
M2            | 3

MASUKKAN COMMAND: COOK M0
Berhasil memasak M0

=====
Saldo: 0

Daftar Pesanan
Makanan      | Durasi memasak | Ketahanan | Harga
-----
M0            | 1              | 1         | 40526
M1            | 3              | 1         | 18016
M2            | 3              | 5         | 35773

Daftar Makanan yang sedang dimasak
Makanan      | Sisa durasi memasak
-----
M0            | 1
M1            | 3
M2            | 3

Daftar Makanan yang dapat disajikan
Makanan      | Sisa ketahanan makanan
-----
M0            | 1
M1            | 3
M2            | 3

```

Gambar 6.12 Tampilan BNMO saat user menjalankan play game

```

ENTER COMMAND: PLAY GAME
Berikut adalah daftar antrian game-mu:
1. RISEWOMAN
2. RNG

Game RISEWOMAN masih dalam maintenance, belum dapat dimainkan.
Silahkan pilih game lain.

ENTER COMMAND: █

```

Gambar 6.13 Tampilan BNMO saat user menjalankan play game namun belum ada antrian game

## 6.10 SKIP GAME <n>

Deskripsi
<i>Test</i> ini digunakan untuk menguji command SKIP GAME. Command ini akan melewati permainan sebanyak n.
Hasil yang seharusnya diberikan
<p>ENTER COMMAND: <b>SKIP GAME 2</b></p> <p>Berikut adalah daftar Game-mu</p> <ol style="list-style-type: none"> <li>RISEWOMAN</li> <li>LUNCH SLOW</li> <li>RISEWOMAN</li> </ol> <p>Loading RISEWOMAN ...</p>

Berikut adalah daftar Game-mu

1. RISEWOMAN
2. LUNCH SLOW
3. RISEWOMAN

Tidak ada permainan lagi dalam daftar game-mu.

## Data Test

```
ENTER COMMAND: SKIP GAME 2
Berikut adalah daftar antrian game-mu:
1. Diner DASH
2. RISEWOMAN
3. RNG

Loading RNG ...

----- - -----
|_ _ _ \ \ \ |_- _ _ \
|_|_ / / \ ||_- _ \
|_|_ / / . ||_- _ \
|_|_ \ \ \ \ \ ||_- _ \
\_|_ \ \ \ | \ \ \ _ _ _ \
RNG Telah dimulai. Uji keberuntungan Anda dengan menebak X.
Tebakan: █
```

Gambar 6.14 Tampilan BNMO saat *user* menjalankan skip game sebanyak 2

```
ENTER COMMAND: SKIP GAME 3
Berikut adalah daftar antrian game-mu:
1. Diner DASH
2. RNG
Tidak ada permainan lagi dalam daftar game-mu.
ENTER COMMAND: █
```

Gambar 6.15 Tampilan BNMO saat *user* menjalankan skip game lebih dari jumlah antrian *game*.

## 6.11 QUIT

## Deskripsi

*Test* ini digunakan untuk menguji command QUIT yang akan membuat user keluar dari program.

**Hasil yang seharusnya diberikan**

ENTER COMMAND: QUIT

Anda keluar dari game BNMO.  
Bye bye ...

#### Data Test

```
ENTER COMMAND: QUIT
Save file berhasil disimpan ^^

Anda Keluar dari game BNMO
Bye byee.....
```

Gambar 6.16 Tampilan BNMO saat user menjalankan Quit pada program

## 6.12 HELP

#### Deskripsi

Test ini digunakan untuk menguji command HELP yang akan membantu user untuk mengetahui command yang terdapat di program.

#### Data Test

```
'Kamu Nanya Menunya Apa Aja?'
```

1	START	Untuk menjalankan BNMO
2	LOAD	Untuk membaca save file yang berisi list game yang dapat dimainkan, histori dan scoreboard game
3	SAVE	Untuk menyimpan state game pemain saat ini
4	CREATE GAME	Untuk menambahkan game baru pada daftar game
5	LIST GAME	Untuk menampilkan daftar game yang disediakan oleh sistem
6	DELETE GAME	Untuk menghapus sebuah game dari daftar game
7	QUEUE GAME	Untuk mendaftarkan permainan kedalam list
8	PLAY GAME	Untuk memainkan sebuah permainan
9	SKIPGAME	Untuk melewatkan permainan sebanyak yang kamu mau
10	QUIT	Untuk keluar dari program
11	HELP	Untuk bantuin kamu

```
ENTER COMMAND: _
```

Gambar 6.17 Tampilan BNMO saat user menjalankan help pada program

## 6.13 COMMAND LAIN

#### Deskripsi

Test ini digunakan untuk menguji program apabila command yang dimasukkan tidak valid. Program akan meminta kembali inputan yang valid.

#### Data Test

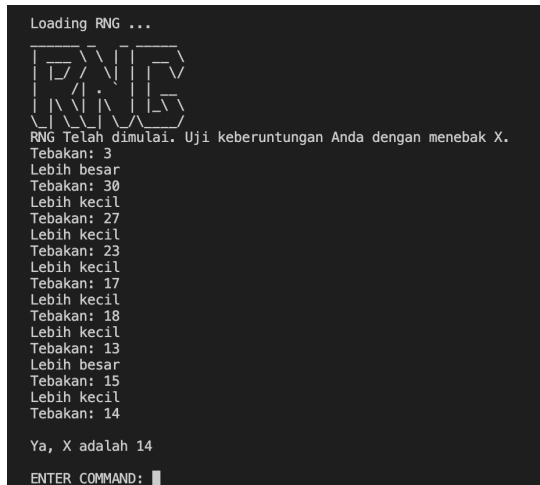
```

PILIHAN MENU:
=> START
=> LOAD (filename.txt)
ENTER COMMAND: COMMAND LAIN
Command tidak dikenali, silakan masukkan command yang valid.
ENTER COMMAND: █

```

Gambar 6.18 Tampilan BNMO saat user memasukkan input yang tidak dikenali.

## 6.14 RNG

Deskripsi
<p><i>Test</i> ini digunakan untuk menguji permainan RNG. Permainan ini akan menghasilkan outcome berupa <i>random number</i>. Permainan ini pemain akan diberi kesempatan untuk menebak angka X. Permainan ini akan selesai jika pemain menebak angka X dengan benar.</p>
Hasil yang seharusnya diberikan
<p>RNG Telah dimulai. Uji keberuntungan Anda dengan menebak X.  Tebakan: <b>80</b>  Lebih kecil  Tebakan: <b>40</b>  Lebih besar  Tebakan: <b>41</b>    Ya, X adalah 41.</p>
Data Test
 <p>The screenshot shows the RNG game interface. It starts with a loading screen, followed by a grid of numbers. The user is prompted to guess a number X. The interface shows a series of guesses and feedback messages: 'Tebakan: 3', 'Lebih besar', 'Tebakan: 30', 'Lebih kecil', 'Tebakan: 27', 'Lebih kecil', 'Tebakan: 23', 'Lebih kecil', 'Tebakan: 17', 'Lebih kecil', 'Tebakan: 18', 'Lebih kecil', 'Tebakan: 13', 'Lebih besar', 'Tebakan: 15', 'Lebih kecil', 'Tebakan: 14'. The final message is 'Ya, X adalah 14'.</p>

Gambar 6.19 Tampilan BNMO saat user memainkan game Diner Dash pada program

## 6.15 Diner Dash

Deskripsi																																
<p>Test ini digunakan untuk menguji permainan Diner Dash. Permainan ini merupakan permainan mengantar makanan tapi berurut berdasarkan prioritasnya. Pada game ini terdapat 3 command yakni Cook, Serve, dan skip. Permainan dimulai dengan 3 pelanggan. Setiap pelanggan dapat memesan satu makanan. Skor akhir dari pemain adalah total uang yang diterima oleh pemain.</p>																																
Hasil yang seharusnya diberikan																																
<p>Selamat Datang di Diner Dash!</p> <p>SALDO: 0</p> <p>Daftar Pesanan</p> <table><tr><th>Makanan</th><th>Durasi memasak</th><th>Ketahanan</th><th>Harga</th></tr><tr><td>M0</td><td>2</td><td>3</td><td>15000</td></tr><tr><td>M1</td><td>3</td><td>1</td><td>15000</td></tr><tr><td>M2</td><td>1</td><td>4</td><td>15000</td></tr></table> <p>Daftar Makanan yang sedang dimasak</p> <table><tr><th>Makanan</th><th>Sisa durasi memasak</th></tr><tr><td></td><td></td></tr></table> <p>Daftar Makanan yang dapat disajikan</p> <table><tr><th>Makanan</th><th>Sisa ketahanan makanan</th></tr><tr><td></td><td></td></tr></table> <p>MASUKKAN COMMAND: <b>COOK M0</b></p> <p>Berhasil memasak M0</p> <p>=====</p> <p>SALDO: 0</p> <p>Daftar Pesanan</p> <table><tr><th>Makanan</th><th>Durasi memasak</th><th>Ketahanan</th><th>Harga</th></tr><tr><td></td><td></td><td></td><td></td></tr></table>	Makanan	Durasi memasak	Ketahanan	Harga	M0	2	3	15000	M1	3	1	15000	M2	1	4	15000	Makanan	Sisa durasi memasak			Makanan	Sisa ketahanan makanan			Makanan	Durasi memasak	Ketahanan	Harga				
Makanan	Durasi memasak	Ketahanan	Harga																													
M0	2	3	15000																													
M1	3	1	15000																													
M2	1	4	15000																													
Makanan	Sisa durasi memasak																															
Makanan	Sisa ketahanan makanan																															
Makanan	Durasi memasak	Ketahanan	Harga																													

M0	2	3	15000
M1	3	1	15000
M2	1	4	15000
M3	1	4	15000

Daftar Makanan yang sedang dimasak  
Makanan | Sisa durasi memasak

-----  
M0 | 2

Daftar Makanan yang dapat disajikan  
Makanan | Sisa ketahanan makanan

-----  
|

MASUKKAN COMMAND: **COOK M1**

Berhasil memasak M1

=====

SALDO: 0

Daftar Pesanan

Makanan | Durasi memasak | Ketahanan | Harga

M0	2	3	15000
M1	3	1	15000
M2	1	4	15000
M3	1	4	15000
M4	1	4	15000

Daftar Makanan yang sedang dimasak  
Makanan | Sisa durasi memasak

-----  
M0 | 1

M1 | 3

Daftar Makanan yang dapat disajikan  
Makanan | Sisa ketahanan makanan

-----  
|

MASUKKAN COMMAND: **COOK M2**



Berhasil memasak M2

Makanan M0 telah selesai dimasak

=====

SALDO: 0

Daftar Pesanan

Makanan	Durasi memasak	Ketahanan	Harga
---------	----------------	-----------	-------

M0	2	3	15000
M1	3	1	15000
M2	1	4	15000
M3	3	1	15000
M4	1	4	15000
M5	1	4	15000

Daftar Makanan yang sedang dimasak

Makanan	Sisa durasi memasak
---------	---------------------

M1	2
M2	1

Daftar Makanan yang dapat disajikan

Makanan	Sisa ketahanan makanan
---------	------------------------

M0	3
----	---

MASUKKAN COMMAND: **SERVE M0**

Berhasil mengantarkan M0

Berhasil memasak M2

=====

SALDO: 15000

Daftar Pesanan

Makanan	Durasi memasak	Ketahanan	Harga
---------	----------------	-----------	-------

M1	3	1	15000
M2	1	4	15000
M3	3	1	15000
M4	1	4	15000

M5	1	4	15000
M6	1	4	15000

Daftar Makanan yang sedang dimasak  
Makanan | Sisa durasi memasak

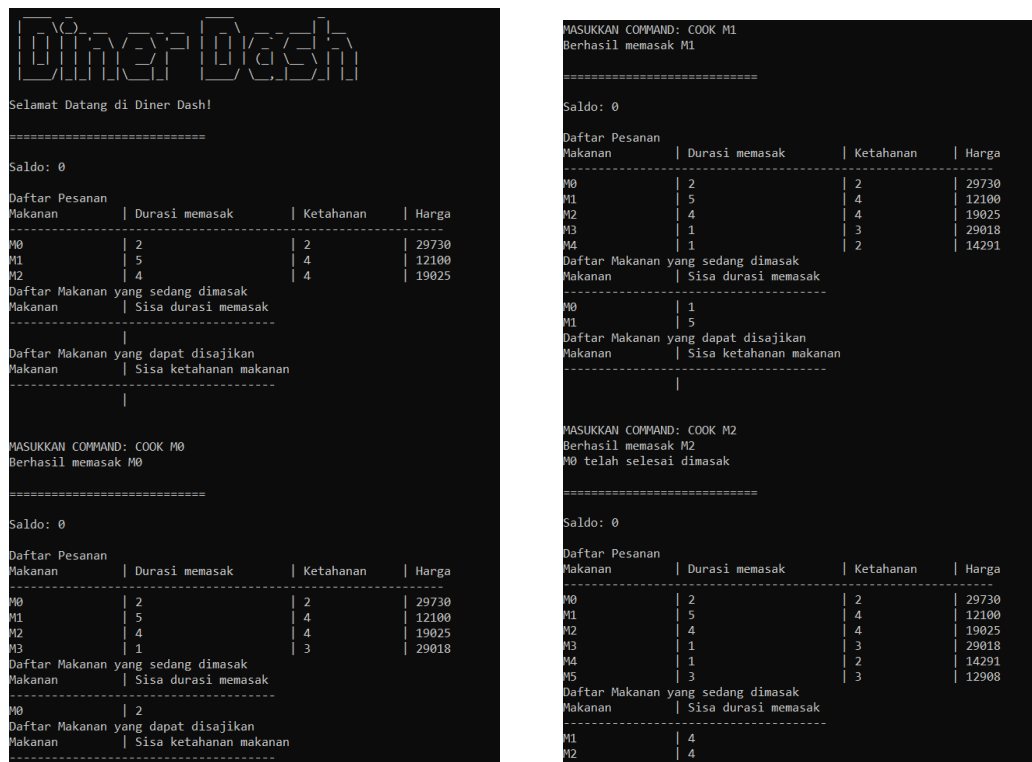
-----  
M1 | 1

Daftar Makanan yang dapat disajikan  
Makanan | Sisa ketahanan makanan

-----  
M2 | 4

MASUKKAN COMMAND: **SERVE M2**  
M2 belum dapat disajikan karena M1 belum selesai

### Data Test



Gambar 6.20 Tampilan BNMO saat user memainkan game Diner Dash pada program

## 6.16 ADT Array

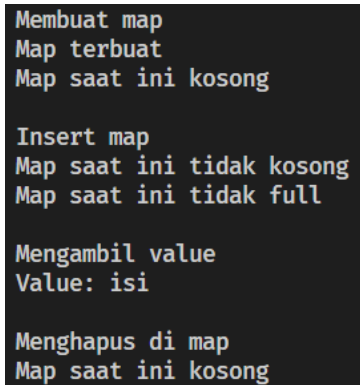
Deskripsi
<i>Test</i> ini digunakan untuk menguji ADT Array yang akan berguna di dalam program.
Hasil yang seharusnya diberikan
Membuat arraydin Arraydin telah terbuat [] Arraydin saat ini kosong Kapasitas arraydin: 10  Insert first arraydin [ISI] Insert last arraydin [ISI, ISI 2] Insert di idx 1 [ISI, ISI 3, ISI 2]  Copy arraydin [ISI, ISI 3, ISI 2] Mencari Index arraydin ISI 3 index isi 3: 1  Mengambil arraydin index 1 index 1: ISI 3  Isi arraydin sekarang: [ISI, ISI 2, ISI 3]  Delete index 1 arraydin [ISI, ISI 2]  Delete last arraydin [ISI]

Delete first arraydin []
<b>Data Test</b>
<pre> Membuat arraydin Arraydin telah terbuat [] Arraydin saat ini kosong Kapasitas arraydin: 10  Insert first arraydin [ISI] Insert last arraydin [ISI, ISI 2] Insert di idx 1 [ISI, ISI 3, ISI 2]  Copy arraydin [ISI, ISI 3, ISI 2] Mencari Index arraydin ISI 3 index isi 3: 1  Mengambil arraydin index 1 index 1: ISI 3  Isi arraydin sekarang: [ISI, ISI 3, ISI 2]  Delete index 1 arraydin [ISI, ISI 2]  Delete last arraydin [ISI]  Delete first arraydin [] </pre>
<i>Gambar 6.21 Tampilan BNMO saat user mencoba menjalankan ADT Array yang telah dibuat.</i>

## 6.17 ADT Mesin Karakter dan Mesin Kata

<b>Deskripsi</b>
<i>Test</i> ini digunakan untuk menguji apakah mesin kata dan mesin karakter yang dibuat dapat mengolah lebih dari 1 kata.
<b>Hasil yang seharusnya diberikan</b>
Halo Saya Iyal Halo Saya Iyal
<b>Data Test</b>
<pre> C:\C\TUBES-ALSTRUKDAT-KELOMPOK-10\program\ADT\mesinkarkata&gt;driver Halo Saya Iyal Halo Saya Iyal </pre>
<i>Gambar 6.22 Tampilan BNMO saat user mencoba memakai mesin kata dan mesin karakter</i>

## 6.18 ADT Map

Deskripsi
<i>Test</i> ini digunakan untuk ADT Map yang akan berguna di dalam program.
Hasil yang seharusnya diberikan
Membuat map Map terbuat Map saat ini kosong  Insert map Map saat ini tidak kosong Map saat ini tidak full  Mengambil value Value: isi  Menghapus di map Map saat ini kosong
Data Test
 <p>Membuat map Map terbuat Map saat ini kosong  Insert map Map saat ini tidak kosong Map saat ini tidak full  Mengambil value Value: isi  Menghapus di map Map saat ini kosong</p>

*Gambar 6.23 Tampilan BNMO saat user mencoba memakai ADT Map.*

## 6.19 ADT Queue

Deskripsi
-----------

*Test* ini digunakan untuk ADT Queue yang akan berguna di dalam program.

**Hasil yang seharusnya diberikan**

## Membuat queue

[ ]

## Queue terbuat

Queue saat ini kosong

Mengisi queue

[ISI]

Queue saat ini tidak kosong

Queue saat ini tidak full

Panjang queue sekarang: 1

Menghapus val pada q

[ ]

Val queue yang terambil: ISI

Mengisi queue sampai full

[ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI,  
ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, I  
SI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, IS  
I, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI  
, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI,  
ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI, ISI]

Queue saat ini full

## Data Test

[illegible]

Gambar 6.24: Tampilan BNMO saat *user* mencoba memakai ADT Queue.

## 7 Test Script

No.	Fitur yang Dites	Tujuan Testing	Langkah-Langkah Testing	Input Data Test	Hasil yang Diharapkan	Hasil yang Keluar
1	Fitur Welcome Page	Memeriksa apakah BNMO dapat dijalankan.	Melakukan <i>compile file</i> "main.c" di terminal lalu menjalankan <i>file</i> main.exe.	Data Test 6.1	BNMO berhasil menampilkan <i>opening message</i> di layar.	Sesuai harapan.
2	Fitur Start	Memeriksa apakah BNMO dapat membaca <i>file</i> konfigurasi.	Memasukkan <i>command</i> START.	Data Test 6.2	BNMO berhasil membaca <i>file</i> konfigurasi dan program dapat dijalankan.	Sesuai harapan.
3	Fitur Load	Memeriksa apakah BNMO dapat membaca <i>save file</i> .	Memasukkan <i>command</i> LOAD beserta nama <i>save file</i> -nya.	Data Test 6.3	BNMO berhasil membaca <i>save file</i> dan program dapat dijalankan.	Sesuai harapan.
4	Fitur Save	Memeriksa apakah BNMO dapat menyimpan <i>state game</i> saat ini.	Memasukkan <i>command</i> SAVE beserta nama <i>save file</i> -nya.	Data Test 6.4	BNMO berhasil menyimpan <i>save file</i> .	Sesuai harapan.
5	Fitur Create Game	Memeriksa apakah BNMO dapat menambahkan <i>game</i> baru pada daftar <i>game</i> .	Memasukkan <i>command</i> CREATE GAME, lalu memasukkan nama <i>game</i> yang ingin ditambahkan.	Data Test 6.5	BNMO berhasil menambahkan <i>game</i> sesuai masukan <i>user</i> jika <i>game</i> tersebut tidak ada di dalam daftar <i>game</i> .	Sesuai harapan.
6	Fitur List Game	Memeriksa apakah BNMO menampilkan daftar <i>game</i> yang disediakan sistem	Memasukkan <i>command</i> LIST GAME.	Data Test 6.6	BNMO berhasil menampilkan daftar <i>game</i> yang tersedia.	Sesuai harapan.
7	Fitur Delete Game	Menguji apakah BNMO dapat menghapus <i>game</i> dari daftar <i>game</i> yang dapat dihapus.	Memasukkan <i>command</i> DELETE GAME lalu memasukkan nomor <i>game</i> yang ingin dihapus.	Data Test 6.7	BNMO berhasil menghapus <i>game</i> , <i>game</i> yang dapat dihapus. Serta, menampilkan pesan jika <i>game</i> gagal dihapus	Sesuai harapan.
8	Fitur Queue Game	Menguji apakah BNMO dapat mendaftarkan permainan yang valid ke dalam listMenguji apakah BNMO dapat memasukkan <i>game</i> ke dalam queue.	Memasukkan <i>command</i> QUEUE GAME, setelah itu memasukkan nomor <i>game</i> yang ingin ditambahkan ke dalam queue.	Data Test 6.8	BNMO berhasil menambahkan <i>game</i> yang valid ke dalam daftar antrian, dan menampilkan pesan jika permainan tidak valid.BNMO berhasil menambahkan <i>game</i> ke queue	Sesuai harapan.

9	Fitur Play Game	Menguji apakah BNMO dapat memainkan sebuah game, dan main sesuai dengan antrian game.	Memasukkan <i>command</i> PLAY GAME.	Data Test 6.9	BNMO berhasil memainkan game yang ada pada spesifikasi game. Jika tidak ada dalam spesifikasi maka akan ditampilkan pesan game tidak dapat dimainkan.	Sesuai harapan.
10	Fitur Skip Game	Menguji apakah BNMO dapat melewati permainan.	Memasukkan <i>command</i> SKIP GAME beserta jumlah permainan yang ingin dilewatkan.	Data Test 6.10	BNMO berhasil melewati permainan sebanyak N, dan Jika masih ada game dalam list. Maka BNMO memainkan Game yang ada di list selanjutnya. Jika tidak maka akan ditampilkan pesan bahwa Tidak ada permainan lagi dalam daftar game.	Sesuai harapan.
11	Fitur Quit	Menguji apakah BNMO dapat keluar dari program, serta menjadikan List dalam Queue game hilang	Memasukkan Command QUIT.	Data Test 6.11	BNMO menyimpan terlebih dahulu <i>current state</i> dari BNMO lalu keluar dari game serta menampilkan <i>end page</i> .	Sesuai harapan.
12	Fitur Help	Menguji apakah BNMO dapat menampilkan list command dalam BNMO	Memasukkan Command HELP	Data Test 6.12	BNMO menampilkan daftar-daftar fitur yang disediakan beserta deskripsi fungsinya.	Sesuai harapan.
13	Fitur Command Lain	Menguji BNMO jika inputan command tidak valid	Memasukkan command yang tidak ada di pada list command	Data Test 6.13	BNMO menampilkan pesan bahwa masukan <i>command</i> tidak valid lalu meminta masukan <i>command</i> kepada <i>user</i> .	Sesuai harapan.
14	RNG	Menguji Permainan RNG apakah sesuai dengan spesifikasi, serta mampu menghasilkan outcome berupa <i>random number</i>	Memasukkan <i>command</i> PLAY GAME ketika game RNG telah menjadi urutan pertama dalam list game atau masukan SKIP GAME dan game selanjutnya adalah RNG	Data Test 6.14	BNMO mampu memainkan <i>Game</i> RNG.	Sesuai harapan.
15	Diner Dash	Menguji Permainan Diner Dash apakah sesuai dengan spesifikasi	Memasukkan <i>command</i> PLAY GAME ketika game Diner Dash telah menjadi urutan pertama dalam list game atau masukan SKIP GAME dan	Data Test 6.15	BNMO memainkan <i>game</i> Diner Dash.	Sesuai harapan.



			game selanjutnya adalah Diner Dash			
16	ADT Array	Menguji ADT array apakah berfungsi dengan benar	Melakukan testing di driver array.	Data Test 6.16	Menghasilkan kalimat yang seharusnya muncul di terminal.	Sesuai harapan.
17	ADT Mesin Karakter & Mesin Kata	Menguji ADT mesin karakter dan mesin kata apakah berfungsi dengan benar	Melakukan testing di driver mesin karakter dan mesin kata.	Data Test 6.17	Menghasilkan kalimat yang seharusnya muncul di terminal.	Sesuai harapan.
18	ADT Map	Menguji ADT map apakah berfungsi dengan benar	Melakukan testing di driver map.	Data Test 6.18	Menghasilkan kalimat yang seharusnya muncul di terminal.	Sesuai harapan.
19	ADT Queue	Menguji ADT queue apakah berfungsi dengan benar	Melakukan testing di driver queue.	Data Test 6.19	Menghasilkan kalimat yang seharusnya muncul di terminal.	Sesuai harapan.

## 8 Pembagian Kerja dalam Kelompok

No	Fitur/ ADT	NIM Coder	NIM Tester
1	ADT Array	18221043, 18221055	18221127, 18221109, 18221113
2	ADT Mesin Karakter dan Mesin Kata	18221043, 18221055	18221127, 18221109, 18221113
3	ADT Queue	18221043, 18221055	18221127, 18221109, 18221113
4	Main Menu	18221043	18221127, 18221109, 18221113, 18221055
5	Start	18221043	18221127, 18221109, 18221113, 18221055
6	Load	18221043	18221127, 18221109, 18221113, 18221055
7	Save	18221043	18221127, 18221109, 18221113, 18221055
8	Create Game	18221127	18221043, 18221109, 18221113, 18221055
9	List Game	18221127	18221043, 18221109, 18221113, 18221055

10	Delete Game	18221127	18221043, 18221109, 18221113, 18221055
11	Queue Game	18221109	18221043, 18221127, 18221113, 18221055
12	Play Game	18221109	18221043, 18221127, 18221113, 18221055
13	Skip Game	18221109	18221043, 18221127, 18221113, 18221055
14	Quit	18221113	18221043, 18221127, 18221109, 18221055
15	Help	18221113	18221043, 18221127, 18221109, 18221055
16	Command Lain	18221113	18221043, 18221127, 18221109, 18221055
17	Konfigurasi System	18221043	18221127, 18221109, 18221113, 18221055
18	RNG	18221055	18221127, 18221109, 18221113, 18221043
19	Diner Dash	18221055	18221127, 18221109, 18221113, 18221043
20	Game Tambahan	18221055	18221127, 18221109, 18221113, 18221043

## 9 Lampiran

### 9.1 Deskripsi Tugas Besar 1

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.

#### System Mechanics

##### 1. About the System

BNMO merupakan suatu robot game console yang dapat menjalankan permainan. BNMO memiliki beberapa fitur utama, yaitu Memainkan game, Menambahkan game, Menghapus game, dan Mengurutkan game yang akan dimainkan

##### 2. Main Menu

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

### 3. Command

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

#### a. START

START merupakan salah satu command yang dimasukkan pertama kali oleh pemain ke BNMO. Setelah menekan Enter, dibaca *file* konfigurasi *default* yang berisi list game yang dapat dimainkan.

#### b. LOAD <filename>

LOAD merupakan salah satu command yang dimasukkan pertama kali oleh pemain ke BNMO. Memiliki satu argumen yaitu filename yang merepresentasikan suatu *save file* yang ingin dibuka. Setelah menekan Enter, akan dibaca *save file* <filename> yang berisi list game yang dapat dimainkan, histori dan scoreboard game, lebih detailnya bisa dilihat pada Konfigurasi Sistem.

#### c. SAVE <filename>

SAVE merupakan command yang digunakan untuk menyimpan state game pemain saat ini ke dalam suatu *file*. Command SAVE memiliki satu argumen yang merepresentasikan nama *file* yang akan disimpan pada disk.

#### d. CREATE GAME

CREATE GAME merupakan command yang digunakan untuk menambahkan game baru pada daftar game. Spesifikasi game yang dibuat dapat dilihat pada section Spesifikasi Game

#### e. LIST GAME

LIST GAME merupakan command yang digunakan untuk menampilkan daftar game yang disediakan oleh sistem.

#### f. DELETE GAME

DELETE GAME merupakan command yang digunakan untuk menghapus sebuah game dari daftar game. Adapun aturan penghapusan game adalah:

- Game yang dapat dihapus hanya game yang dibuat secara custom oleh pengguna.
- 5 game pertama pada *file* konfigurasi tidak dapat dihapus.
- Game yang saat itu terdapat di dalam queue game tidak dapat dihapus.

#### g. QUEUE GAME

QUEUE GAME merupakan command yang digunakan untuk mendaftarkan permainan kedalam list. List dalam queue akan hilang ketika pemain menjalankan command **QUIT**.

#### h. PLAY GAME

PLAY GAME merupakan command yang digunakan untuk memainkan sebuah permainan. Game yang dimainkan adalah game dengan urutan pertama di antrian game. Ketika salah satu permainan dimulai, sistem akan menjalankan game sesuai pada section Spesifikasi Game. Permainan selain yang dispesifikasikan pada Spesifikasi Game akan menampilkan pesan bahwa game tidak dapat dimainkan.

i. SKIP GAME <n>

SKIP GAME merupakan command yang digunakan untuk melewati permainan sebanyak n.

j. QUIT

Keluar dari program.

k. HELP

Bantuan command-command yang disebutkan di atas. Tampilan dan kata-kata dibebaskan.




l. COMMAND LAIN



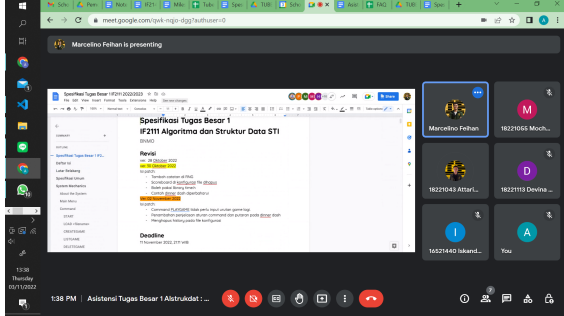

Command-command lain selain yang disebutkan diatas tidak valid.

Keluar dari program.



## 9.2 Notulen Rapat




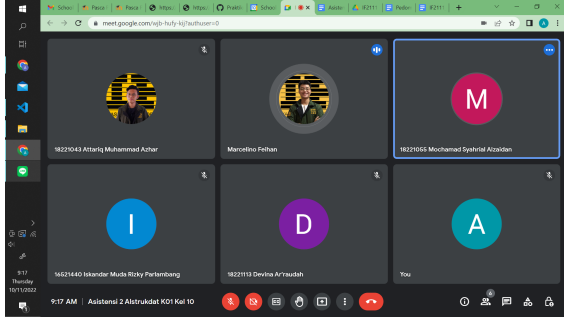
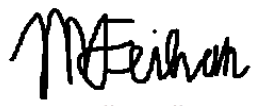
### A. Asistensi I

Tanggal : Kamis, 3 November 2022	<b>Catatan Asistensi:</b> <b>Penjabaran Garis Besar Spesifikasi</b> <ol style="list-style-type: none"><li><b>CREATE GAME</b> -&gt; wajib di update ke konfigurasi.txt</li><li><b>PLAYGAME</b> -&gt; pakai queue, harus urutan mainnya kalo ga di skip</li><li><b>SKIP GAME</b> -&gt; pakai dequeue</li><li><b>DINER DASH</b> -&gt; Daftar pesanan bisa pake adt list/queue.</li></ol> <b>SESI QnA</b> Q : Pada pembacaan load dan save, kan harus pakai mesin karakter dan mesin kata. Itu gimana ya kak buat EOP nya? A : Oh itu bisa pakai blank aja.  Q : Cara bedain blank di spasi nama fungsi ke nama file sama di akhir file itu gimana kak? A : Kata aku cek aja karakter abis blank. Kalo masih ada karakter, berarti lanjut. Kalau ternyata blank lagi, yaudah stop.  Q : Kakak perlu masuk ke repository kita? A : Perlu.
Tempat: <a href="#">GoogleMeet</a>	
<b>Kehadiran Anggota Kelompok:</b> No NIM Tanda tangan  1 18221055 	
2 18221109 	
3 18221113 	

<p>4 18221043 </p> <p>5 18221127 </p>	<p>Q : Buat di Dinner Dash itu kata kayak baiknya pake tiga Queue ya. Kalo cuma dua gimana? A : Gapapa yang menurut kalian enak gimana.</p> <p><b>TIPS</b></p> <ul style="list-style-type: none"> <li>• Bagi tugas berdasarkan ADT dan Fungsinya</li> </ul>
 <p>Gambar 9.1 Screenshot asistensi I</p>	<p><b>Tanda Tangan Asisten:</b></p> 

## B. Asistensi II

<p><b>Tanggal : 10 November 2022</b></p>	<p><b>Catatan Asistensi:</b></p>
<p><b>Tempat : Google Meet</b></p>	<p><b>Sesi QnA</b></p>
<p><b>Kehadiran Anggota Kelompok:</b></p> <p>No NIM Tanda tangan</p> <p>1 18221055 </p> <p>2 18221109 </p> <p>3 18221113</p>	<p>Q: Ada folder bin. Aku tanya ke temen aku masih bingung itu gimana? A: Makefile itu sebenarnya isinya buat gcc dst itu biar gampang nge-run nya. Tetep dibuat aja.</p> <p>Q: Kan ada Data Test di laporan itu. Kami itu udah bikin driver untuk ADT. Itu termasuk Data Test ga? A: Iya termasuk.</p> <p>Q: Kalo nge-load file txt yang ga ada di folder data, bakal langsung ke end. Itu gimana? A: Cara nge-handle dibebaskan. Kalo aku prefer ditangani aja, misal “error” atau “nama tidak sesuai” gitu.</p> <p>Q: Kalau di save itu aku bikin fopen sama filenya. Kalo misalnya save terus input nama filenya gaada, jadinya nanti dia bikin file txt sesuai inputan gitu. Gitu gapapa ga ya? A: Gapapa harusnya. Ntar ditanyain ke asisten</p> <p>Q: Kalau pake library tambahan buat UI/UX boleh ga? A: Jangan, pake 4 library itu aja.</p>

 4 18221043  5 18221127 	<p>Q: Oh Demo itu nanti kita pake device sendiri ya? A: Belum tau si, jaga-jaga saja.</p> <p>Q: Kan tiap ADT perlu dibuat drivernya ya kak. Kalo buat Diner Dash kan aku buat ADT yang agak beda. Itu kira-kira masih perlu ga ya? A: Tetep bikin aja buat keperluan testing dari kita (asisten) juga.</p> <p>Q: Di laporan, Game bonus bisa masuk ke fitur tambahan ga? A: Bisa</p> <p>Q: Kalau ngesave file namanay beda dari yg di load boleh ga, misal aku load konfigurasi.txt, mainin, terus aku simpan sebagai save.txt gitu? A: Boleh</p> <p>Q: Perlu bikin driver buat adt queue dinner dash ga? atau yaudah dibiarin di main prog krn udh ada game dinner dash JUGA A: Main program saja</p> <p>TIPS: Jaga-jaga aja, asisten belum ngasih tau untuk Demo nya kaya gimana. Saran aku, salah satu teman kalian install VM terus install Ubuntu buat Demo. Ini pengalaman aku waktu dulu. Dulu itu banyak yang kendala di mesinkata dan mesinkarakter.</p>
 <p>Gambar 9.2 Screenshot asistensi II</p>	<p><b>Tanda Tangan Asisten:</b></p> 

### C.Meeting Kelompok

No	Hari/Tanggal	Notulensi	Kehadiran Anggota Kelompok
1	Sabtu, 29 Oktober 2022	Diskusi terkait spesifikasi dan ADT program, serta pembagian kerja, dan	1. Attariq Muhammad Azhar

		pembuatan drive, github kelompok	2. Mochamad Syahril A 3. Iskandar Muda Rizky Parlambang 4. Devina Ar'raudah 5. Ariffudin Achmad Subagja
2.	Kamis, 3 November 2022	Diskusi terkait ADT serta main program	
3.	Minggu, 6 November 2022	Diskusi terkait fungsi masing-masing dan progress report terkait fitur-fitur BNMO.	
4	Selasa, 8 November 2022	Diskusi terkait Debugging dan penyatuan program utama	

### 9.3 Log Activity Anggota Kelompok

No	Jadwal	Keterangan
1	Sabtu, 29 Oktober 2022	Meeting Kelompok Online
2	Kamis, 3 November 2022	Asistensi 1
3	Kamis, 3 November 2022	Meeting Kelompok Hybrid
3	Minggu, 6 November 2022	Meeting Kelompok Offline
4	Senin, 7 November 2022	Deadline pengerjaan fungsi masing-masing
5	Selasa, 8 November 2022	Meeting Kelompok Online
6	Kamis, 10 November 2022	Asistensi 2
7	Kamis, 10 November 2022	Penyusunan Laporan
8	Jumat, 11 November 2022	Pengumpulan Laporan
9	TBD	Demo Program