**LAPORAN TUGAS BESAR**

**IF211**1 **Algoritma dan Struktur Data**

BNMO

Dipersiapkan oleh:

Kelompok 12

| Ivan Aldy Ganesen | 18221045 |
| --- | --- |
| Muhammad Shulhan | 18221051 |
| Timothy Subekti | 18221063 |
| Ken Azizan | 18221107 |
| Nicholas | 18221165 |

Sekolah Teknik Elektro dan Informatika - Institut Teknologi Bandung

Jl. Ganesha 10, Bandung 40132

|  | **Sekolah Teknik Elektro dan Informatika ITB** | **Nomor Dokumen** | | **Halaman** |
| --- | --- | --- | --- | --- |
| *IF2111-TB1-12* | | *33* |
| *Revisi* | *1* | *11 November 2022* |

**Daftar Isi**

[**1 Ringkasan**](#_heading=h.gjdgxs) **3**

[**2 Penjelasan Tambahan Spesifikasi Tugas**](#_heading=h.e9c3z93vlwbz) **5**

[2.1 Tambahan pada CREATE GAME](#_heading=h.khfp86ffp2zu) 5

[2.2 Tambahan pada PLAY GAME dan SKIP GAME](#_heading=h.ex8c2051d7cd) 5

[2.3 Tambahan Bonus Game (Mole)](#_heading=h.u1zho21v6uj6) 5

[**3 Struktur Data (ADT)**](#_heading=h.jrjwh0cgopj2) **5**

[3.1 ADT Array Statis](#_heading=h.6klhdhbgbl9i) 5

[3.2 ADT Array Dinamis](#_heading=h.wh98tyx8xbvl) 7

[3.3 ADT Mesin Karakter](#_heading=h.ihrhtm6vv4e3) 8

[3.4 ADT Mesin Kata](#_heading=h.yhj8wcrzr60a) 9

[3.5 ADT Queue Versi Integer](#_heading=h.t2hwrbpi881x) 10

[3.6 ADT Queue Versi String](#_heading=h.ih2c65j2ibkr) 11

[**4 Program Utama**](#_heading=h.7nyk067z15wn) **12**

[**5 Algoritma-Algoritma Menarik**](#_heading=h.188zftyb46) **13**

[5.1 Algoritma Time Delay](#_heading=h.a5ci6t2uoqqq) 13

[5.2 Algoritma Pembacaan dan penulisan File](#_heading=h.8534ul3xxdwo) 13

[5.3 Algoritma Random integer](#_heading=h.fe7ktq2enzn1) 13

[**6 Data Test**](#_heading=h.4urrbrx0y8po) **13**

[6.1 Data Test 1 (kompilasi program)](#_heading=h.a1n8truuckmo) 13

[6.2 Data Test 2 (START)](#_heading=h.q670pwhm35wj) 14

[6.3 Data Test 3 (LOAD)](#_heading=h.ijyi87h7aqah) 14

[6.4 Data Test 4 (SAVE)](#_heading=h.3hvxm7762yc8) 15

[6.5 Data Test 5 (CREATE GAME)](#_heading=h.sdomr2mzlzdp) 16

[6.6 Data Test 6 (LIST GAME)](#_heading=h.a60g8mi89yhw) 17

[6.7 Data Test 7 (DELETE GAME)](#_heading=h.490j97izdasc) 17

[6.8 Data Test 8 (QUEUE GAME)](#_heading=h.5st5awwhamhm) 18

[6.9 Data Test 9 (PLAY GAME)](#_heading=h.5ieupiego35o) 20

[6.10 Data Test 10 (SKIP GAME)](#_heading=h.1hccuzqz1kz0) 21

[6.11 Data Test 11 (QUIT)](#_heading=h.q8yi6351lu5r) 22

[6.12 Data Test 12 (HELP)](#_heading=h.i46siwv0eczr) 22

[**7 Test Script**](#_heading=h.nwgqtnxogm8g) **23**

[**8 Pembagian Kerja dalam Kelompok**](#_heading=h.ginrh4sb85k7) **24**

[**9 Lampiran**](#_heading=h.2jxsxqh) **25**

[9.1 Deskripsi Tugas Besar 1](#_heading=h.m9ti09984sw7) 25

[9.2 Notulen Rapat](#_heading=h.udz3vpwipwqb) 27

[9.3 Log Activity Anggota Kelompok](#_heading=h.3kfe5e402hry) 27

[9.4 <Lain-Lain>](#_heading=h.6ftno6y6ix1z) 31

# **Ringkasan**

Laporan ini dibuat dalam rangka pengerjaan Tugas Besar IF2111 Algoritma dan Struktur Data. Tugas ini bertujuan untuk membuat sebuah program dengan bahasa pemrograman C dan mengimplementasikan materi yang diajarkan pada kuliah. Program ini berbasis pada CLI (command-line interface). BNMO adalah game console yang memiliki beberapa fitur diantaranya adalah memainkan game yang tersedia, membuat game baru pada console BNMO, menghapus game dari BNMO, dan membuat urutan game yang akan dimainkan.

Laporan ini berisikan spesifikasi program yang dibuat, struktur data ADT dan penggunaannya dalam membantu menyelesaikan pembuatan program, algoritma program utama, algoritma - algoritma yang menarik, pembagian tugas di dalam kelompok, dan lampiran-lampiran dalam pengerjaan tugas besar ini.

Tugas besar ini merupakan implementasi dari bermacam - macam ADT yang telah dipelajari pada kuliah IF2111 Algoritma dan Struktur Data STI. Implementasi tersebut direpresentasikan dalam pembuatan fitur-fitur yang ada pada BNMO game console. Kesimpulan yang didapat dari tugas besar ini adalah ADT digunakan untuk memecahkan masalah dalam mengolah data pada sebuah program sehingga dapat tertata secara efektif dan efisien agar program dapat tersusun dengan baik.

# **Penjelasan Tambahan Spesifikasi Tugas**

## Tambahan pada CREATE GAME

Program melakukan validasi input, yaitu memeriksa apakah game dengan nama yang diinput sudah ada atau belum. Jika nama game sudah ada, program akan meminta user untuk melakukan input ulang. Tambahan ini dilakukan agar penambahan game dipermudah pada jumlah game yang besar.

## Tambahan pada PLAY GAME dan SKIP GAME

Apabila PLAY GAME atau SKIP GAME dijalankan ketika tidak ada game dalam antrian, maka program akan mengirimkan pesan bahwa tidak ada game di dalam antrian.

## Tambahan Bonus Game (Mole)

Mole adalah game dimana bermain sebagai seorang petani dengan lahan berluas 5x5. Tapi pada lahan tersebut ada seekor hama mole yang berada di bawah tanah dalam lahan pemain. Pemain perlu mencari mole tersebut dengan menggali lahan 5x5 dengan input “X Y” dan pemain tidak bisa menggali grid yang sudah digali dan begitu juga dengan mole yang tidak bisa berada di grid yang sudah digali. Skor yang didapatkan pemain berdasarkan seberapa cepat pemain menemukan mole.

# **Struktur Data (ADT)**

Dalam membuat program ini, kami membuat ADT-ADT yang digunakan dalam menyelesaikan program utama. ADT yang dimaksud adalah ADT Array Dinamik versi string, ADT Queue versi integer, ADT Queue versi string, ADT Mesin Karakter, dan ADT Mesin Kata. Seluruh ADT yang telah disebutkan sudah sesuai dengan tujuan utamanya, yaitu untuk menunjang struktur program utama. Contohnya, penggunaan ADT Mesin Karakter dan ADT Mesin Kata digunakan untuk membaca file konfigurasi pada program utama dan membaca masukan command dari user. ADT Queue versi integer digunakan dalam pembuatan game Diner Dash. ADT Queue versi string digunakan untuk keperluan antrian game dan untuk melakukan skip pada antrian game tersebut.

## ADT Array Statis

Sketsa Struktur Data ADT Array Dinamis Versi Integer

ADT Array Statis memiliki DataType ArrayStat sebagai array statik yang dapat diakses. Adapun komponen yang terdapat dalam ArrayStat:

* ElType Ar[100]

ElType memiliki tipe integer sehingga arr adalah array of integer dengan kapasitas 100.

* int Neff

Neff adalah elemen efektif dalam array yaitu berapa elemen yang terisi dalam array Arr. Neff juga merupakan nomor indeks terakhir yang diisi jika ditambahi satu.

ADT ini memiliki beberapa primitif:

* StatArray CreateStatArray()

Digunakan untuk membuat sebuah StatArray kosong dengan Neff 0.

* void DealokasiStatArray(StatArray \*array)

Digunakan untuk dealokasi arr pada StatArray.

* boolean IsEmptyStat(StatArray array)

Mengirimkan nilai true jika StatArray array kosong yaitu dengan Neff 0.

* int LengthStat(StatArray array)

Mengembalikan panjang dari array yang terisi yaitu Neff.

* ElType GetStat(StatArray array,IdxType i)

Mengembalikan elemen array dengan indeks i.

* void InsertStat(StatArray \*array,ElType elmt,IdxType i)

Menambah elemen elmt pada StatArray array pada indeks i

* void InsertFStat(StatArray \*array,ElType elmt)

Menambah elemen Elmt pada awal StatArray array

* void InsertLStat(StatArray \*array,ElType elmt)

Menambah elemen elmt pada akhir StatArray array.

* void DeleteStat(StatArray \*array,IdxType i)

Menghapus elemen pada indeks i pada StatArray array

* void DeleteFStat(StatArray \*array)

Menghapus elemen pertama pada StatArray array

* void DeleteLStat(StatArray \*array)

Menghapus elemen pada akhir di StatArray array

* void ShowStatArray(StatArray array)

Menampilkan semua elemen array dengan format [Elmt(1),Elmt(2),...,Elmt(LengthStat(array)-1)]

* void ReverseStatArray(StatArray array)

Membalikkan elemen elemen pada ArrayStat array

* StatArray CopyStatArray(StatArray array)

Mengembalikan duplikat dari StatArray array yang diberikan

* IdxType FindStatArray(StatArray array,Eltype elmt)

Mengembalikan Indeks dari suatu elemen pada StatArray array.

Persoalan yang diselesaikan

Array Statis dengan ElType integer ini digunakan dalam merancang dan mengoperasikan game Diner Dash. Array Statis ini digunakan untuk menyimpan durasi dan ketahanan pada antrian, masakan dan penyajian.

Alasan pemilihan

Array Statis paling cocok digunakan pada Diner Dash karena pada masakan dan penyajian dapat diambil elemen yang bukan head ketika durasi atau ketahanan 0 sehingga kurang cocok untuk menggunakan queue yang penghapusan data utamanya menggunakan dequeue.

## ADT Array Dinamis

Sketsa Struktur Data ADT Array Dinamis Versi String

ADT ini memiliki ArrayDyn sebagai tipe array dinamik yang akan dipakai secara keseluruhan. Adapun komponen dari ArrayDyn:

* TypeEl \*Ar

Karena TypeEl bertipe pointer to char, maka isi array adalah string.

* int Kapasitas

Kapasitas maksimum array yang dapat diisi oleh elemen. Biasanya bernilai StartSize (100) tetapi dapat berubah apabila array mengalami realokasi.

* int Neff

Banyaknya elemen efektif dalam array. Dalam array yang rata kiri ini, Neff juga berarti satu indeks setelah indeks terakhir yang diisi elemen.

ADT ini memiliki beberapa primitif:

* ArrayDyn CreateStrArrayDyn()

Digunakan untuk membuat sebuah array baru yang teralokasikan secara dinamis, dengan kapasitas senilai StartSize dan Neff bernilai 0.

* void DealokasiStrArrayDyn(ArrayDyn \*array)

Digunakan untuk mendealokasikan array yang sebelumnya terdefinisi.

* boolean IsStrEmpty(ArrayDyn array)

Mengirimkan nilai true apabila array kosong, dan false jika tidak kosong.

* int StrLength(ArrayDyn array)

Mengembalikan nilai berupa banyaknya elemen efektif yang terdapat di array tersebut.

* TypeEl GetStr(ArrayDyn array, TypeIdx i)

Mengembalikan nilai elemen berindeks i dalam array.

* int GetStrCapacity(ArrayDyn array)

Mengembalikan kapasitas dari array.

* void InsertStrIn(ArrayDyn \*array, TypeEl elmt, TypeIdx i)

Memasukkan elemen baru (elmt) ke dalam array di indeks ke-i.

* void InsertStrFirst(ArrayDyn \*array, TypeEl elmt)

Memasukkan elemen baru (elmt) di awal array (indeks ke-0).

* void InsertStrLast(ArrayDyn \*array, TypeEl elmt)

Memasukkan elemen baru (elmt) di akhir array (indeks ke-Neff).

* void DeleteStrIn(ArrayDyn \*array, TypeIdx i)

Menghapus elemen yang berada dalam indeks ke-i di array.

* void DeleteStrFirst(ArrayDyn \*array)

Menghapus elemen pertama dalam array (indeks ke-0).

* void DeleteStrLast(ArrayDyn \*array)

Menghapus elemen terakhir dalam array (indeks ke - Neff-1).

* void ShowStrArrayDyn(ArrayDyn array)

Menampilkan array ke layar dengan format [elemen-1,elemen-2,...,elemen-n].

* void ReverseStrArrayDyn(ArrayDyn \*array)

Mengubah urutan array menjadi terbalik (reversed).

* ArrayDyn CopyStrArrayDyn(ArrayDyn array)

Menyalin array ke suatu array yang lain.

* TypeIdx FindStrArrayDyn(ArrayDyn array, TypeEl elmt)

Mencari indeks pertama kemunculan elmt dalam array. Jika tidak ditemukan, indeks bernilai -1.

Persoalan yang Diselesaikan

ADT Array Dinamis versi string ini yang paling sering digunakan dan menjadi fondasi di program ini. ADT ini menangani list game yang terdaftar di BNMO, sehingga user dapat melihat game apa saja yang tersedia; mereka dapat menambahkan/menghapus game yang ada (LIST GAME, ADD GAME, DELETE GAME).

Alasan Pemilihan

Untuk menunjang penuh keberjalanan BNMO, maka dia harus bisa menampilkan daftar game agar bisa dilihat oleh user. Disinilah array dinamik versi string ini dibutuhkan; agar user dapat melihat game-game apa saja yang tersedia.

## ADT Mesin Karakter

Sketsa Struktur Data ADT Mesin Karakter

ADT ini memiliki beberapa Primitif:

* State

1. extern char CC

2. extern boolean EOP

* void START()

Berfungsi untuk membaca file yang dimasukkan. Karakter pertama merupakan pita yang posisinya di jendela.

* void STARTINPUT()

Digunakan untuk membaca command yang dimasukan. Karakter pertama merupakan pita yang posisinya di jendela.

* void ADVSTOPMARK()

Berfungsi untuk memajukan karakter apabila karakter bukan MARK.

* void ADV()

Berfungsi untuk memajukan karakter apabila karakter bukan ENTER.

* void ADVFILE()

Berfungsi untuk memajukan karakter apabila karakter bukan End-Of-File.

* char GetCC()

Berfungsi untuk mengirimkan CC.

* boolean IsEOP()

Mengirimkan true jika CC sama dengan MARK, mengirimkan false jika tidak.

* boolean IsNewLine()

Mengirimkan true jika CC sama dengan ‘\n’; false jika sebaliknya.

* boolean IsFeof()

Mengirimkan true jika CC sama dengan End-Of-File; false jika sebaliknya.

* boolean IsPitaNull()

Mengirimkan true jika pita sama dengan Null; false jika sebaliknya..

Persoalan yang diselesaikan

ADT ini berfungsi menyelesaikan persoalan ADT Mesin Kata.

Alasan pemilihan

ADT mesin kata memerlukan ADT mesin karakter dalam fungsi-fungsi *prototype*-nya.

Implementasi sebagai "mesinkarakter.h", "mesinkarakter.c", dan driver.

## ADT Mesin Kata

Sketsa Struktur Data ADT Mesin Kata

ADT ini memiliki beberapa primitif berupa:

* State

1. extern boolean EndWord

2. extern Word CWord

3. extern Word CCommand

* void IgnoreNewLine()

Mengabaikan satu atau beberapa newline.

* void IgnoreBlanks()

Mengabaikan satu atau beberapa blanks.

* void IgnoreDot()

Mengabaikan satu atau beberapa titik.

* void STARTFILE(char \*FileName)

Berfungsi untuk membaca file yang dimasukan dalam bentuk pita karakter. Word pertama diakuisisi dan disimpan pada buffer CWord.

* void INPUT()

Berfungsi untuk membaca command yang dimasukan dalam bentuk pita karakter. CCommand pertama diakuisisi dan disimpan pada buffer CCommand.

* void ADVWORD()

Berfungsi untuk memajukan pita karakter dan mengakuisisi Word hingga bertemu MARK.

* void ADVLINEFILE()

Berfungsi untuk memajukan pita karakter dan mengakuisisi Word hingga bertemu ENTER.

* void ADVLINE()

Berfungsi untuk memajukan pita karakter dan mengakuisisi Word hingga bertemu ENTER.

* void CopyWord()

Berfungsi untuk menyalin kata yang dibaca ke dalam buffer CWord hingga bertemu MARK atau BLANK.

* void CopyLineFile()

Berfungsi untuk menyalin kata yang dibaca ke dalam buffer CWord hingga bertemu MARK , '\n', atau End-Of-File .

* void CopyLine()

Berfungsi untuk menyalin kata yang dibaca ke dalam buffer CWord hingga bertemu MARK atau '\n'.

* boolean IsBlank()

Mengirimkan true jika CC sama dengan " "; mengirimkan false jika tidak.

Persoalan yang diselesaikan

ADT ini berfungsi untuk membaca informasi game dari file eksternal. Hal ini digunakan pada fitur *start*, *load*, dan *save.* Membaca command yang diinput oleh user.

Alasan pemilihan

Pembacaan informasi dari file eksternal perlu diakuisisi kata satu per satu dibentuk menjadi string. Pembacaan input command yang dimasukan oleh user juga perlu disimpan pada bentuk string.

Implementasi sebagai "mesinkata.h", "mesinkata.c", dan driver.

## 

## ADT Queue Versi Integer

Sketsa Struktur Data ADT Queue Versi Integer

ADT ini memiliki Queue sebagai tipe queue yang akan dipakai secara keseluruhan.

Adapun komponen dari Queue:

* ElmtType buffer[InitSize]

Karena ElmtType bertipe integer, maka isi queue adalah integer. Ukuran awalnya adalah InitSize, tetapi dapat berubah apabila queue sudah penuh dengan realokasi memori.

* int IdxHead

Indeks yang menandakan letak elemen pertama dalam queue.

* int IdxTail

Indeks yang menandakan letak elemen terakhir dalam queue.

Queue yang dipakai bersifat *circular buffer*, dimana IdxHead tidak harus berada di indeks ke-0, melainkan dapat berada di mana saja.

ADT ini memiliki beberapa primitif:

* void CreateQueue(Queue \*q)

Menciptakan sebuah queue kosong, dengan ketentuan IdxHead dan IdxTail bernilai IDX\_UNDEF (-1).

* boolean isEmpty(Queue q)

Mengirimkan nilai true apabila queue kosong; false jika sebaliknya.

* boolean isFull(Queue q)

Mengirimkan nilai true apabila queue penuh, yaitu apabila IdxTail berada di belakang IdxHead.

* int length(Queue q)

Mengembalikan panjang dari queue, yaitu banyaknya elemen yang berada di queue.

* void enqueue(Queue \*q, ElmtType var)

Memasukkan elemen var ke bagian paling belakang queue, dimana queue pasti tidak penuh. Penambahan elemen ini mengikuti prinsip *First In First Out*.

* void dequeue(Queue \*q, ElmtType \*var)

Mengambil elemen di bagian paling depan queue (IdxHead) sebagai var , lalu IdxHead dimundurkan sedemikian rupa sehingga mantan IdxHead tadi sudah terhapus. Penghapusan elemen ini mengikuti prinsip *First In First Out*.

* void DisplayQueue(Queue q)

Menampilkan queue ke layar dengan format setiap elemen yang dipisahkan oleh newline, mulai dari elemen pertama (IdxHead) sampai elemen terakhir (IdxTail).

Persoalan yang Diselesaikan

ADT Queue versi integer ini erat kaitannya dengan game Diner Dash, dimana nantinya pada antrian customer yang memesan makanan.

Alasan Pemilihan

ADT Queue ini adalah representasi yang paling cocok terkait Diner Dash, karena

diperlukan fungsi dequeue pada ADT queue terutama pada proses penyajian di mana prosesnya harus dilakukan satu per satu dari head.

## ADT Queue Versi String

Sketsa Struktur Data ADT Queue Versi String

ADT ini memiliki QueueStr sebagai tipe queue yang akan dipakai secara keseluruhan.

Adapun komponen dari QueueStr:

* ElemType buffer[InitSize]

Karena ElemType bertipe pointer to char, maka isi queue adalah string. Ukuran awalnya adalah InitSize, tetapi dapat berubah apabila queue sudah penuh dengan realokasi memori.

* int IdxHead

Indeks yang menandakan letak elemen pertama dalam queue.

* int IdxTail

Indeks yang menandakan letak elemen terakhir dalam queue.

Queue yang dipakai bersifat *circular buffer*, dimana IdxHead tidak harus berada di indeks ke-0, melainkan dapat berada di mana saja.

ADT ini memiliki beberapa primitif:

* void CreateStrQueue(QueueStr \*q)

Menciptakan sebuah queue kosong, dengan ketentuan IdxHead dan IdxTail bernilai IDX\_UNDEF (-1).

* boolean isStrEmpty(QueueStr q)

Mengirimkan nilai true apabila queue kosong; false jika sebaliknya.

* boolean isStrFull(QueueStr q)

Mengirimkan nilai true apabila queue penuh, yaitu apabila IdxTail berada di belakang IdxHead.

* int lengthStr(QueueStr q)

Mengembalikan panjang dari queue, yaitu banyaknya elemen yang berada di queue.

* void enqueueStr(QueueStr \*q, ElemType var)

Memasukkan elemen var ke bagian paling belakang queue, dimana queue pasti tidak penuh. Penambahan elemen ini mengikuti prinsip *First In First Out*.

* void dequeueStr(QueueStr \*q, ElemType \*var)

Mengambil elemen di bagian paling depan queue (IdxHead) sebagai var , lalu IdxHead dimundurkan sedemikian rupa sehingga mantan IdxHead tadi sudah terhapus. Penghapusan elemen ini mengikuti prinsip *First In First Out*.

* void DisplayStrQueue(QueueStr q)

Menampilkan queue ke layar dengan format setiap elemen yang dipisahkan oleh newline, mulai dari elemen pertama (IdxHead) sampai elemen terakhir (IdxTail).

Persoalan yang Diselesaikan

ADT Queue versi string ini erat kaitannya dengan user, dimana nantinya game yang ingin dimainkan oleh user ditampung melalui queue ini. Apabila user ingin melangkahi game melalui SKIP GAME sebanyak n kali, maka dequeue akan dilakukan sebanyak n kali juga. Apabila user ingin menambahkan game melalui QUEUE GAME, maka enqueue akan dilakukan.

Alasan Pemilihan

ADT Queue ini adalah representasi yang paling cocok terkait antrian game dari user, karena user pasti ingin memainkan game yang paling mereka inginkan, yang berada di bagian atas queue. Sedangkan game yang ingin mereka tambahkan, pasti berada di akhir rentetan keinginan mereka, dan dapat langsung dimainkan apabila user melangkahi game hingga game tersebut. Semua kebutuhan user ini akan sangat terbantu dengan penggunaan queue dibanding array dinamis.

# **Program Utama**

Program utama tugas besar ini terdapat pada file dengan nama "main.c". Program diawali dengan looping pertama yang menampilkan main menu. Looping ini akan selesai jika user memilih untuk keluar dari permainan. Apabila array yang berisi list game masih kosong maka user diminta input masukan menu dari user. Input dari user berupa "START" atau "LOAD".

Pembacaan input menggunakan ADT mesin kata. Jika user memilih "START" maka program akan mulai membaca file konfigurasi default yang berisikan list game dengan menggunakan ADT Mesin Kata. Setelah itu, string yang dihasilkan dimasukkan ke dalam array menggunakan ADT Array Dinamis. Jika user memilih "LOAD", user diminta memasukkan nama file. Berikutnya, file tersebut akan dibaca menggunakan ADT mesin kata.

Apabila, array game sudah diisi maka user dapat memasukkan command, di antara lain "CREATE GAME", "LIST GAME", "DELETE GAME", "QUEUE GAME", "PLAY GAME", "SKIP GAME", "HELP", "SAVE", atau "EXIT". Apabila user memasukan command selain yang disebutkan diatas maka command tersebut tidak valid dan user harus memasukan command baru. Apabila user memasukan "CREATE GAME", user dapat membuat dan menyimpan game(s) baru tersebut pada BNMO. Apabila user memasukan "LIST GAME", program akan menampilkan seluruh daftar game yang tersedia. Jika user memasukkan "DELETE GAME", program akan menghapus game sesuai input dari user. Jika user memasukkan "QUEUE GAME", program akan memasukkan game dalam antrian permainan sesuai dari input user. Jika user memasukan command "PLAY GAME", maka program akan menjalankan permainan sesuai dari antrian game. Apabila user memasukan command "SKIP GAME", maka antrian game akan dilewatkan sebanyak N kali, dimana N merupakan input dari user. Apabila pengguna memasukan input "SAVE", maka program akan menyimpan state dari permainan. Apabila pengguna memasukan command "HELP", program akan memunculkan penjelasan dari command -command yang lain. Apabila user memasukan command "QUIT", looping program akan berakhir dan user keluar dari program.

# **Algoritma-Algoritma Menarik**

## Algoritma Time Delay

Algoritma ini akan memberikan jeda waktu untuk mengeksekusi baris program berikutnya. Algoritma ini digunakan untuk menampilkan kata “loading <game> . . .” menjadi lebih menarik.

## Algoritma Pembacaan dan penulisan File

Algoritma pembacaan dan penulisan file digunakan membaca dan menuliskan file eksternal. Algoritma ini digunakan dalam fitur start , save dan load. Pada dasarnya algoritma ini menggunakan fitur-fitur dalam ADT mesin kata dan mesin karakter. Algoritma ini menarik karena melalui Algoritma ini , Program utama dapat mendapatkan informasi Game game yang tersedia serta menuliskan State List game terakhir ke suatu file txt baru ataupun jika sudah ada akan di overwrite.

## Algoritma Random Integer

Algoritma Random integer adalah Algoritma yang menghasilkan suatu nilai Random. Algoritma ini digunakan untuk permainan RNG , DINNER DASH , dan MOLE. Algoritma ini menarik karena Random Integer yang di *generate* dapat di kontrol dengan mengikutsertakan operasi modulo (%).

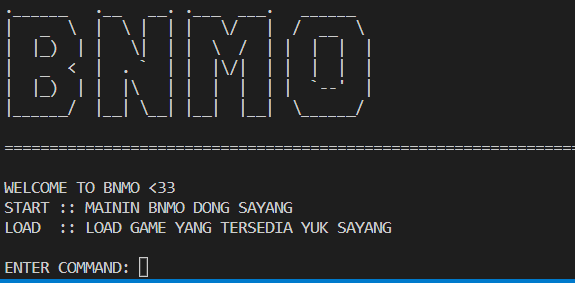
# **Data Test**

## Data Test 1 (kompilasi program)

Test ini dilakukan untuk memastikan bahwa program dapat dijalankan dengan menampilkan judul game dan main menu. Cara melakukan kompilasi program adalah dengan mengetik perintah:

“gcc gcc main.c console.c ../Game/dinerdash.c ../Game/RNG.c ADT/array/arraydinamikstr.c ADT/array/arraydinamik.c ADT/mesinkarakter/mesinkarakter.c ADT/mesinkata/mesinkata.c ADT/queue/queue.c ADT/queue/queuestr.c ../Game/mole.c -o main”

pada terminal, kemudian mengetik executable code program tersebut.

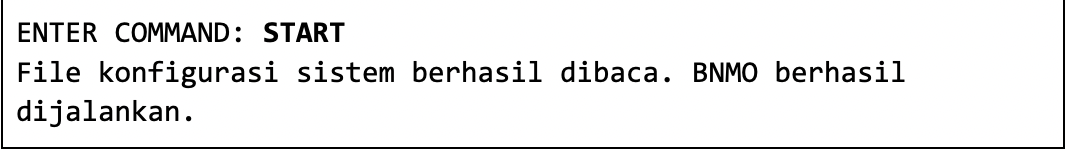


Gambar 6.1. Tampilan Judul dan Main Menu

## **Data Test 2 (S**TART)

Test ini dilakukan untuk menguji fitur START. START akan membaca file config default list permainan yang tersimpan pada file "config.txt".

* output yang diharapkan



Gambar 6.2. Fitur START sesuai yang diharapkan

* output yang dihasilkan

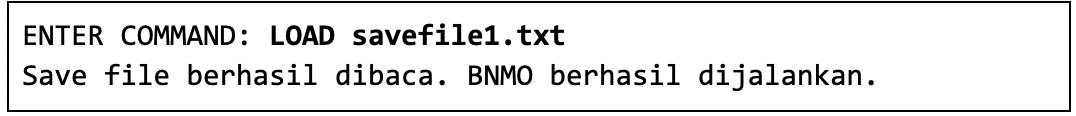


Gambar 6.3. START berhasil dilakukan

## Data Test 3 (LOAD)

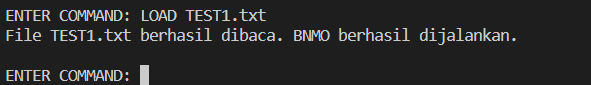
Test ini dilakukan untuk menguji fitur LOAD. LOAD akan membaca file yang pernah disimpan oleh pengguna. Pengguna juga diharuskan untuk menulis nama file yang pernah disimpan sebelumnya.

* Output yang diharapkan



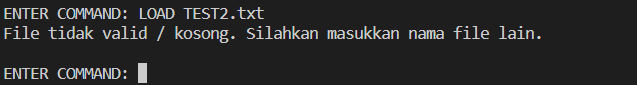
Gambar 6.4. Fitur Load yang diharapkan

* Output yang dihasilkan



Gambar 6.5. Load berhasil dilakukan

Apabila file yang ingin di-load tidak ditulis atau tidak ditemukan, maka program akan mengembalikan pesan error. Kemudian program akan meminta user untuk mengetik command lain.

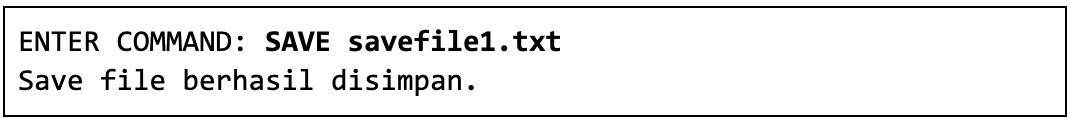


Gambar 6.6. Load gagal dilakukan karena masukan file tidak valid/kosong

## Data Test 4 (SAVE)

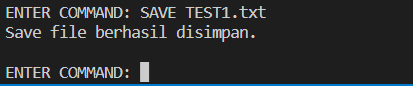
Test ini dilakukan untuk menguji fitur SAVE. SAVE akan menyimpan perubahan yang telah dilakukan oleh pengguna dengan menuliskan nama file yang diinginkan untuk disimpan. Apabila file tersebut sudah ada sebelumnya, maka file tersebut akan di-overwrite.

* Output yang diharapkan



Gambar 6.7. Fitur save yang diharapkan

* Output yang dihsilkan

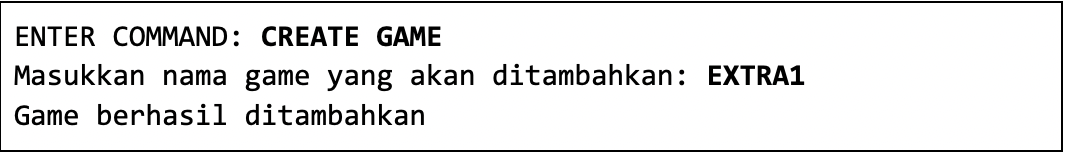


Gambar 6.8. SAVE barhasil dilakukan

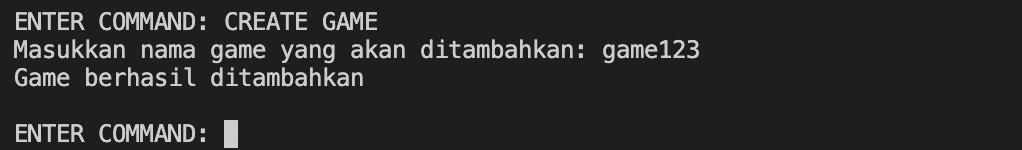
## Data Test 5 (CREATE GAME)

Test ini dilakukan untuk menguji fitur CREATE GAME. CREATE GAME akan menambahkan game pada list game yang tersedia.

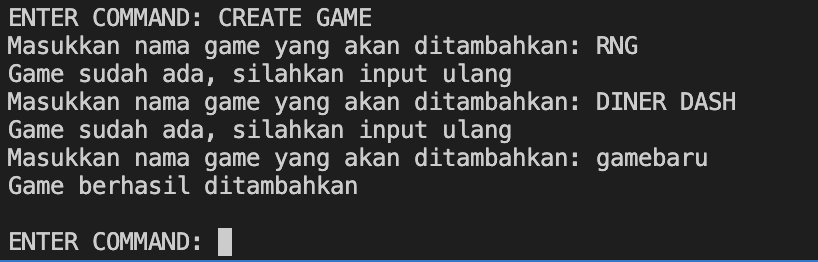
* Output yang diharapkan



Gambar 6.9. FItur create game yang diharapkan

* Output yang dihasilkan

Gambar 6.10. Create game berhasil dilakukan

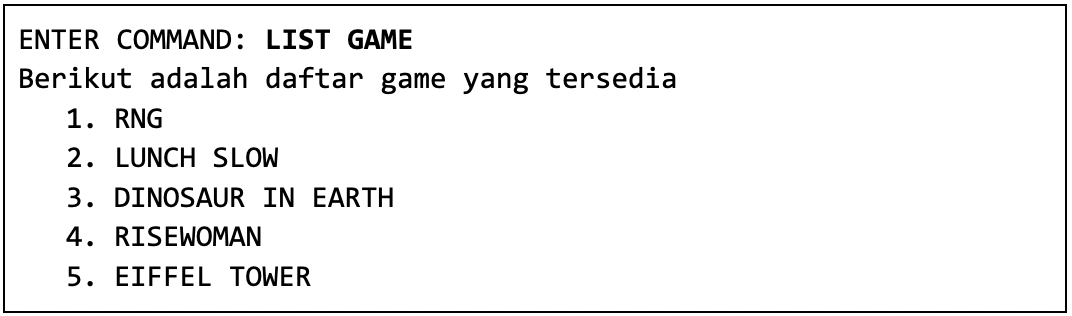
Apabila nama game yang sudah terdapat pada program, maka user diminta memasukan ulang nama game yang tidak ada pada program.

Gambar 6.11. Create game akan meminta input nama game berulang kali

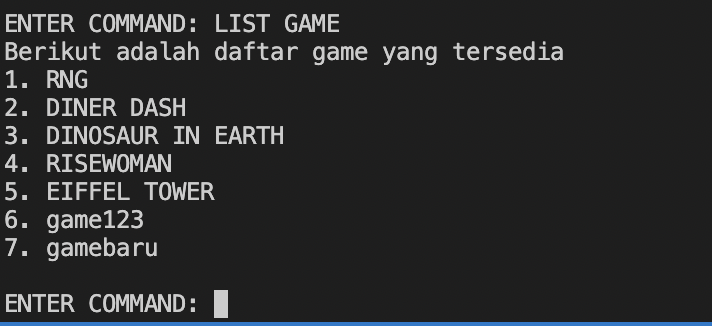
## Data Test 6 (LIST GAME)

Test ini dilakukan untuk menguji fitur LIST GAME. LIST GAME akan menampilkan seluruh game yang tersedia pada program.

* Output yang diharapkan



Gambar 6.12. Fitur List game yang diharpakan

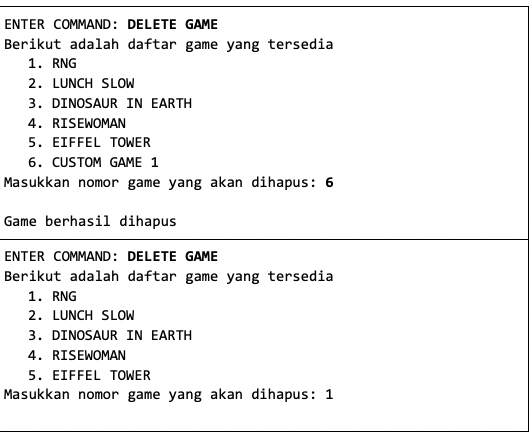
* Output yang dihasilkan

Gambar 6.13. List game menampilkan seluruh nama game yang terdapat pada program

## Data Test 7 (DELETE GAME)

Test ini dilakukan untuk menguji fitur DELETE GAME. DELETE GAME akan menghapus game seusai nomor game yang dimasukan oleh user.

* Output yang diharapkan



Gambar 6.14. Fitur delete game yang diharapkan

* Output yang dihasilkan

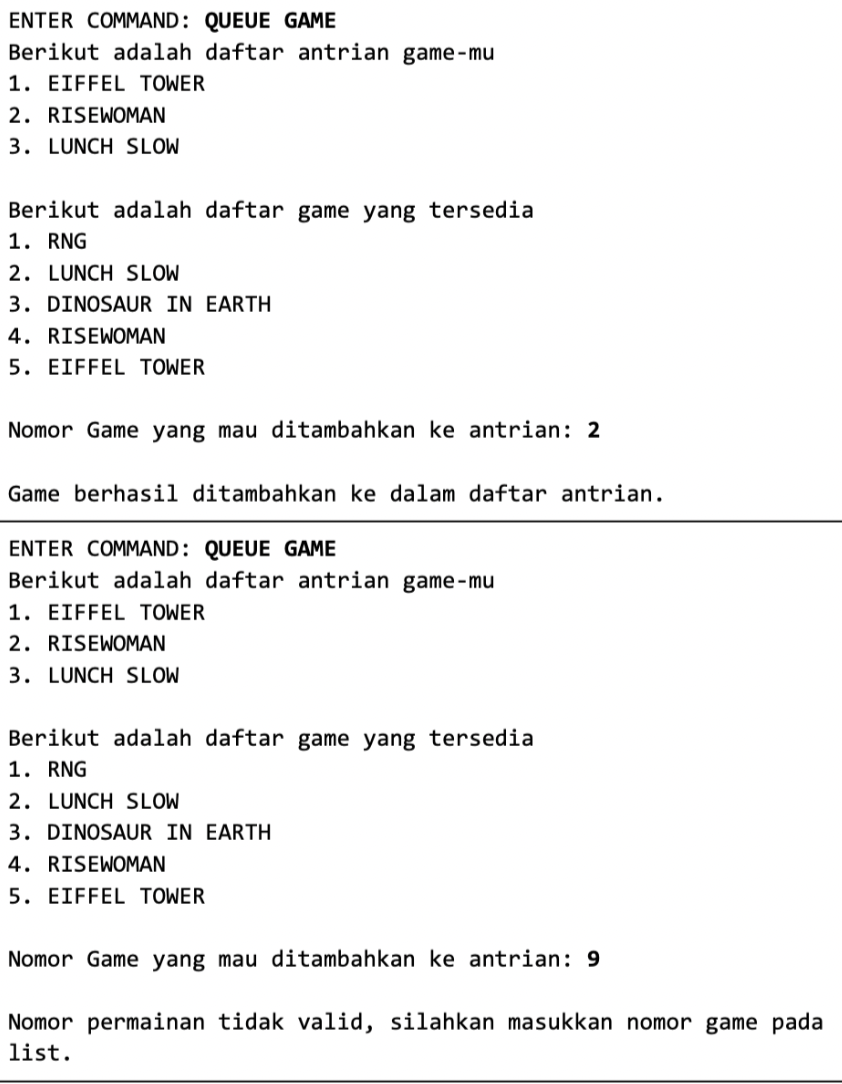


Gambar 6.15. list game menampilkan seluruh nama game yang ada

## Data Test 8 (QUEUE GAME)

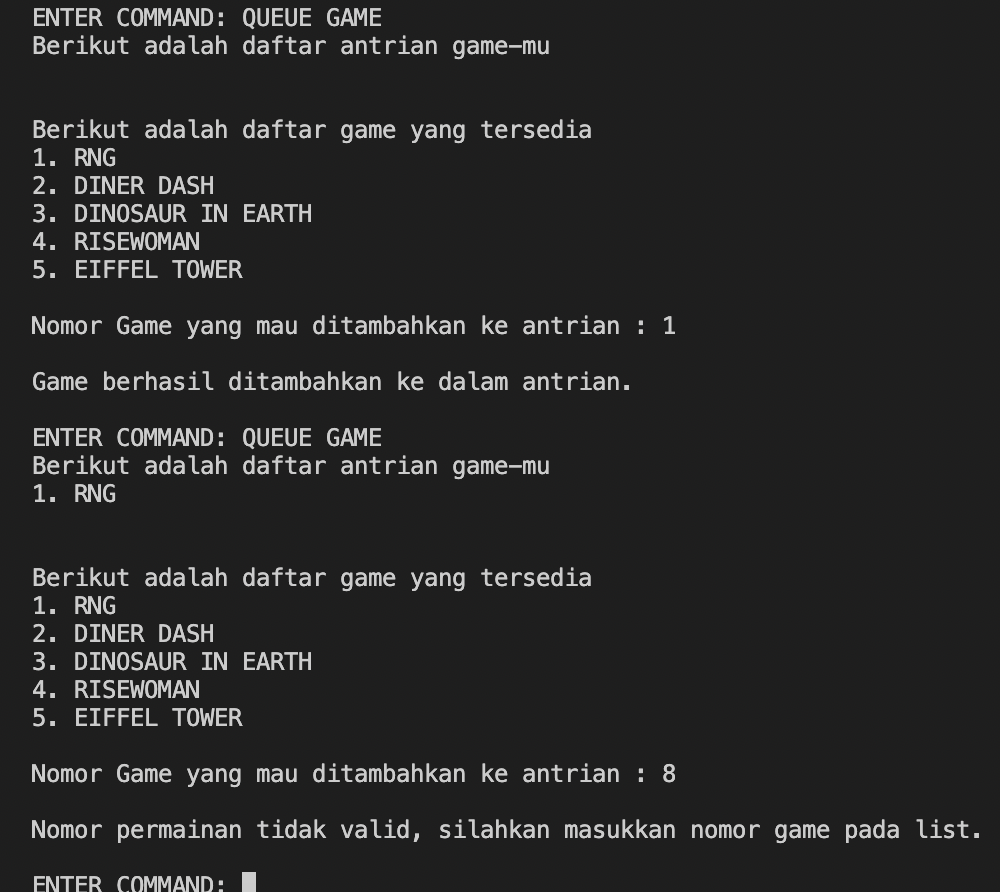
Test ini dilakukan untuk menguji fitur QUEUE GAME. QUEUE GAME akan menampilkan antrian game yang akan dimainkan oleh user dan user dapat memasukan game ke dalam antrian dengan memasukkan nomor game.

* Output yang diharapkan



Gambar 6.16. Fitur Queue game yag diharapkan

* Output yang dihasilkan

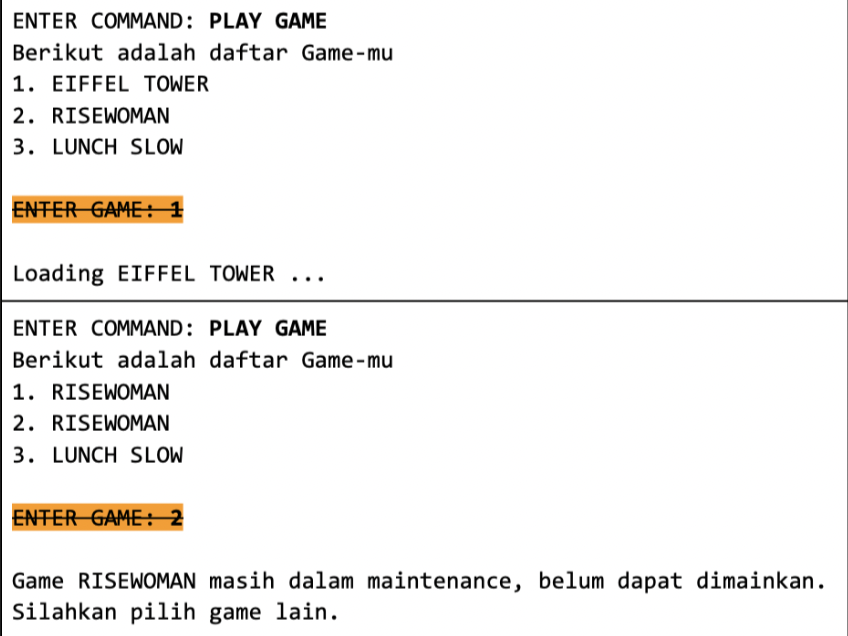


Gambar 6.17. Queue game yang akan dimainkan user dan nomor game yang dimasukan user

## Data Test 9 (PLAY GAME)

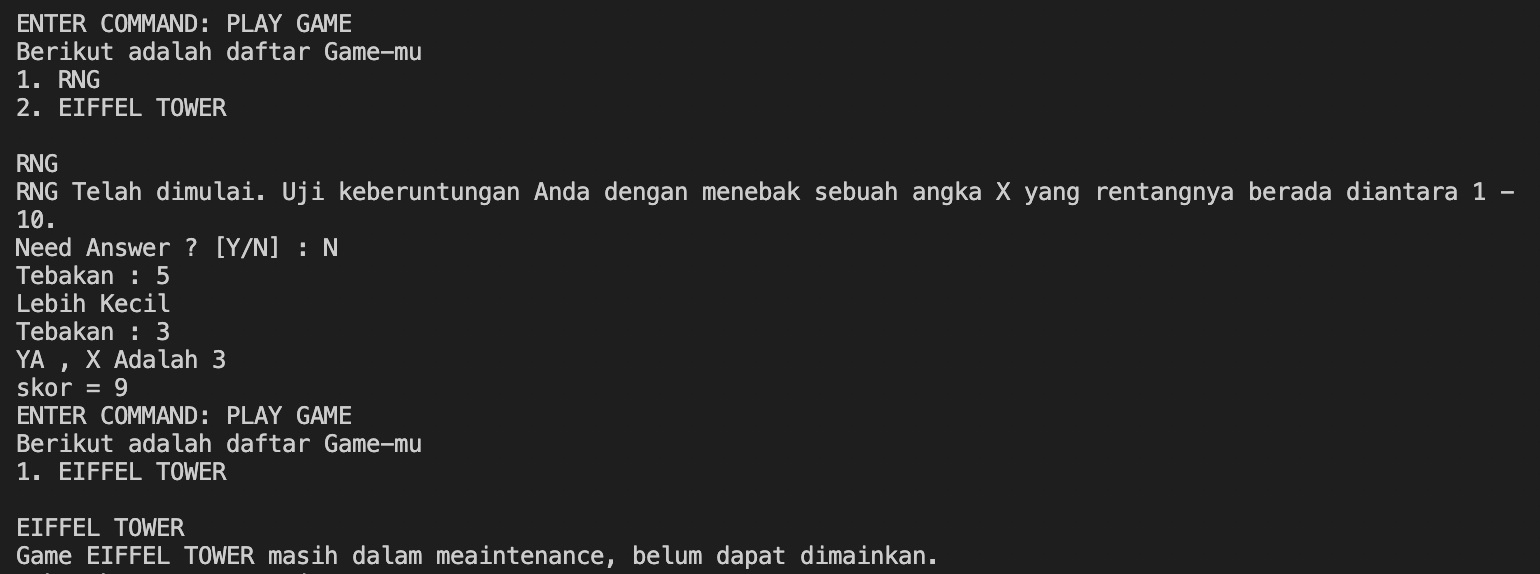
Test ini dilakukan untuk menguji fitur PLAY GAME. PLAY GAME akan memainkan game sesuai urutan antrian game yang sebelumnya telah ada.

* Output yang diharapkan



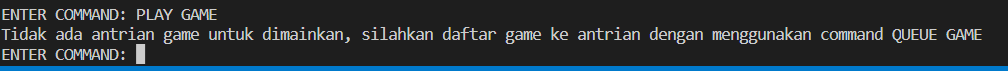
Gambar 6.18. Fitur Play game yang diharapkan

* Output yang dihasilkan



Gambar 6.19. play game memainkan game sesuai antrian game

Namun, apabila tidak terdapat game pada antrian, maka program akan mengirimkan pesan bahwa tidak ada game yang sedang mengantri.



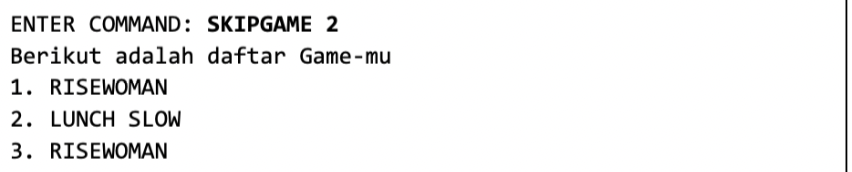
Gambar 6.20. play game ketika tidak ada game dalam antrian

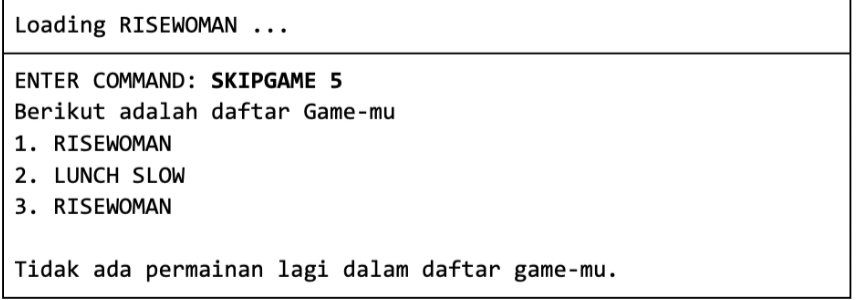
## Data Test 10 (SKIP GAME)

Test ini dilakukan untuk menguji fitur SKIP GAME. SKIP GAME memainkan game pada

antrian dengan melewatkan beberapa antrian terlebih dahulu.

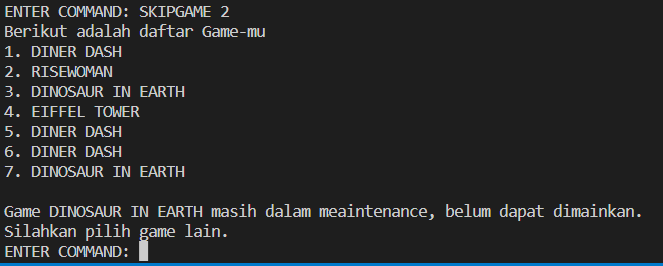
* Output yang diharapkan





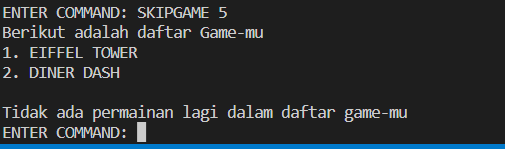
Gambar 6.21. Fitur skip game yang diharapkan

* Output yang dihasilkan



Gambar 6.22. skip game memainkan game dengan melewati beberapa game sesuai inputan

Apabila program melewati game sampai tidak ada lagi game yang tersedia di dalam antrian, maka program akan mengirimkan pesan bahwa tidak ada lagi game yang ada pada antrian.



Gambar 6.23. skip game dengan tidak ada game dalam antrian setelah diskip

Selain itu, apabila memang tidak terdapat game dalam antrian sejak awal, maka program akan mengirimkan pesan bahwa tidak ada game dalam antrian.

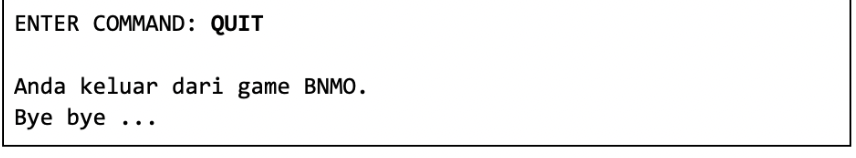


Gambar 6.24. skip game ketika tidak ada game dalam antrian

## Data Test 11 (QUIT)

Test ini dilakukan untuk menguji fitur QUIT. QUIT akan mengakhiri program mengirimkan pesan akhir ke pengguna

* Output yang diharapkan



Gambar 6.25. Fitur quit yang diharapkan

* Output yang dihasilkan



Gambar 6.26 Tampilan quit ketika dijalankan

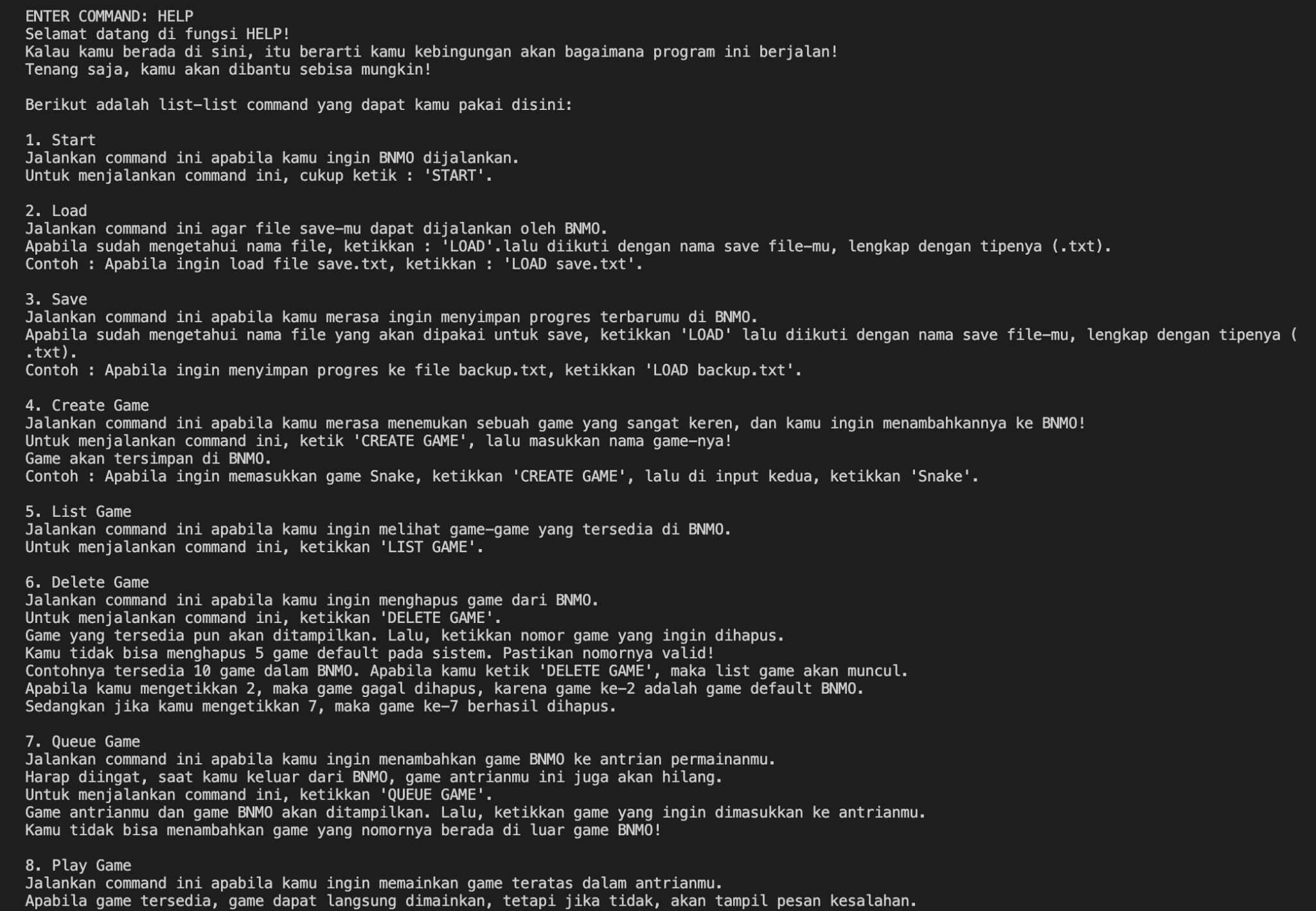
## Data Test 12 (HELP)

Test ini dilakukan untuk menguji fitur HELP. Fitur ini menampilkan penjelasan tentang tiap-tiap command.

* Output yang diharapkan

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

* Output yang dihasilkan



Gambar 6.27. Tampilan help ketika dijalankan

## 

# **Test Script**

| **No.** | **Fitur yang Dites** | **Tujuan Testing** | **Langkah-Langkah Testing** | **Input Data Test** | **Hasil yang Diharapkan** | **Hasil yang Keluar** |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | Fitur Main Menu | Memeriksa apakah permainan dapat dijalankan | Melakukan compile file "main.c" pada terminal | Data Test 1 | Permainan berhasil dan menampilkan main menu pada layar | Sesuai dengan yang diharapkan |
| 2 | Fitur Start | Memeriksa apakah permainan dapat membaca file konfigurasi default | Memasukkan input "START" saat permainan dapat dijalankan | Data Test 2 | Permainan berhasil membaca file dan menyimpannya dalam bentuk array | Sesuai dengan yang diharapkan |
| 3 | Fitur Load | Memeriksa apakah permainan dapat membaca file yang nama file-nya dimasukan user | Memasukkan input "LOAD" saat permainan dapat dijalankan. | Data Test 3 | Permainan berhasil membaca file dan menyimpannya dalam bentuk array | Sesuai dengan yang diharapkan |
| 4 | Fitur Save | Memeriksa apakah program dapat melakukan save permainan | Memasukkan command "SAVE" dan nama file sebagai tempat penyimpanan | Data Test 4 | File berhasil disimpan | Sesuai dengan yang diharapkan |
| 5 | Fitur Create Game | Memeriksa apakah program dapat menambah game baru pada daftar game | Memasukkan command "CREATE GAME" dan memasukan nama game yang ditambahkan | Data Test 5 | Pemain berhasil menambahkan game baru pada daftar game | Sesuai dengan yang diharapkan |
| 6 | Fitur List Game | Memeriksa apakah program menampilkan daftar game yang benar | Memasukkan command "LIST GAME" | Data Test 6 | Program berhasil menampilkan daftar game yang dapat dimainkan | Sesuai dengan yang diharapkan |
| 7 | Fitur Delete Game | Memeriksa apakah program dapat melakukan delete pada daftar game | Memasukkan command "DELETE GAME" dan memasukan nomor game yang dihapus | Data Test 7 | Pemain berhasil menghapus game dari daftar game | Sesuai dengan yang diharapkan |
| 8 | Fitur Queue Game | Memeriksa apakah program dapat membuat antrian game yang akan dimainkan | Memasukkan command "QUEUE GAME" dan dan memasukan nomor game yang ditambahkan pada antrian | Data Test 8 | Program berhasil menambahkan game pada antrian | Sesuai dengan yang diharapkan |
| 9 | Fitur Play Game | Memeriksa apakah program dapat menjalankan game sesuai antrian game | Memasukkan command "PLAY GAME" | Data Test 9 | Program berhasil menjalankan game sesuai urutan game. | Sesuai dengan yang diharapkan |
| 10 | Fitur Skip Game | Memeriksa apakah program dapat melewatkan antrian game | Memasukkan command "SKIP GAME" dan memasukkan jumlah game yang ingin dilewatkan | Data Test 10 | Program berhasil melewatkan urutan game sesuai jumlah game yang ingin dilewatkan | Sesuai dengan yang diharapkan |
| 11 | Fitur Quit | Memeriksa apakah program dapat menjalankan command QUIT | Memasukkan command "QUIT GAME" | Data Test 11 | Program selesai | Sesuai dengan yang diharapkan |
| 12 | Fitur Help | Memeriksa apakah program dapat menjalankan command HELP | Memasukkan command "HELP" | Data Test 12 | Program menampilkan penjelasan dari tiap - tiap command | Sesuai dengan yang diharapkan |

# **Pembagian Kerja dalam Kelompok**

| *Nama* | *NIM* | *Deskripsi Pekerjaan* |
| --- | --- | --- |
| Ivan Aldy Ganesen | 18221045 | * Mengerjakan ADT untuk Array Dinamis versi string, ADT Queue versi integer, dan ADT Queue versi string. * Mengerjakan HELP dan QUEUE GAME. * Mengisi laporan bagian Struktur Data (ADT). * Memeriksa kembali *spelling* dan informasi yang tersedia di laporan. * Menjadwalkan asistensi dengan asisten serta menjadi perantara kelompok dengan asisten. |
| Muhammad Shulhan | 18221051 | * Mengerjakan SAVE, CREATE GAME, PLAY GAME, dan SKIP GAME. * Mengisi laporan bagian Data Test, penjelasan tambahan spesifikasi tugas, dan melakukan notul ketika asistensi berlangsung. |
| Timothy Subekti | 18221063 | * Mengerjakan ADT untuk Mesin Karakter, ADT untuk Mesin Kata, dan RNG. * Mengerjakan LOAD dan LIST GAME. * Membuat driver untuk seluruh ADT. * Membuat function tambahan dan disimpan di dalam console. * Menggabungkan seluruh function menjadi main program. * Melakukan *debug* dan revisi dari seluruh function yang digabungkan. * Melakukan *testing* program. * Membuat Readme bagian kompilasi. * Membuat Bash Shell Script File. * Mengisi laporan bagian Algoritma Menarik. |
| Ken Azizan | 18221107 | * Mengerjakan menu utama, START, COMMAND LAIN, dan QUIT. * Mengisi laporan bagian Ringkasan, Program utama, Data Test, Test Script, dan Notulen. |
| Nicholas | 18221165 | * Mengerjakan ADT Array Statis dengan ElType integer. * Mengerjakan DELETE GAME, Diner Dash, serta game bonus (Mole). * Mengisi laporan bagian Penjelasan Tambahan, Spesifikasi Tugas dan Struktur Data.. * Melakukan testing, fix dan spellcheck pada beberapa fungsi dan aspek lain dalam program. |

# **Lampiran**

## ***Deskripsi Tugas Besar*** 1

**Game Mechanics**

1. **About the Game**

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

1. Memainkan game
2. Menambahkan game
3. Menghapus game
4. Mengurutkan game yang akan dimainkan
5. **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 menerima input commands yang akan dijelaskan pada bagian berikutnya.

1. **Command**

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

1. **START**

Command ini berfungsi untuk membaca file konfigurasi default yang berisi game yang dapat dimainkan.

1. **LOAD**

Command ini berfungsi untuk membaca save file yang berisi list game yang dapat dimainkan, histori, dan scoreboard game.

1. **SAVE**

Command ini berfungsi untuk menyimpan state game permainan ke dalam suatu file.

1. **CREATE GAME**

Command ini berfungsi untuk menambahkan game baru pada daftar game.

1. **LIST GAME**

Command ini berfungsi untuk menampilkan daftar game yang disediakan sistem.

1. **DELETE GAME**

Command ini berfungsi untuk menghapus sebuah game dari daftar game.

1. **QUEUE GAME**

Command ini berfungsi untuk mendaftarkan permainan ke dalam list.

1. **PLAY GAME**

Command ini berfungsi untuk memainkan sebuah permainan sesuai dengan urutannya.

1. **SKIP GAME**

Command ini berfungsi untuk melewatkan permainan sebanyak n kali.

1. **HELP**

Command ini berfungsi untuk menampilkan informasi terkait command - command lainnya.

1. **QUIT**

Command ini berfungsi untuk keluar dari program.

## ***Notulen Rapat***

Sabtu, 29 Oktober 2022

Memahami spesifikasi dengan tugas dan membagi job description untuk masing-masing anggota kelompok.

Kamis, 3 November 2022

Asistensi pertama untuk menanyakan lebih lanjut perihal spesifikasi pada tugas besar.

Jumat, 4 November 2022

Melakukan breakdown ADT - ADT yang digunakan dan saling memberitahu progress yang telah dilakukan tiap orang.

Selasa, 8 November 2022

Menyatukan program pada file main.c dan melakukan debugging pada program.

Jumat, 11 November 200

Asistensi kedua untuk mengkonfirmasi beberapa hal mengenai program dan laporan kepada asisten.

## ***Log Activity Anggota Kelompok***

| Hari / Tanggal | Nama | Kegiatan |
| --- | --- | --- |
| Minggu / 30 Oktober 2022 | Timothy Subekti | Membuat repository Github dan menginvite Team |
| Kamis / 3 November 2022 | Ivan Aldy Ganesen | Menyelesaikan ADT Queue versi integer. |
| Sabtu / 5 November 2022 | Ivan Aldy Ganesen | Menyelesaikan ADT Array Dinamik versi string. |
| Senin / 7 November 2022 | Ivan Aldy Ganesen | Menyelesaikan HELP. |
| Senin / 7 November 2022 | Muhammad Shulhan | Menyelesaikan SAVE dan CREATE GAME. |
| Senin / 7 November 2022 | Timothy Subekti | Membuat ADT mesin karakter dan kata serta additional function untuk ditaruh di console |
| Senin / 7 November 2022 | Timothy Subekti | Menyelesaikan fungsi Load dan ADT mesin karakter dan Kata |
| Selasa / 8 November 2022 | Nicholas | Menyelesaikan dinerdash.c. |
| Selasa / 8 November 2022 | Ivan Aldy Ganesen | Menyelesaikan QUEUE GAME dan ADT Queue versi string; memperbaiki tipe typedef untuk ADT Array Dinamis dan Queue. |
| Selasa / 8 November 2022 | Muhammad Shulhan | Menyelesaikan PLAY GAME dan SKIP GAME. |
| Selasa / 8 November 2022 | Timothy Subekti | Membuat Struktur Main Program |
| Selasa / 8 November 2022 | Timothy Subekti | Menyelesaikan bug dari Load File. |
| Selasa / 8 November 2022 | Ken Azizan | Menyelesaikan START, QUIT, dan command lain. |
| Rabu / 9 November 2022 | Nicholas | Menyelesaikan dinerdash.h dan DELETEGAME. |
| Rabu / 9 November 2022 | Timothy Subekti | Menyelesaikan RNG dan fungsi LIST GAME |
| Rabu / 9 November 2022 | Timothy Subekti | Menggabungkan seluruh Fungsi , melakukan Debug , serta merevisi seluruh FUngsi |
| Rabu / 9 November 2022 | Nicholas | Menyelesaikan mole.c |
| Kamis / 10 November 2022 | Nicholas | Menyelesaikan mole.h |
| Kamis / 10 November 2022 | Timothy Subekti | Melakukan revisi main program dan membuat semua driver ADT |
| Kamis / 10 November 2022 | Timothy Subekti | Membuat .sh file |
| Kamis / 10 November 2022 | Muhammad Shulhan | Menyelesaikan laporan bagian data test dan tambahan spesifikasi. |
| Jumat / 11 November 2022 | Ivan Aldy Ganesen & Nicholas | Menyelesaikan laporan bagian Struktur Data. |
| Jumat / 11 November 2022 | Ivan Aldy Ganesen | Menyelesaikan laporan bagian Pembagian Kerja. |
| Jumat / 11 November 2022 | Ivan Aldy Ganesen | Memperbaiki beberapa frasa dan letak kalimat untuk keseluruhan laporan. |
| Jumat / 11 November 2022 | Timothy Subekti | Melengkapi Readme dan membantu membuat laporan |
| Jumat / 11 November 2022 | Muhammad Shulhan | Melengkapkan laporan di bagian yang belum lengkap, formatting, dll. |
| Jumat / 11 November 2022 | Nicholas | Menyelesaikan Penjelasan Tambahan pada program. |
| Jumat / 11 November 2022 | Ken Azizan | Menyelesaikan Ringkasan, Penjelasan Program Utama, Data Test, dan Test Script pada laporan |

## Lain - Lain

