

LAPORAN TUGAS BESAR

IF2111 Algoritma dan Struktur Data


BNMO

Dipersiapkan oleh:

Kelompok 05

Rahmah Putri Azzahra	18219052
Vincent Franstyo	18221100
Karina Rahadiani	18221104
Christina Wijaya	18221106
Sultan Alta Alvaro Valencia	18221110

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

	Sekolah Teknik Elektro dan Informatika ITB	Nomor Dokumen		Halaman
		<i>IF2111-TB1-05</i>		<i>40</i>
		<i>Revisi</i>	<i>0</i>	<i>28/10/2022</i>

Daftar Isi

1 Ringkasan	3
2 Penjelasan Tambahan Spesifikasi Tugas	4
2.1 Spesifikasi Fitur Header BNMO	4
2.2 Spesifikasi Fitur displayQueueGame	4
2.3 Spesifikasi Fitur Game Hangman (Bonus)	4
2.4 Spesifikasi Fitur Game Tic Tac Toe (Bonus)	5
3 Struktur Data (ADT)	5
3.1 ADT Array	5
3.2 ADT ArrayOfGame	6
3.3 ADT List	7
3.4 ADT Mesin Karakter	9
3.5 ADT Mesin Kata	9
3.6 ADT Priority Queue	10
3.7 ADT Queue	11
4 Program Utama	12
5 Algoritma menarik	15
6 Data Test	15
6.1 START	15
6.2 LOAD	16
6.3 SAVE	16
6.4 CREATE GAME	17
6.5 LIST GAME	17
6.6 DELETE GAME	18
6.7 QUEUE GAME	18
6.8 PLAY GAME	19
6.9 SKIP GAME	19
6.10 QUIT	20
6.11 HELP	20
6.12 COMMAND LAIN	20
6.13 DINER DASH	21
6.14 RNG	23
6.15 TICTACTOE	23
7 Test Script	25

8	Pembagian Kerja dalam Kelompok	29
9	Lampiran	29
9.1	Deskripsi Tugas Besar 1	37
9.2	Notulen Rapat	37
9.3	Log Activity Anggota Kelompok	39

1 Ringkasan

BNMO adalah sebuah sistem game yang diakses melalui *command-line interface*. Program ini memiliki empat fungsi utama yaitu memainkan game, menambahkan game, menghapus game, dan mengurutkan game yang akan dimainkan sesuai antrian yang dimasukan. Jadi user bisa menyunting game mereka juga memainkannya dalam program BNMO ini.

Pada pemenuhan tugas besar kali ini, pembuatan program BNMO juga diikuti oleh pemenuhan pembuatan dua game utama yaitu Diner Dash dan Random Number Generator. Diner Dash adalah game yang diadaptasi dari sistem pelayanan pada restoran, dimana *user* yang memainkan game ini harus bisa mengatur sistem antrian pelayanan kepada pelanggan restorannya. Sedangkan dalam game Random Number Generator *user* diminta menebak sebuah angka acak yang dibuat oleh mesin dengan *clue* tebakan apakah lebih besar atau lebih kecil. Selain game-game di atas, ada dua game bonus yang berhasil ditambahkan yaitu Tic Tac Toe yang diadaptasi dari permainan Tic Tac Toe pada umumnya dan Hangman yang di dalamnya terdapat kata-kata yang harus ditebak dan masih bernuansa topik sehari-hari dari mahasiswa STI.

Laporan ini akan memuat semua proses pembuatan sistem BNMO dimulai dari penjelasan spesifikasi, struktur data yang digunakan, program utama, penjelasan algoritma-algoritma yang ditemui menarik, data test, test script, pembagian tugas antar anggota, dan lampiran.

2 Penjelasan Tambahan Spesifikasi Tugas

2.1 Spesifikasi Fitur Header BNMO

Fitur Header digunakan sebagai penanda program berhasil di compile dan siap dijalankan saat dimasukkan command LOAD atau START. Fitur ini hanya berisi tampilan ke layar menggunakan printf yang direpresentasikan oleh karakter karakter unik.

2.2 Spesifikasi Fitur displayQueueGame

Fitur displayQueueGame ditambahkan untuk mempermudah pengerjaan program BNMO. Fitur ini akan menampilkan antrian game yang sudah user masukan sendiri lewat fitur QUEUE GAME. displayQueueGame dibuat sebagai upaya untuk penyederhanaan dan pengorganisasian program utama.

2.3 Spesifikasi Fitur Game Hangman (Bonus)

Hangman merupakan permainan menebak kata dengan cara memasukkan tebakan berupa huruf-huruf tertentu. Berikut adalah spesifikasi game ini:

- Pada awal permainan, program merandom kata yang akan ditebak pemain, lalu menampilkan simbol *underscore* (`_`) sebanyak jumlah huruf kata tebakan. Program juga akan menampilkan gambar tiang gantungan yang masih kosong.
- Pemain diminta untuk memasukan satu huruf sebagai tebakan. Jika huruf yang ditebak ada pada kata tebakan, maka huruf yang benar tersebut ditampilkan pada layar dengan mengganti simbol *underscore* sesuai urutan di mana huruf tersebut berada. Jika huruf yang ditebak tidak ada pada kata tebakan, maka gambar tiang gantungan akan bertambah sebanyak 1 kali dengan urutan (tali, kepala, badan, tangan kanan, tangan kiri, kaki kanan, dan kaki kiri).

- Permainan selesai apabila pemain berhasil menebak kata sebelum gambar tiang gantungan mencapai gambar yang lengkap (hingga kaki kiri tergambar) atau jika pemain tidak berhasil menebak kata (gambar tiang gantungan sudah mencapai gambar yang lengkap).
- Skor akhir game ini diperoleh dengan rumus $100 - (\text{jumlah salah}^2)$. Namun, jika pemain tidak berhasil menebak kata, maka skor pemain 0.

2.4 Spesifikasi Fitur Game Tic Tac Toe (Bonus)

Game Tic Tac Toe merupakan game strategi sederhana yang dimainkan oleh dua user. Disini user akan melawan mesin. Game ini akan menjalankan :

- Game akan menampilkan tabel kosong berukuran 3x3
- User akan memasukan karakter O pada salah satu tabel yang tersedia
- Program akan mengeluarkan gerakannya dengan mengisi karakter X di sisa tabel yang tersedia
- User harus membentuk membuat karakter O berderet tiga dengan bentuk vertikal, horizontal, maupun diagonal agar bisa menang dalam permainan.
- Skor akhir user akan berupa 20 - banyak langkah dari user.

3 Struktur Data (ADT)

3.1 ADT Array

3.1.1. Sketsa Struktur Data

```
#define InitialSize 10

typedef int IdxTypeArrayDin;
typedef char* ElTypeArrayDin;
typedef struct {
    ElTypeArrayDin *A;
    int Capacity;
    int Neff;
} ArrayDin;
```

Berikut primitif yang digunakan dalam ADT ini:

1. Konstruktor:
 - a. ArrayDin MakeArrayDin()
2. Destruktor:
 - a. void DeallocateArrayDin(ArrayDin *array)
3. Prototype:
 - a. boolean IsEmpty(ArrayDin array)
 - b. int Length(ArrayDin array)
4. Selektor:
 - a. ElTypeArrayDin Get(ArrayDin array, IdxTypeArrayDin i)
 - b. void Set(ArrayDin* array, IdxTypeArrayDin i, ElTypeArrayDin x)
 - c. int GetCapacity(ArrayDin array)
5. Operasi-Operasi:
 - a. void InsertAt(ArrayDin *array, ElTypeArrayDin el, IdxTypeArrayDin i)

- b. void InsertLast(ArrayDin *array, ElTypeArrayDin el)
- c. void InsertFirst(ArrayDin *array, ElTypeArrayDin el)
- d. void DeleteAt(ArrayDin *array, IdxTypeArrayDin i)
- e. void DeleteLast(ArrayDin *array)
- f. void DeleteFirst(ArrayDin *array)
- g. void PrintArrayDin(ArrayDin array)
- h. void ReverseArrayDin(ArrayDin *array)
- i. ArrayDin CopyArrayDin(ArrayDin array)
- j. IdxTypeArrayDin SearchArrayDin(ArrayDin array, ElTypeArrayDin el)

3.1.2. Persoalan yang dapat diselesaikan

ADT ini digunakan untuk menyimpan daftar kata tebakan, daftar gambar hangman, serta daftar blank (urutan simbol *underscore* (`_`) sesuai jumlah huruf kata tebakan yang ditampilkan ke pengguna) pada game Hangman (Bonus).

3.1.3. Alasan Pemilihan ADT

ADT ini digunakan karena diperlukannya penyimpanan untuk daftar kata tebakan, daftar gambar hangman, dan daftar blank yang akan digunakan di sepanjang permainan. Kata tebakan yang muncul saat permainan akan diambil secara random dari array kata tebakan, daftar gambar hangman akan ditampilkan satu per satu sesuai jumlah kesalahan pemain, serta daftar blank akan selalu ditampilkan menyesuaikan dengan jumlah huruf yang berhasil ditebak pemain.

3.1.4. Implementasi sebagai ADT

arraydin.c, arraydin.h, driver_arraydin.c

3.2 ADT ArrayOfGame

3.2.1. Sketsa Struktur Data

```
#define InitialSize 10

typedef int IdxTypeArrayOfGame;
typedef Word ElTypeArrayOfGame;
typedef struct
{
    ElTypeArrayOfGame *A;
    int Capacity;
    int Neff;
} ArrayOfGame;
```

Berikut primitif yang digunakan dalam ADT ini:

1. Konstruktor:
 - a. ArrayOfGame MakeArrayOfGame()
2. Destruktor
 - a. void DeallocateArrayOfGame(ArrayOfGame *array)
3. Prototype

- a. `IdxTypeArrayOfGame SearchArrayOfGame(ArrayOfGame array, ElTypeArrayOfGame el)`
- 4. Selektor
 - a. `ElTypeArrayOfGame GetGame(ArrayOfGame array, IdxTypeArrayOfGame i)`
 - b. `int GetCapacityArrayOfGame(ArrayOfGame array)`
- 5. Tes Empty/Full
 - a. `boolean IsEmptyArrayOfGame(ArrayOfGame array)`
 - b. `boolean IsFullArrayOfGame(ArrayOfGame array)`
- 6. Insert/Delete
 - a. `void InsertGameAt(ArrayOfGame *array, ElTypeArrayOfGame el, IdxTypeArrayOfGame i)`
 - b. `void InsertGameLast(ArrayOfGame *array, ElTypeArrayOfGame el)`
 - c. `void InsertGameFirst(ArrayOfGame *array, ElTypeArrayOfGame el)`
 - d. `void DeleteGameAt(ArrayOfGame *array, IdxTypeArrayOfGame i)`
 - e. `void DeleteGameLast(ArrayOfGame *array)`
 - f. `void DeleteGameFirst(ArrayOfGame *array)`
- 7. Lain Lain
 - a. `void ReverseArrayOfGame(ArrayOfGame *array)`
 - b. `ArrayOfGame CopyArrayOfGame(ArrayOfGame array)`
 - c. `void PrintArrayOfGame(ArrayOfGame array)`

3.2.2. Persoalan yang dapat diselesaikan

ADT ini digunakan pada banyak prosedur utama, yaitu LIST GAME, QUEUE GAME, dan turunan dari prosedur-prosedur tersebut. `ArrayOfGame` menyimpan list game yang tersedia.

3.2.3. Alasan pemilihan penggunaan ADT

Dibutuhkannya array yang menampung daftar game yang tersedia dan berbentuk game.

3.2.4. Implementasi sebagai ADT

`arrayOfGame.c`, `arrayOfGame.h`, `driver_arrayOfGame.c`

3.3 ADT List

3.3.1. Sketsa Struktur Data

```
#define IdxType int

typedef struct
{
    /* data */
    int foodID;
```

```

        int cookDuration;
        int stayDuration;
        int price;
    } PQEType;

typedef struct
{
    PQEType food[MaxEl]; /* Memori tempat penyimpanan elemen
(container) */
} List;

```

Berikut primitif dalam ADT List:

1. Selektor
 - a. PQEType GetListChar(List L, IdxType i)
 - b. void SetListChar(List *L, IdxType i, PQEType v)
 - c. int LengthList(List L)
 - d. IdxType FirstIdxList (List L)
 - e. IdxType LastIdxList (List L)
2. Konstruktor
 - a. List MakeList()
3. Test Kosong/Penuh
 - a. boolean IsEmptyList(List L)
4. Test Index
 - a. boolean IsIdxValidList (List L, IdxType i)
 - b. boolean IdIdxEffList(List L, IdxType i)
5. Prototype
 - a. boolean SearchList (List L, PQEType X)
6. Insert/Delete
 - a. InsertFirstList (List *L, PQEType X)
 - b. InsertAtList (List *L, PQEType X, IdxType i)
 - c. InsertLastList (List *L, PQEType X)
 - d. DeleteFirstList (List *L)
 - e. DeleteAtList (List *L, IdxType i)
 - f. DeleteLastList (List *L)
 - g. ConcatList(List L1, List L2)
7. Operasi Lain
 - a. void displayTime (List L)
 - b. void displayStay (List L)
 - c. IdxType minList (List L)

3.3.2. Persoalan yang dapat diselesaikan:

ADT ini digunakan untuk menyimpan antrian makanan yang sedang dimasak antrian makanan yang dapat disajikan pada game Diner Dash.

3.3.3. Alasan pemilihan penggunaan ADT:

ADT ini digunakan karena diperlukannya penghapusan dan penambahan elemen pada indeks tertentu.

3.3.4. Implementasi sebagai ADT:

list.c, list.h, list_driver.c

3.4 ADT Mesin Karakter

3.4.1 Sketsa Struktur Data

```
#define MARK '\\0'
/* State Mesin */
extern char currentChar;
extern boolean EOP;
```

Primitif yang digunakan dalam ADT ini:

1. void START(char *FILE)
2. void STARTINPUT()
3. void ADV()
4. char GetCC()
5. boolean IsEOP()

3.4.2 Persoalan yang dapat diselesaikan

ADT ini digunakan untuk mengakses char dari sebuah command yang berasal dari user ataupun dari file yang nantinya akan diakses di ADT mesin kata.

3.4.3 Alasan Pemilihan ADT

ADT ini diperlukan karena kita membutuhkan START dan ADV untuk mengakses char dari sebuah input dari user ataupun dari file.

3.4.4 Implementasi sebagai ADT

mesinkarakter.h, mesinkarakter.c, driver_mesinkata_mesinkarakter.c

3.5 ADT Mesin Kata

3.5.1. Sketsa Struktur Data

```
#define NMax 50
#define BLANK '\\n'

typedef struct
{
    char TabWord[NMax];
    int Length;
} Word;

/* State Mesin Kata */
extern boolean EndWord;
extern Word currentWord;
```

Berikut primitif yang digunakan dalam ADT ini:

1. Operasi-Operasi:
 - a. void IgnoreBlanks()

- b. void STARTWORD(char * FILE)
 - c. void STARTINPUTKATA()
 - d. void ADVWORD()
 - e. void CopyWord()
 - f. char * ConcatChar(char * path, char * filename)
2. Konverter:
- a. int WordToInt(Word word)
 - b. char * WordToString (Word word)
 - c. Word StringtoWord (char*string)
3. Selektor:
- a. Word takeword(Word command, int ke)
4. Tes Kebenaran:
- a. boolean IsEqual(Word w, char *c)
 - b. boolean IsInWord(char* dicari, Word sumber)

3.1.2. Persoalan yang dapat diselesaikan

ADT ini digunakan untuk melakukan parsing command program dan pembacaan file konfigurasi ke dalam aplikasi. Selain itu, ADT ini juga dipakai dalam beberapa fungsi dan game, khususnya untuk mengkonversi tipe data tertentu.

3.1.3. Alasan Pemilihan ADT

ADT ini digunakan karena diperlukannya suatu fungsi untuk membaca file konfigurasi dan file penyimpanan, serta membaca masukan dari pengguna tanpa menggunakan fungsi scanf bawaan C yang tidak diperbolehkan digunakan dalam tugas besar ini. Karena hasil pembacaan yang didapat dari fungsi ini berbentuk tipe data Word, maka ADT ini juga diperlukan untuk mengkonversi dan melakukan operasi-operasi tertentu terhadap hasil pembacaan tersebut.

3.1.4. Implementasi sebagai ADT

mesinkata.c, mesinkata.h, driver_mesinkata_mesinkarakter.c

3.6 ADT Priority Queue

3.6.1 Skema Struktur Data

```
typedef struct
{
    /* data */
    int foodID;
    int cookDuration;
    int stayDuration;
    int price;
} PQElementType;
typedef struct
{
    PQElementType buffer[PQCAPACITY];
    int idxHead;
    int idxTail;
} PrioQueue;
```

Berikut primitif dalam ADT Prioqueue:

1. Konstruktor
 - a. void CreateQueuePQ (PrioQueue *q)
2. Test Full/ Kosong
 - a. boolean isEmptyPQ(PrioQueue q)
 - b. boolean isFullPQ(PrioQueue q)
3. Selektor
 - a. int lengthPQ(PrioQueue q)
4. Enqueue/Dequeue
 - a. void enqueuePQ(PrioQueue *q, PQElType val)
 - b. void enqueueCSQ(PrioQueue *q, PQElType val)
 - c. void dequeuePQ(PrioQueue *q, PQElType *val)
5. Display
 - a. void displayQueuePQ(PrioQueue q)
 - b. void displayTimePQ(PrioQueue q)
 - c. void displayStayPQ(PrioQueue q)
6. Tes Anggota
 - a. boolean isMemberPQ(PrioQueue q, int id)

3.6.2 Persoalan yang dapat diselesaikan

ADT ini digunakan dalam membuat queue di game diner dash .

3.6.3 Alasan pemilihan ADT

Karena di game diner dash setiap elemen membutuhkan 4 variable integer yaitu ID, durasi memasak, ketahanan, dan harga.

3.6.4 Implementasi sebagai ADT

prioqueue.c, prioqueue.h, prioqueue_driver.h

3.7 ADT Queue

3.7.1 Skema Struktur Data

```
typedef Word ElTypeQueue;
typedef struct
{
    ElTypeQueue buffer[CAPACITY];
    int idxHead;
    int idxTail;
} Queue;
```

Berikut primitif yang digunakan dalam ADT ini:

1. Kreator:
 - a. void CreateQueue (Queue *q)
2. Prototype
 - a. boolean isEmpty(Queue q)
 - b. boolean isFull (Queue q)\
 - c. length (Queue q)
3. Enqueue/Dequeue:

- a. enqueue(Queue *q, ElTypeQueue val)
- b. dequeue(Queue *q, ElTypeQueue *val)
4. Operasi lain :
 - a. void displayQueue (Queue q)
 - b. void copyQueue (Queue *queueInput, Queue *queueOutput)
 - c. boolean isInQueue (Queue q, ElTypeQueue x)

3.7.2 Persoalan yang dapat diselesaikan

ADT ini digunakan untuk memasukkan game yang akan dimainkan ke dalam sebuah Game Queue agar dapat dimainkan sesuai dengan urutannya.

3.7.3 Alasan Pemilihan Penggunaan ADT

ADT ini digunakan karena diperlukan keterurutan dalam pemanggilan sebuah game yang akan dimainkan. Pemanggilan tersebut dimulai dari game terawal yang telah dimasukkan ke dalam ADT Queue. Queue memiliki sifat First In First Out (FIFO). Maka dari itu, Queue adalah ADT yang cocok untuk implementasi fungsi ini.

3.7.4 Implementasi sebagai ADT

queue.c, queue.h, main.C

4 Program Utama

Berikut adalah algoritma utama yaitu main.c

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

#include "../src/game_util/game_util.h"
#include "../src/games/diner_dash/diner_dash.h"
#include "../src/games/random_number_generator/random_number_generator.h"
#include "../src/games/game_tambahan/game_tambahan.h"
#include "../src/games/hangman/hangman.h"
#include "../src/main_util/main_util.h"

void displaybinomo()
{
    printf("
    _____\\ /_____ | / \\ \\ / | / _____ \\ \\ / \\ \\ / | / _____ \\ \\ \\n");
    printf("$$$$$$$$$ |$$$$$$$/ $$ \\ \\ $$ |/$$$$$$ |$$ \\ \\ /$$ |/$$$$$$ |\\n");
    printf("$$_$ |__$$_$ | $$ $ | $$$ \\ \\ $$ |$$ | $$ |$$$ \\ \\ /$$$ |$$ | $$ |\\n");
    printf("$$_$ __$$< $$ | $$$ $ $ |$$ | $$ |$$$$ /$$$$ |$$ | $$ |\\n");
    printf("$$_$ |__$$_$ | _$$ | _$$ |$$$$ |$$ \\ \\ $$ |$$ |$$$ /$ $ |$$ \\ \\ _$$ |\\n");
    printf("$$_$ __$$ / _$$ | $$$ | $$$ |$$ __$$ /$ $ | $ / $ $ |$ $ __$$ /\\n");
    printf("$$$$$$$$$/ $$$$$$/ $$ / $$/ $$$$$$/ $$ / $$/ $$$$$$/ \\n");
}

int main()
{
    displaybinomo();
    printf("\\n");
    ArrayOfGame ListGame;
    MakeArrayOfGame (&ListGame);
```

```

Queue QueueGame;
CreateQueue(&QueueGame);
boolean cek = false;
while (!cek)
{
    printf("Masukkan Command (START / LOAD <file_name>): ");
    STARTINPUTKATA();
    if (IsEqual(takeword(currentWord, 1), "LOAD"))
    {
        if (currentWord.Length == 4)
        {
            printf("Tolong masukkan nama file!\n");
        }
        else
        {
            Word x = takeword(currentWord, 2);
            char *path = "Data/";
            char *newfile = ConcateChar(path, WordToString(x));
            FILE *p;
            p = fopen(newfile, "r");
            if (p == NULL) {
                printf("Masukkan file dengan benar\n");
            }
            else {
                load(WordToString(x), &ListGame);
                cek = true;
                fclose(p);
            }
        }
    }
    else if (IsEqual(takeword(currentWord, 1), "START"))
    {
        printf("Selamat datang di Binomo!\n");
        start(&ListGame);
        cek = true;
    }
    else
    {
        printf("Masukkan command START atau LOAD di awal permainan\n");
    }
    while (cek)
    {
        printf("\nKetik HELP untuk melihat list command yang dapat
digunakan\n");
        printf("Masukkan Command: ");
        STARTINPUTKATA();
        if (IsEqual(currentWord, "CREATE GAME"))
        {
            createGame(&ListGame);
        }
        else if (IsEqual(currentWord, "LIST GAME"))
        {
            listGame(&ListGame);
        }
        else if (IsEqual(currentWord, "DELETE GAME"))
        {
            deleteGame(&ListGame, QueueGame);
        }
        else if (IsEqual(currentWord, "QUEUE GAME"))
        {
            queueGame(&QueueGame, ListGame);
        }
    }
}

```

```

        else if (IsEqual(currentWord, "PLAY GAME"))
        {
            playGame(&QueueGame);
        }
        else if (IsEqual(takeword(currentWord, 1), "SKIP GAME")) // ini belum
        {
            skipGame(WordToInt(takeword(currentWord, 3)), &QueueGame);
            if (takeword(currentWord, 3).Length == 0)
            {
                printf("Tolong masukkan nomor game yang ingin di skip dengan
format 'SKIPGAME <n>'!\n");
            }
        }
        else if (IsEqual(currentWord, "HELP"))
        {
            help();
        }
        else if (IsEqual(takeword(currentWord, 1), "SAVE"))
        {
            if (currentWord.Length == 4)
            {
                printf("Tolong masukkan nama file!\n");
            }
            else
            {
                Word x = takeword(currentWord, 2);
                save(WordToString(x), ListGame);
            }
        }
        else if (IsEqual(currentWord, "QUIT"))
        {
            quit(ListGame);
        }
        else
        {
            otherCommand();
        }
    }
}
return 0;
}

```

Program utama dimulai dengan pendefinisian header yang boleh digunakan untuk spesifikasi yaitu <stdio.h>, <stdlib.h>, <time.h>, <math.h> dan semua header dari semua fungsi/prosedur maupun ADT yang telah dibuat. Terdapat enam header yang telah dibuat yaitu game_util.h, diner_dash.h, random_number_generator.h, game_tambahan.h, hangman.h, main_util.h.

Setelah pendefinisian semua header yang dibutuhkan, dilanjutkan oleh pembuatan fungsi displaybinomo() yang berfungsi sebagai header selamat datang saat baru memulai program, sesuai yang terlampir pada bagian 2.1.

Kemudian masuk ke bagian utama dari program int main() yang di dalamnya diawali dengan pemanggilan fungsi displaybinomo(). Sebelum memanggil fungsi/prosedur tertentu, diperlukan adanya deklarasi parameter yaitu ListGame berbentuk ArrayOfGame yang dipakai untuk MakeArrayOfGame, QueueGame berbentuk Queue yang dipakai untuk CreateQueue, dan pendefinisial boolean cek. Validasi ini kemudian dipakai untuk menentukan looping menuju program utama.

Setelah memenuhi syarat looping, akan dicetak sebuah perintah untuk memasukan command untuk START atau LOAD guna menjalankan program utama. Pembacaan command dari user kemudian dilakukan dengan ADT mesin kata yang dilakukan sesuai dengan arahan spesifikasi. Jika user memasukan command LOAD maka harus diikuti nama file yang kemudian akan dibaca sistem. Jika memasukan command START, user akan langsung bisa mengakses semua command yang terdapat di BNMO yaitu CREATE GAME, LIST GAME, DELETE GAME, QUEUE GAME, PLAY GAME, SKIP GAME, HELP, SAVE, dan QUIT. Dari masing-masing command yang berhasil dibaca, akan langsung memanggil masing-masing fungsi.

5 Algoritma menarik

Tidak ada algoritma yang menarik. Kita doang yang menarik.

6 Data Test

Berikut data data test yang terdapat dalam program

6.1 START

Program akan mulai membaca file konfigurasi default sistem yaitu config.txt dan akan menambahkan game game default di list game. Jika berhasil program menuliskan pesan “File Konfigurasi sistem berhasil dibaca. BNMO berhasil dijalankan”

```

$$$$$$ \ /$$$$$ / \ / \ /$$$$$ \ $ / \ /$$$$$ \
$$ |__$$ | $$ | $$$ \ $$ | $ $ |$$$ \ /$$$ |$ | $
$$ $ $< $ $ | $$$ $ $ |$ | $ $ |$$$ /$$$ |$ | $
$$ |__$$ | _$$ | _$ |$$$ |$ \_$$ |$ |$$$ /$ |$ \_$$ |
$$ $ $ / / $ $ |$ $ |$ $ |$ $ /$ $ | /$ $ |$ $ $ $ /
$$$$$$ / $$$$$ / $ / $ / $$$$$ / $ / $ / $$$$$ /
Masukkan Command (START / LOAD <file_name>): |

```

Gambar 6.1.1 Tampilan main ketika START dijalankan

```

$$$$$$ \ /$$$$$ / \ / \ /$$$$$ \ $ / \ /$$$$$ \
$$ |__$$ | $$ | $$$ \ $$ | $ $ |$$$ \ /$$$ |$ | $
$$ $ $< $ $ | $$$ $ $ |$ | $ $ |$$$ /$$$ |$ | $
$$ |__$$ | _$$ | _$ |$$$ |$ \_$$ |$ |$$$ /$ |$ \_$$ |
$$ $ $ / / $ $ |$ $ |$ $ |$ $ /$ $ | /$ $ |$ $ $ $ /
$$$$$$ / $$$$$ / $ / $ / $$$$$ / $ / $ / $$$$$ /
Masukkan Command (START / LOAD <file_name>): START U
Masukkan command START atau LOAD di awal permainan

```

Gambar 6.1.2 Input salah pada start

```

Masukkan Command (START / LOAD <file_name>): ST
Masukkan command START atau LOAD di awal permainan
Masukkan Command (START / LOAD <file_name>): START
Selamat datang di Binomo!
File Konfigurasi sistem berhasil dibaca. BNMO berhasil dijalankan
Ketik HELP untuk melihat list command yang dapat digunakan

```

Gambar 6.1.3. Input sesuai

```

config.txt
7
RNG
OTHER DASH
DINOSAUR IN EARTH
RESEKMAN
EIFFEL TOWER
HANGMAN
TIC TAC TOE

```

Gambar 6.1.4 File config.txt

6.2 LOAD

Program akan membaca save file yang berisi game game yang akan dimasukkan ke list game. Program akan membaca 2 parameter input yaitu kata “LOAD” dan nama file yang akan dibaca. Jika user hanya menginput satu kata “LOAD” tanpa menginput nama file akan keluar pesan “Tolong masukkan nama file”. Jika file yang diinput user salah maka akan keluar pesan “Masukkan file dengan benar”. Jika berhasil maka keluar pesan “Save file berhasil dibaca. BNMO berhasil dijalankan”.

```

/-----\ /-----\ /-----\ /-----\ /-----\ /-----\ /-----\ /-----\
$$$$$$$ |$$$$$/ $$ \ $$ /$$$$$ |$$ \ $$ /$$$$$ |$$$$$/
$$ |__$$ | $$ | $$$ \ $$ |$$ | $$ |$$$ \ /$$$ |$$ | $$ |
$$ $$$< $$ | $$$ $ $ |$$ | $$ |$$$ /$$$ |$$ | $$ |
$$ |__$$ | _$$ | _$$ |$$$ |$$ \_$$ |$$ |$$$/ $$ |$$ \_$$ |
$$ $$$/ / $$ |$$ | $$$ |$$ $$$/ $$ | / $$ |$$ $$$/
$$$$$$$/ $$$$/ $$$/ $$$/ $$$$/ $$$/ $$$/ $$$$/

Masukkan Command (START / LOAD <file_name>): LOAD
Tolong masukkan nama file!
Masukkan Command (START / LOAD <file_name>): |
```

Gambar 6.2.1 Tampilan ketika hanya input “LOAD” tanpa nama file

```

/-----\ /-----\ /-----\ /-----\ /-----\ /-----\ /-----\ /-----\
$$$$$$$ |$$$$$/ $$ \ $$ /$$$$$ |$$ \ $$ /$$$$$ |$$$$$/
$$ |__$$ | $$ | $$$ \ $$ |$$ | $$ |$$$ \ /$$$ |$$ | $$ |
$$ $$$< $$ | $$$ $ $ |$$ | $$ |$$$ /$$$ |$$ | $$ |
$$ |__$$ | _$$ | _$$ |$$$ |$$ \_$$ |$$ |$$$/ $$ |$$ \_$$ |
$$ $$$/ / $$ |$$ | $$$ |$$ $$$/ $$ | / $$ |$$ $$$/
$$$$$$$/ $$$$/ $$$/ $$$/ $$$$/ $$$/ $$$/ $$$$/

Masukkan Command (START / LOAD <file_name>): LOAD uwu
Masukkan file dengan benar
```

Gambar 6.2.2 Tampilan ketika input nama file salah

```

Masukkan Command (START / LOAD <file_name>): LOAD savefile.txt
Save file berhasil dibaca. BNMO berhasil dijalankan.

Ketik HELP untuk melihat list command yang dapat digunakan
Masukkan Command: |
```

Gambar 6.2.3 Tampilan ketika berhasil

6.3 SAVE

Program akan menyimpan state game pemain saat ini ke dalam suatu file. Command SAVE memiliki satu argumen yang merepresentasikan nama file yang akan disimpan pada disk. Jika nama file yang diinput belum ada pada folder maka program akan otomatis membuat file pada folder Data. Setelah berhasil maka program akan mengeluarkan pesan “Save file berhasil disimpan”

```

Ketik HELP untuk melihat list command yang dapat digunakan
Masukkan Command: SAVE
Tolong masukkan nama file!
```

Gambar 6.3.1 Tampilan program saat SAVE tanpa nama file

```

Ketik HELP untuk melihat list command yang dapat digunakan
Masukkan Command: SAVE savefile.txt
Save file berhasil disimpan.
```

Gambar 6.3.2 Tampilan program saat SAVE dengan nama file yang benar



Gambar 6.3.4 Tampilan file sebelum di-save



Gambar 6.3.5 Tampilan file setelah di save

6.4 CREATE GAME

Program akan meminta input user untuk nama game yang akan ditambahkan ke List Game.

```
Ketik HELP untuk melihat list command yang dapat digunakan
Masukkan Command: CREATE GAME
Masukkan nama game yang akan ditambahkan: 
```

Gambar 6.4.1 Tampilan program saat command CREATE GAME

```
Ketik HELP untuk melihat list command yang dapat digunakan
Masukkan Command: CREATE GAME
Masukkan nama game yang akan ditambahkan: DINER DASH
Nama game tersebut sudah ada dalam list. Game gagal ditambahkan.
```

Gambar 6.4.2 Tampilan program saat game yang dibuat sudah ada di dalam file konfigurasi

```
Ketik HELP untuk melihat list command yang dapat digunakan
Masukkan Command: CREATE GAME
Masukkan nama game yang akan ditambahkan: gameahahaha
Game berhasil ditambahkan
```

Gambar 6.4.3 Tampilan program saat game yang dibuat belum ada di dalam file konfigurasi

```
Ketik HELP untuk melihat list command yang dapat digunakan
Masukkan Command: LIST GAME
Berikut adalah daftar game yang tersedia
1. RNG
2. DINER DASH
3. DINOSAUR IN EARTH
4. RISEWOMAN
5. EIFFEL TOWER
6. HANGMAN
7. TIC TAC TOE
8. gameahahaha
```

Gambar 6.4.4 Tampilan list game setelah ditambahkan

6.5 LIST GAME

Program akan menampilkan daftar game yang disediakan oleh sistem.

```
Ketik HELP untuk melihat list command yang dapat digunakan
Masukkan Command: LIST GAME
Berikut adalah daftar game yang tersedia
1. RNG
2. DINER DASH
3. DINOSAUR IN EARTH
4. RISEWOMAN
5. EIFFEL TOWER
6. HANGMAN
7. TIC TAC TOE
```

Gambar 6.5.1 Tampilan List Game

6.6 DELETE GAME

Program akan meminta input dari user untuk nomor game yang akan dihapus sesuai aturan. Game yang terdapat pada konfigurasi sistem default tidak dapat dihapus.

```
Ketik HELP untuk melihat list command yang dapat digunakan
Masukkan Command: DELETE GAME
Berikut adalah daftar game yang tersedia
1. RNG
2. DINER DASH
3. DINOSAUR IN EARTH
4. RISEWOMAN
5. EIFFEL TOWER
6. HANGMAN
7. TIC TAC TOE
8. uwu
Masukkan nomor yang akan dihapus : 
```

Gambar 6.6.1 Tampilan Delete Game

```
Ketik HELP untuk melihat list command yang dapat digunakan
Masukkan Command: DELETE GAME
Berikut adalah daftar game yang tersedia
1. RNG
2. DINER DASH
3. DINOSAUR IN EARTH
4. RISEWOMAN
5. EIFFEL TOWER
6. HANGMAN
7. TIC TAC TOE
8. uwu
Masukkan nomor yang akan dihapus :9
Maaf, nomor yang Anda masukkan melebihi jumlah game yang ada.
```

Gambar 6.6.2 Tampilan program jika nomor game tidak valid

```
Ketik HELP untuk melihat list command yang dapat digunakan
Masukkan Command: DELETE GAME
Berikut adalah daftar game yang tersedia
1. RNG
2. DINER DASH
3. DINOSAUR IN EARTH
4. RISEWOMAN
5. EIFFEL TOWER
6. HANGMAN
7. TIC TAC TOE
8. uwu
Masukkan nomor yang akan dihapus :1
Game tidak dapat dihapus
```

Gambar 6.6.3 Tampilan program jika game merupakan game default

```
Ketik HELP untuk melihat list command yang dapat digunakan
Masukkan Command: DELETE GAME
Berikut adalah daftar game yang tersedia
1. RNG
2. DINER DASH
3. DINOSAUR IN EARTH
4. RISEWOMAN
5. EIFFEL TOWER
6. HANGMAN
7. TIC TAC TOE
8. uwu
Masukkan nomor yang akan dihapus :8
Game berhasil dihapus
```

Gambar 6.6.4 Tampilan program jika nomor game valid

6.7 QUEUE GAME

Program akan meminta input dari user untuk nomor game yang akan ditambahkan ke queue game sebelum dimainkan.

Gambar 6.7.1 Tampilan program ketika nomor game valid

```

Ketik HELP untuk melihat list command yang dapat digunakan
Masukkan Command: QUEUE GAME
Berikut adalah daftar antrian game-mu sekarang:

Berikut adalah daftar game yang tersedia
1. RNG
2. DINER DASH
3. DINOSAUR IN EARTH
4. RISEWOMAN
5. EIFFEL TOWER
6. HANGMAN
7. TIC TAC TOE

Nomor game yang akan ditambahkan ke antrian: 8
Nomor game yang Anda masukkan tidak valid. Mohon ulangi.
Nomor game yang akan ditambahkan ke antrian: █

```

Gambar 6.7.2 Tampilan program ketika nomor game tidak valid

6.8 PLAY GAME

Program akan memanggil fungsi pada game untuk memainkan game yang berada pada head di queue game.

```

Ketik HELP untuk melihat list command yang dapat digunakan
Masukkan Command: PLAY GAME
Berikut adalah daftar antrian game-mu sekarang:
1. DINOSAUR IN EARTH
2. RISEWOMAN
3. EIFFEL TOWER
4. DINOSAUR IN EARTH
Game DINOSAUR IN EARTH masih dalam maintenance, belum dapat dimainkan. Silakan p
ilih game lain.

```

Gambar 6.8.1 Tampilan program ketika play game

```

Ketik HELP untuk melihat list command yang dapat digunakan
Masukkan Command: PLAY GAME
Berikut adalah daftar antrian game-mu sekarang:
Maaf, antrian game-mu kosong. Silakan menambahkan game ke antrian terlebih dahul
u dengan mengetik command: QUEUE GAME

```

Gambar 6.8.2 Tampilan program ketika tidak ada game di dalam antrian

6.9 SKIP GAME

Program akan memanggil fungsi pada game untuk memainkan game ke $n + 1$ pada queue game. Di mana n merupakan input dari user.

```

Ketik HELP untuk melihat list command yang dapat digunakan
Masukkan Command: SKIP GAME 2
Berikut adalah daftar antrian game-mu sekarang:
1. DINOSAUR IN EARTH
2. RISEWOMAN
3. EIFFEL TOWER

Game EIFFEL TOWER masih dalam maintenance, belum dapat dimainkan. Silakan pilih
game lain.
Sekarang antrian game-mu adalah :

Ketik HELP untuk melihat list command yang dapat digunakan
Masukkan Command: █

```

Gambar 6.9.1 Tampilan program saat input valid

```

Ketik HELP untuk melihat list command yang dapat digunakan
Masukkan Command: SKIP GAME 5
Berikut adalah daftar antrian game-mu sekarang:
1. DINOSAUR IN EARTH
2. RISEWOMAN
3. EIFFEL TOWER

Maaf, jumlah game yang ingin dilewati melebihi jumlah game yang ada di antrian.
Silakan coba lagi.
Sekarang antrian game-mu adalah :
1. DINOSAUR IN EARTH
2. RISEWOMAN
3. EIFFEL TOWER

Ketik HELP untuk melihat list command yang dapat digunakan
Masukkan Command: █

```

Gambar 6.9.2 Tampilan program saat input lebih banyak dari antrian game

```

Ketik HELP untuk melihat list command yang dapat digunakan
Masukkan Command: SKIP GAME 1
Berikut adalah daftar antrian game-mu sekarang:

Tidak ada permainan lagi dalam daftar game-mu.
Sekarang antrian game-mu adalah :

Ketik HELP untuk melihat list command yang dapat digunakan
Masukkan Command:

```

Gambar 6.9.3 Tampilan program saat tidak ada game di dalam antrian

6.10 QUIT

Program akan keluar dan meminta input dari user apakah user ingin save atau tidak list game yang ada dalam program.

```

Ketik HELP untuk melihat list command yang dapat digunakan
Masukkan Command: QUIT
Apakah kamu ingin save game kamu? (Y/N)
Y
Masukkan nama file penyimpanan:

```

Gambar 6.10.1 Tampilan program saat user quit dan save

```

Ketik HELP untuk melihat list command yang dapat digunakan
Masukkan Command: QUIT
Apakah kamu ingin save game kamu? (Y/N)
N
Game has ended
Thank you for playing BNMO

```

Gambar 6.10.2 Tampilan program saat user quit dan tidak save

6.11 HELP

Program akan menampilkan kata kata yang berisi deskripsi dari command command yang ada untuk membantu user

```

Ketik HELP untuk melihat list command yang dapat digunakan
Masukkan Command: HELP
Hai Hai Jangan Merasa Tersesat,
Berikut adalah hal-hal yang bisa kamu lakukan :
01. SAVE <filename> : untuk menyimpan state game pemain saat ini ke dalam suatu file
02. CREATE GAME      : untuk menambahkan game baru pada daftar game
03. LIST GAME        : untuk menampilkan daftar game yang disediakan oleh sistem
04. DELETE GAME      : untuk menghapus sebuah game dari daftar game
05. QUEUE GAME       : untuk mendaftarkan permainan kedalam list
06. PLAY GAME        : untuk memainkan sebuah permainan
07. SKIPGAME <n>     : untuk melewati n permainan yang ada di dalam list
07. QUIT             : untuk keluar dari program
08. HELP             : untuk melihat informasi dari command-command

Ketik HELP untuk melihat list command yang dapat digunakan
Masukkan Command:

```

Gambar 6.11.1 Tampilan Help pada program

6.12 COMMAND LAIN

Program akan menampilkan pesan “Command yang diberikan tidak sesuai yang diinginkan, silahkan masukan input yang valid”.

```

Ketik HELP untuk melihat list command yang dapat digunakan
Masukkan Command: Q
Command yang diberikan tidak sesuai yang diinginkan.
Silahkan masukan input yang valid.

Ketik HELP untuk melihat list command yang dapat digunakan
Masukkan Command:

```

Gambar 6.12.1 Tampilan program saat command tidak valid

6.13 DINER DASH

Diner dash adalah game dimana user dapat menggunakan 3 command, yaitu cook, serve, dan skip.

1. Cook adalah command untuk memasak makanan
 2. Serve adalah command untuk menyajikan makanan
 3. Skip adalah command untuk menyelesaikan 1 putaran tanpa melakukan apa-apa
- Permainan selesai jika antrian melebihi 7 pelanggan atau telah melayani 15 pelanggan. Score merupakan saldo akhir pemain,

```
Game yang dimainkan adalah Diner Dash
Selamat datang di Diner Dash

SALDO : 0

Daftar pesanan:
Makanan | Durasi memasak | Ketahanan | Harga
-----
M0      | 3                | 2        | 12800
M1      | 3                | 5        | 37000
M2      | 4                | 4        | 44300

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

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

Masukkan permintaan ('COOK / SERVE' + ' ' + M {indeks yang ada pada daftar pesan
an}) / 'SKIP': COOK M0
```

Gambar 6.13.1 Tampilan awal program Diner Dash

```
Masukkan permintaan ('COOK / SERVE' + ' ' + M {indeks yang ada pada daftar pesan}) / 'SKIP': COOK M0
Makanan M0 telah dimasukan ke dalam antrian memasak

=====
SALDO : 0

Daftar pesanan:
Makanan | Durasi memasak | Ketahanan | Harga
-----
M0      | 3                | 2        | 12800
M1      | 3                | 5        | 37000
M2      | 4                | 4        | 44300
M3      | 3                | 1        | 26600

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

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

Masukkan permintaan ('COOK / SERVE' + ' ' + M {indeks yang ada pada daftar pesan}) / 'SKIP': █
```

Gambar 6.13.2 Tampilan masakan yang di cook valid

```
SALDO : 0

Daftar pesanan:
Makanan | Durasi memasak | Ketahanan | Harga
-----
M0      | 3                | 2        | 12800
M1      | 3                | 5        | 37000
M2      | 4                | 4        | 44300
M3      | 3                | 1        | 26600

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

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

Masukkan permintaan ('COOK / SERVE' + ' ' + M {indeks yang ada pada daftar pesan}) / 'SKIP': COOK M10
Makanan tidak ada dalam daftar pesanan
Masukkan permintaan ('COOK / SERVE' + ' ' + M {indeks yang ada pada daftar pesan}): █
```

Gambar 6.13.3 Tampilan masakan yang di cook tidak valid

```

Masukkan permintaan ('COOK / SERVE' + ' ' + M {indeks yang ada pada daftar pesanan}) / 'SKIP': SERVE M0
Makanan M0 telah disajikan

=====

SALDO : 12800

Daftar pesanan:
Makanan | Durasi memasak | Ketahanan | Harga
-----
M1      | 3                | 5        | 37000
M2      | 4                | 4        | 44300
M3      | 3                | 1        | 26600
M4      | 2                | 2        | 35100
M5      | 1                | 2        | 12700
M6      | 2                | 2        | 40000
M7      | 4                | 3        | 18500

Daftar Makanan yang sedang dimasak
Makanan | Sisa durasi memasak
-----
M2      | 2
M3      | 2

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

Masukkan permintaan ('COOK / SERVE' + ' ' + M {indeks yang ada pada daftar pesanan}) / 'SKIP':

```

Gambar 6.13.4 Tampilan masakan yang di serve valid

```

Masukkan permintaan ('COOK / SERVE' + ' ' + M {indeks yang ada pada daftar pesanan}) / 'SKIP': SERVE M10
Makanan tidak ada dalam daftar pesanan
Masukkan permintaan ('COOK / SERVE' + ' ' + M {indeks yang ada pada daftar pesanan}):

```

Gambar 6.13.5 Tampilan masakan yang di serve tidak valid

```

Daftar Makanan yang sedang dimasak
Makanan | Sisa durasi memasak
-----
M2      | 2
M3      | 2

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

Masukkan permintaan ('COOK / SERVE' + ' ' + M {indeks yang ada pada daftar pesanan}) / 'SKIP': SERVE M10
Makanan tidak ada dalam daftar pesanan
Masukkan permintaan ('COOK / SERVE' + ' ' + M {indeks yang ada pada daftar pesanan}): SERVE M2
Makanan M2 belum dapat disajikan karena M1 belum disajikan

```

Gambar 6.13.6 Tampilan masakan yang di serve belum seharusnya di serve

```

Makanan | Durasi memasak | Ketahanan | Harga
-----
M0      | 2              | 1         | 32500
M1      | 1              | 1         | 22100
M2      | 5              | 3         | 16200

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

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

Masukkan permintaan ('COOK / SERVE' + ' ' + M {indeks yang ada pada daftar pesanan}) / 'SKIP': SKIP
Kamu telah skip ronde kali ini

=====

SALDO : 0

Daftar pesanan:
Makanan | Durasi memasak | Ketahanan | Harga
-----
M0      | 2              | 1         | 32500
M1      | 1              | 1         | 22100
M2      | 5              | 3         | 16200
M3      | 3              | 1         | 10400

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

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

```

Gambar 6.13.7 Tampilan command skip

```

=====
Berikut hasil akhir daftar pesanan anda:
Makanan | Durasi memasak | Ketahanan | Harga
-----
M1      | 3              | 5         | 37000
M2      | 4              | 4         | 44300
M3      | 3              | 1         | 26600
M4      | 2              | 2         | 35100
M5      | 1              | 2         | 12700
M6      | 2              | 2         | 40000
M7      | 4              | 3         | 18500
M8      | 4              | 2         | 32700

Congratulation!
You've served 1 food
Game Over

Ketik HELP untuk melihat list command yang dapat digunakan
Masukkan Command:

```

Gambar 6.13.8 Tampilan saat game sudah gameOver

6.14 RNG

Game Random Number Generator, di mana user menebak angka yang sudah ditentukan di awal secara random. Program akan mengeluarkan pesan “Lebih besar” atau “Lebih kecil” sampai user berhasil menebak angka dengan benar.

```
Game yang dimainkan adalah RNG
RNG Telah dimulai. Uji keberuntungan Anda dengan menebak X.
Tebakan: 
```

Gambar 6.14.1 Tampilan game RNG

```
1. RNG
Game yang dimainkan adalah RNG
RNG Telah dimulai. Uji keberuntungan Anda dengan menebak X.
Tebakan: 10
Lebih kecil
Tebakan: 3
Lebih besar
Tebakan: 5
Lebih kecil
Tebakan: 4

Ya, X adalah 4
--- GAME OVER ---
Ketik HELP untuk melihat list command yang dapat digunakan
Masukkan Command: 
```

6.14.2 Tampilan game RNG setelah selesai

6.15 TICTACTOE

Game TicTacToe berisi board yang berisi 3 x 3 di mana user dengan simbol “O” melawan komputer “X”.

```
1. TIC TAC TOE
Game yang dimainkan adalah Tic Tac Toe

===== WELCOME TO TIC TAC TOE =====

Anda : 'O' || Com : 'X'

 1 | 2 | 3
--|---|
 4 | 5 | 6
--|---|
 7 | 8 | 9
--|---|

Masukkan nomor kotak: 
```

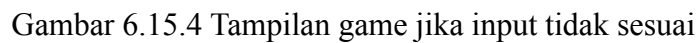
Gambar 6.15.1 Tampilan game saat user memasukan di board

```
Masukkan nomor kotak: 4

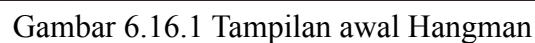
 1 | 2 | X
--|---|
 0 | 0 | 0
--|---|
 7 | X | 9
--|---|

====Selamat Anda menang====
Skor yang Anda peroleh adalah: 18
```

Gambar 6.15.2 Tampilan game saat user menang

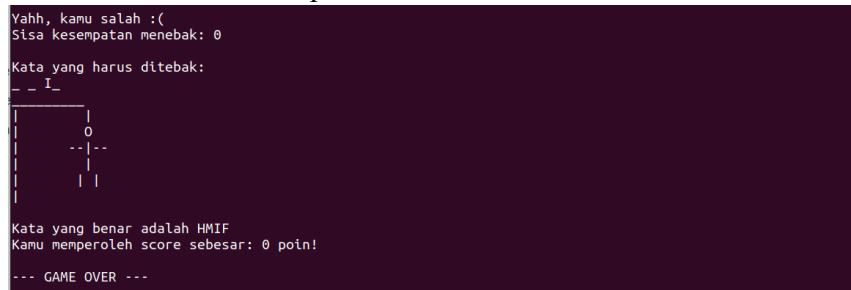


Hangman adalah game dimana user mengisi dengan huruf yang sesuai dengan kata-kata yang telah disediakan. Game selesai saat jumlah try habis atau huruf dari kata tersebut telah terisi semua.

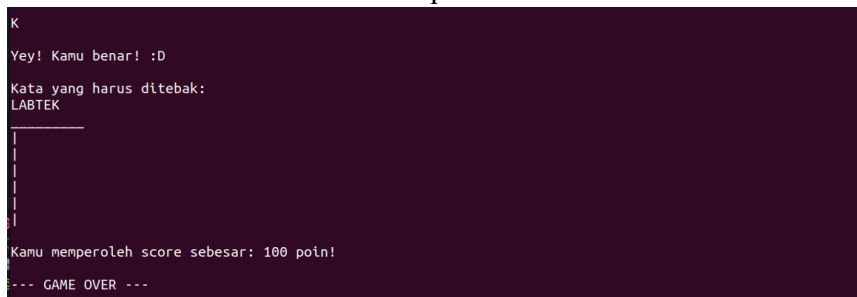




Gambar 6.16.2 Tampilan saat huruf tidak ada di dalam kata



Gambar 6.16.3 Tampilan saat telah kalah



Gambar 6.16.4 Tampilan saat menang dari game

7 Test Script

No.	Fitur yang Dites	Tujuan Testing	Langkah-Langkah Testing	Input Data Test	Hasil yang Diharapkan	Hasil yang Keluar
1	START	Memastikan prosedur START bisa menjalankan keseluruhan program BNMO	Memasukan command START saat program berhasil di compile	Gambar 6.1.3.	Gambar 6.1.3.	Gambar 6.1.3.
2	START	Memastikan prosedur START gagal jika input salah	Memasukan command START yang salah saat program berhasil di compile	Gambar 6.1.2.	Gambar 6.1.3.	Gambar 6.1.2.
3	LOAD	Memastikan prosedur bisa membaca file yang merupakan bentuk dari game yang bisa dimainkan.	Memasukan command LOAD diikuti file saat program berhasil di compile	Gambar 6.2.3	Gambar 6.2.3	Gambar 6.2.3

		history, dan scoreboard				
4	LOAD	Memastikan prosedur LOAD gagal jika input salah	Memasukan command LOAD tanpa diikuti nama file	Gambar 6.2.1	Gambar 6.2.3	Gambar 6.2.1
5	LOAD	Memastikan prosedur LOAD gagal jika input salah	Memasukan command LOAD diikuti nama file yang salah	Gambar 6.2.2	Gambar 6.2.3	Gambar 6.2.2
6	SAVE	Memastikan prosedur bisa menyimpan state game	Memasukan command SAVE diikuti nama file yang benar	Gambar 6.3.2	Gambar 6.3.2	Gambar 6.3.2
7	SAVE	Memastikan prosedur tidak berjalan saat input salah	Memasukan command SAVE tanpa diikuti nama file	Gambar 6.3.1	Gambar 6.3.2	Gambar 6.3.1
8	CREATE GAME	Memastikan prosedur bisa membuat game baru	Memasukan command CREATE GAME saat program dijalankan	Gambar 6.4.1	Gambar 6.4.1	Gambar 6.4.3
9	CREATE GAME	Memastikan prosedur gagal saat input salah	Memasukan command CREATE GAME diikuti game yang sudah ada sebelumnya	Gambar 6.4.1	Gambar 6.4.1	Gambar 6.4.2
10	LIST GAME	Memastikan prosedur bisa menampilkan list game yang tersedia	Memasukan command LIST GAME saat program dijalankan	Gambar 6.5.1	Gambar 6.5.1	Gambar 6.5.1
11	DELETE GAME	Memastikan prosedur bisa menghapus game	Memasukan command DELETE GAME saat program dijalankan	Gambar 6.6.1	Gambar 6.6.4	Gambar 6.6.4
12	DELETE GAME	Memastikan prosedur gagal jika nomor game yang dimasukan tidak valid	Memasukan command DELETE GAME lalu menghapus game dengan urutan yang tidak valid	Gambar 6.6.1	Gambar 6.6.4	Gambar 6.6.2
13	DELETE GAME	Memastikan prosedur gagal jika game yang dihapus game default	Memasukan command DELETE GAME lalu menghapus game default	Gambar 6.6.1	Gambar 6.6.4	Gambar 6.6.3
14	QUEUE GAME	Memastikan prosedur bisa memasukan game ke antrian game	Memasukan command QUEUE GAME saat program dijalankan	Gambar 6.7.1	Gambar 6.7.1	Gambar 6.7.1
15	QUEUE GAME	Memastikan prosedur tidak berjalan saat input tidak valid	Memasukan command QUEUE GAME dan memasukan nomor yang tidak valid	Gambar 6.7.2	Gambar 6.7.1	Gambar 6.7.2
16	PLAY GAME	Memastikan prosedur bisa memainkan game	Memasukan command PLAY GAME saat program dijalankan	Gambar 6.8.1	Gambar 6.8.1	Gambar 6.8.1

17	PLAY GAME	Memastikan prosedur tidak berjalan saat tidak ada antrian game	Memasukan command PLAY GAME saat tidak ada antrian game	Gambar 6.8.2	Gambar 6.8.1	Gambar 6.8.2
18	SKIP GAME	Memastikan prosedur bisa melewati antrian game	Memasukan command SKIP GAME saat program dijalankan	Gambar 6.9.1	Gambar 6.9.1	Gambar 6.9.1
19	SKIP GAME	Memastikan prosedur gagal saat input lebih banyak dari antrian	Memasukan command SKIP GAME dengan n lebih besar daripada jumlah antrian	Gambar 6.9.2	Gambar 6.9.1	Gambar 6.9.2
20	SKIP GAME	Memastikan prosedur gagal saat tidak ada antrian game	Memasukan command SKIP GAME saat tidak ada antrian game	Gambar 6.9.3	Gambar 6.9.1	Gambar 6.9.3
21	QUIT	Memastikan prosedur bisa mengeluarkan program dan menyimpan data	Memasukan command QUIT GAME lalu men-savenya	Gambar 6.10.1	Gambar 6.10.1	Gambar 6.10.1
22	QUIT	Memastikan prosedur bisa mengeluarkan program tanpa menyimpan data	Memasukan command QUIT GAME tanpa menyimpan data	Gambar 6.10.2	Gambar 6.10.2	Gambar 6.10.2
23	HELP	Memastikan prosedur bisa mengeluarkan list bantuan untuk user	Memasukan command HELP saat program dijalankan	Gambar 6.11.1	Gambar 6.11.1	Gambar 6.11.1
24	COMMAND LAIN	Memastikan prosedur mengirimkan pesan kesalahan	Memasukan command yang salah saat program dijalankan	Gambar 6.12.1	Gambar 6.12.1	Gambar 6.12.1
25	DINER DASH	Memastikan game Diner Dash bisa dijalankan dengan baik	Memasukan game Diner Dash kedalam QUEUE GAME di urutan paling atas	Gambar 6.13.1	Gambar 6.13.1	Gambar 6.13.1
26	DINER DASH	Memastikan masakan yang valid	Memainkan Diner Dash lalu memasukan COOK yang valid	Gambar 6.13.2	Gambar 6.13.2	Gambar 6.13.2
27	DINER DASH	Memastikan masakan yang tidak valid	Memainkan Diner Dash lalu memasukan COOK yang tidak valid	Gambar 6.13.3	Gambar 6.13.2	Gambar 6.13.3
28	DINER DASH	Memastikan serve yang valid	Memainkan Diner Dash lalu memasukan SERVE yang valid	Gambar 6.13.4	Gambar 6.13.4	Gambar 6.13.4
29	DINER DASH	Memastikan serve yang tidak valid	Memainkan Diner Dash lalu memasukan SERVE tidak yang valid	Gambar 6.13.5	Gambar 6.13.4	Gambar 6.13.5
30	DINER DASH	Memastikan masakan serve yang harusnya belum serve	Memainkan Diner Dash lalu memasukan masakan SERVE yang sebelumnya belum serve	Gambar 6.13.6	Gambar 6.13.4	Gambar 6.13.6
31	DINER DASH	Memastikan fitur skip pada Diner Dash	Memainkan Diner Dash lalu memasukan skip	Gambar 6.13.7	Gambar 6.13.7	Gambar 6.13.7

32	DINER DASH	Memastikan GAME OVER	Memainkan Diner Dash lalu menamatkan permainan	Gambar 6.13.8	Gambar 6.13.8	Gambar 6.13.8
33	RNG	Memastikan RNG telah dimulai	Memasukan game RNG kedalam QUEUE GAME di urutan paling atas	Gambar 6.14.1	Gambar 6.14.1	Gambar 6.14.1
34	RNG	Memastikan RNG telah selesai	Menyelesaikan tebakan	Gambar 6.14.2	Gambar 6.14.2	Gambar 6.14.2
35	TICTACTOE	Memastikan game Tic Tac Toe bisa dijalankan dengan baik	Memasukan game Tic Tac Toe kedalam QUEUE GAME di urutan paling atas	Gambar 6.15.1	Gambar 6.15.1	Gambar 6.15.1
36	TICTACTOE	Memastikan game Tic Tac Toe saat menang	Memainkan Tic Tac Toe lalu memenangkannya	Gambar 6.15.2	Gambar 6.15.2	Gambar 6.15.2
37	TICTACTOE	Memastikan game Tic Tac Toe saat kalah	Memainkan Tic Tac Toe lalu mencoba kalah	Gambar 6.15.3	Gambar 6.15.3	Gambar 6.15.3
38	TICTACTOE	Memastikan game Tic Tac Toe saat input salah	Memainkan Tic Tac Toe lalu mencoba inputan salah	Gambar 6.15.4	Gambar 6.15.4	Gambar 6.15.4
39	TICTACTOE	Memastikan game Tic Tac Toe saat seri	Memainkan Tic Tac Toe lalu seri	Gambar 6.15.5	Gambar 6.15.5	Gambar 6.15.5
40	HANGMAN	Memastikan game Hangman bisa dijalankan dengan baik	Memasukan game Hangman kedalam QUEUE GAME di urutan paling atas	Gambar 6.16.1	Gambar 6.16.1	Gambar 6.16.1
41	HANGMAN	Memastikan game Hangman saat tidak ada huruf yang benar	Memainkan Hangman lalu menebak huruf yang salah	Gambar 6.16.2	Gambar 6.16.2	Gambar 6.16.1
42	HANGMAN	Memastikan game Hangman saat kalah	Memainkan Hangman lalu kalah	Gambar 6.16.3	Gambar 6.16.3	Gambar 6.16.3
43	HANGMAN	Memastikan game Hangman saat menang	Memainkan Hangman lalu menang	Gambar 6.16.4	Gambar 6.16.4	Gambar 6.16.4

8 Pembagian Kerja dalam Kelompok

Nama - NIM	Pembagian Tugas
Rahmah Putri Azzahra - 18219052	List Game, Queue Game, Delete Game, Start, Config File
Vincent Franstyo - 18221100	Quit, Diner Dash, Driver, ADT
Karina Rahadiani - 18221104	RNG, Skip Game, Help, Deskripsi

Christina Wijaya - 18221106	Diner Dash, Play Game, Save
Sultan Alta Alvaro Valencia - 18221110	RNG, Create Game, Load

9 Lampiran

9.1 Deskripsi Tugas Besar 1

Deskripsi Umum

Buatlah sebuah permainan berbasis CLI (command-line interface). Sistem ini dibuat dalam **bahasa C** dengan menggunakan **struktur data yang sudah kalian pelajari** di mata kuliah ini. Kalian boleh menggunakan (atau memodifikasi) struktur data yang sudah kalian buat untuk praktikum pada tugas besar ini. Library yang boleh digunakan hanya **stdio.h**, **stdlib.h**, **time.h** dan **math.h**.

Command

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.

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

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.

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

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.

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

d. **CREATEGAME**

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

```
ENTER COMMAND: CREATE GAME  
Masukkan nama game yang akan ditambahkan: EXTRA1  
Game berhasil ditambahkan
```

e. **LISTGAME**

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

```
ENTER COMMAND: LIST GAME  
Berikut adalah daftar game yang tersedia  
1. RNG  
2. LUNCH SLOW  
3. DINOSAUR IN EARTH  
4. RISEWOMAN  
5. EIFFEL TOWER
```

f. **DELETEGAME**

DELETEGAME 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.

```
ENTER COMMAND: DELETE GAME  
Berikut adalah daftar game yang tersedia  
1. RNG  
2. LUNCH SLOW  
3. DINOSAUR IN EARTH  
4. RISEWOMAN  
5. EIFFEL TOWER  
6. CUSTOM GAME 1  
Masukkan nomor game yang akan dihapus: 6  
  
Game berhasil dihapus
```

ENTER COMMAND: **DELETE GAME**

Berikut adalah daftar game yang tersedia

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

Masukkan nomor game yang akan dihapus: 1

Game gagal dihapus

g. **QUEUEGAME**

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

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: 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.

h. **PLAYGAME**

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.

```
ENTER COMMAND: PLAY GAME  
Berikut adalah daftar Game-mu  
1. EIFFEL TOWER  
2. RISEWOMAN  
3. LUNCH SLOW  
  
Loading EIFFEL TOWER ...
```

```
ENTER COMMAND: PLAY GAME  
Berikut adalah daftar Game-mu  
1. RISEWOMAN  
2. RISEWOMAN  
3. LUNCH SLOW  
  
Game RISEWOMAN masih dalam maintenance, belum dapat dimainkan. Silahkan  
pilih game lain.
```

i. **SKIPGAME <n>**

SKIPGAME merupakan command yang digunakan untuk melewati permainan sebanyak n.

```
ENTER COMMAND: SKIPGAME 2  
Berikut adalah daftar Game-mu  
1. RISEWOMAN  
2. LUNCH SLOW  
3. RISEWOMAN  
  
Loading RISEWOMAN ...
```

```
ENTER COMMAND: SKIPGAME 5  
Berikut adalah daftar Game-mu  
1. RISEWOMAN  
2. LUNCH SLOW  
3. RISEWOMAN  
  
Tidak ada permainan lagi dalam daftar game-mu.
```

j. **QUIT**

Keluar dari program.

```
ENTER COMMAND: QUIT
```



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

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.

```
ENTER COMMAND: COMMAND_ANEH  
  
Command tidak dikenali, silahkan masukkan command yang valid.  
ENTER COMMAND:
```

Spesifikasi Game

1. RNG

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

- Setiap permainan dimulai dengan program sudah menentukan sebuah angka acak X.
- Di setiap giliran, pemain diberi kesempatan menebak angka X. *Game* akan memberi tahu apakah tebakan pemain dibandingkan terhadap X lebih besar atau lebih kecil.
- Permainan selesai jika pemain menebak angka X dengan benar.
- Skor untuk *game* ini tergantung dengan seberapa cepat pemain menebak X. Formula skor dibebaskan.
- Batasan X dan maksimal giliran dibebaskan.

```
RNG Telah dimulai. Uji keberuntungan Anda dengan menebak X.  
Tebakan: 80  
Lebih kecil  
Tebakan: 40  
Lebih besar  
Tebakan: 41  
  
Ya, X adalah 41.
```

2. Diner Dash

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

permainan mengantar makanan namun terurut berdasarkan prioritasnya. Berikut adalah spesifikasi game ini:

- Terdapat 3 command yang dapat dilakukan pada game, yaitu **COOK** dan **SERVE**
 - **COOK** merupakan command yang bertujuan untuk memasak makanan
 - **SERVE** merupakan command yang bertujuan untuk menyajikan makanan kepada pelanggan.
 - **SKIP** merupakan command yang bertujuan untuk menyelesaikan 1 putaran tanpa melakukan apa apa (seluruh durasi memasak dan ketahanan berkurang 1, pelanggan bertambah 1).
- Command **selain COOK dan SERVE** dianggap tidak valid dan **tidak terhitung sebagai satu putaran**.
- Command **COOK dan SERVE yang tidak valid juga tidak terhitung sebagai satu putaran**
- Permainan akan dimulai dengan 3 pelanggan. Setiap pelanggan hanya dapat memesan satu makanan. Untuk setiap makanan, terdapat informasi tentang ID makanan yang dihasilkan secara *increment* (M01, M02, M03, dst), durasi memasak, harga makanan, serta ketahanan makanan. Semua informasi tersebut akan didapatkan secara random dengan menggunakan **random number generator**. Durasi dan ketahanan makanan akan berkisar diantara 1-5. Sedangkan, harga makanan akan berkisar diantara 10000 - 50000.
- Kapasitas dari pemain adalah memasak 5 makanan dalam waktu yang sama. Pelanggan yang dilayani adalah pelanggan yang duluan memasuki antrian.
- Permainan selesai apabila antrian melebihi 7 pelanggan atau jumlah pelanggan yang sudah dilayani mencapai 15 pelanggan.
- Pada setiap putaran, akan terdapat 1 pelanggan baru.
- Pada setiap putaran, seluruh durasi dari makanan yang sedang dimasak akan berkurang 1. Ketika durasi makanan mencapai 0, maka makanan sudah dapat di **SERVE**.
- Pada setiap putaran, seluruh ketahanan dari makanan yang sudah dapat disajikan akan berkurang 1. Ketika ketahanan makanan mencapai 0, maka makanan sudah akan hangus dan harus ulang dimasak.
- Ketika makanan sudah di **SERVE**, maka makanan dapat diantar kepada pelanggan dan pelanggan dapat meninggalkan antrian. Setelah pelanggan meninggalkan antrian, maka pemain akan menerima uang
- **SERVE** hanya dapat digunakan untuk pesanan yang berada di paling depan.
- Skor akhir dari pemain adalah total uang yang diterima oleh pemain.

Selamat Datang di Diner Dash!

SALDO: 0

Daftar Pesanan

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

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

Daftar Makanan yang sedang dimasak

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

|

Daftar Makanan yang dapat disajikan

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

|

MASUKKAN COMMAND: **COOK M0**

Berhasil memasak M0

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

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	

3. Game tambahan/ Buatan pemain

Game buatan pemain yang dibuat menggunakan command CREATE GAME akan langsung selesai dan masuk ke tahap game over dengan skor akhir berupa integer random.

9.2 Notulen Rapat

Rapat Pertama	
Tanggal	30/10/2022
<p>Meeting untuk membahas tubes secara keseluruhan, penentuan tanggal asistensi, dan pembagian tugas</p> <p>Pembagian tugas awal</p> <p>Paket 1: List game + Queue game, Delete game, Start, File konfigurasi → Rahma</p> <p>Paket 2: RNG, Create game, Load → Sul</p> <p>Paket 3: Dinner Dash, Play game, Save → Tina</p> <p>Paket 4: Dinner Dash, Quit, Driver → Vincent</p> <p>Paket 5: RNG, Skip game, Help → Ain</p> <p>Yang mau ditanyain :</p> <ul style="list-style-type: none"> - Mesin kata & karakter cuma baca dari file text atau ada yg dari input user juga? - Perbedaan play game & skip game karena sama2 bisa memulai game. Play game ngikutin queue atau ngga? - Perbedaan start & load, karena sama2 ngeload file konfigurasi. - Quit boleh langsung pake exit ga? <p>Asistensi : 02/11/22</p>	

Rapat Ke-dua	
Tanggal	02/11/2022
<p>Meeting lanjutan setelah asistensi pertama</p> <p>Pemrosesan command dari user harus menggunakan mesin kata dan mesin karakter tidak boleh menggunakan scanf sehingga diperlukan adanya ADT baru</p>	

Pemenuhan ADT akan dieksplor masing masing kemudian akan dibahas di meet selanjutnya

Pembagian tugas untuk laporan

Rapat Ke-tiga

Tanggal 06/11/2022

Meeting lanjutan untuk penentuan ADT mesin kata dan karakter

Mesin kata diproses dengan EOP \0

ADT mesin kata dan mesin karakter sudah siap digunakan untuk proses main.c

ADT mesin kata diakses kata kedua untuk skip game

Penentuan tanggal asistensi : 08/11/2022

Rapat Ke-empat

Tanggal 08/11/2022

Meeting offline untuk revisi

Revisi bersama fungsi fungsi yang masih error dan memperbaiki bersama-sama

Pembahasan hal yang perlu dikerjakan untuk finalisasi

- penambahan skip pada diner dash
- concate
- save
- revisi untuk enqueue dequeue

Rapat Ke-lima

Tanggal 08/11/2022

Rapat setelah asistensi untuk finalisasi

pembagian tugas finalisasi

- Benerin diner dash → Vincent
- Benerin displayqueuegame → Christina
- Tes program dengan semua kemungkinan → Rahma
- Buat IS FS dan comment2 yg diperlukan → Karina
- Buat validasi load & save di main → Sultan

9.3 Log Activity Anggota Kelompok

1. Rahmah Putri Azzahra/18219052

30 Oktober 2022 : Meet perdana kelompok
2 November 2022 : Menambah ADT ArrayOfGame
2 November 2022 : Menambahkan file config untuk game-game default
2 November 2022 : Membuat prosedur Delete Game dan Queue Game, dan List Game
2 November 2022 : Asistensi pertama
4 November 2022 : Memperbaiki ADT ArrayOfGame
5 November 2022 : Mengubah input di ArrayOfGame menjadi mesin kata
6 November 2022 : Rapat kelompok
8 November 2022 : Menambah prosedur start
8 November 2022 : Asistensi kedua
8 November 2022 : Membuat prosedur Copy Queue dan display Queue

2. Vincent Franstyo/18221100

30 Oktober 2022 : Meet perdana kelompok, membuat Github Repository
31 Oktober 2022 : Menambahkan ADT, docs, dan file-file yang diperlukan
1 November 2022 : Membuat struktur dari Game Diner Dash dan fungsi Quit
2 November 2022 : Menambahkan ADT food dan PrioQueue untuk Diner Dash
2 November 2022 : Memperbaiki input diner dash dan cook dan serve condition
2 November 2022 : Asistensi pertama
3 November 2022 : debug diner dash
4 November 2022 : menyelesaikan diner dash versi 1
6 November 2022 : Rapat kelompok
6 November 2022 : Menambah opsi untuk save di Quit, membuat makefile
8 November 2022 : Memperbaiki error skip game, help, dan main
8 November 2022 : Asistensi kedua
8 November 2022 : Membuat file readme, ADT List
9 November 2022 : mengganti ADT yang dipakai di diner dash menjadi Queue dan List
9 November 2022 : menyelesaikan diner dash kedua
10 November 2022 : debug diner dash
10 November 2022 : Laporan
11 November 2022 : Membuat driver ADT List dan PrioQueue
11 November 2022 : Laporan

3. Karina Rahadiani/18221104

30 Oktober 2022 : Meet perdana kelompok
2 November 2022 : Membuat Random Number Generator dan menambah deskripsi
2 November 2022 : Menambahkan Help function

2 November 2022 : Membuat prosedur Skip Game
 2 November 2022 : Asistensi pertama
 6 November 2022 : Rapat kelompok
 8 November 2022 : Memperbaiki skip game, displayQueueGame, dan Queue Game
 8 November 2022 : Asistensi kedua
 8 November 2022 : Memperbaiki Skip Game
 10 November 2022 : Menambah comment untuk seluruh fungsi dan prosedur
 10 November 2022 : Laporan

4. Christina Wijaya/18221106

30 Oktober 2022 : Meet perdana kelompok
 2 November 2022 : Menambahkan deskripsi dari game utilities
 2 November 2022 : Membuat fungsi Play Game
 2 November 2022 : Membuat prosedur Game Tambahan
 2 November 2022 : Memperbaiki nama-nama variabel
 2 November 2022 : Asistensi pertama
 5 November 2022 : Mengupdate ADT Mesin Kata
 6 November 2022 : Rapat kelompok
 8 November 2022 : Membuat prosedur save dan mengupdate prosedur Quit
 8 November 2022 : Asistensi kedua
 9 November 2022 : Merapikan isi program dan file” program
 9 November 2022 : Membuat game bonus (Hangman)
 10 November 2022 : debug diner dash
 10 November 2022 : membuat driver bagi ADT yang belum ada drivernya
 11 November 2022 : Laporan

5. Sultan Alta Alvaro Valencia/18221110

30 Oktober 2022 : Meet perdana kelompok
 2 November 2022 : modifikasi ADT mesin kata dan mesin karakter
 2 November 2022 : merevisi prosedur Game Tambahan, Create Game, Load
 2 November 2022 : debug RNG
 2 November 2022 : Asistensi pertama
 3 November 2022 : Modifikasi ADT Mesin kata
 4 November 2022 : Debug ADT Mesin Kata
 6 November 2022 : Rapat kelompok
 7 November 2022 : Menambah display Binomo, mengupdate List Game, membuat main program
 8 November 2022 : Memperbaiki ADT Queue
 8 November 2022 : Asistensi kedua
 8 November 2022 : validasi save dan load
 9 November 2022 : membuat Game bonus (TicTacToe)
 10 November 2022 : Debug TicTacToe
 11 November 2022 : Laporan