

LAPORAN TUGAS BESAR

IF2111 Algoritma dan Struktur Data

BNMO


Dipersiapkan oleh:

Kelompok 9 K02

18221044 Kinanti Wening Asih
18221058 Marvel Subekti
18221094 Raka Admiharfan Fatihah
18221120 Carissa Tabina Rianda
18221170 Amjad Adhie Prasetyo

Sekolah Teknik Elektro dan Informatika - Institut Teknologi Bandung

Jl. Ganesha 10, Bandung 40132

	Sekolah Teknik Elektro dan Informatika ITB	Nomor Dokumen		Halaman
		<i>IF2111-TB1-09</i>		22
		<i>Revisi</i>	<i>0</i>	<i>11 November 2022</i>

Daftar Isi

1 Ringkasan	3
2 Penjelasan Tambahan Spesifikasi Tugas	3
2.1 Spesifikasi Fungsi Pertambahan	3
2.2 Spesifikasi Fungsi Pengurangan	4
2.3 Spesifikasi Fungsi Perkalian	4
3 Struktur Data	4
3.1 ADT Array Dinamis	4
3.2 ADT Mesin Karakter	5
3.3 ADT Mesin Kata	5
3.4 ADT Queue	6
4 Program Utama	6
5 Algoritma-Algoritma Menarik	7
5.1 Konversi Tipe Data	7
5.2 Algoritma <i>Delay</i>	7
6 Data Test	8
6.1 Data Test 1	8
6.2 Data Test 2	8
6.3 Data Test 3	8
6.4 Data Test 4	9
6.5 Data Test 5	9
6.6 Data Test 6	9
6.7 Data Test 7	10
6.8 Data Test 8	10
6.9 Data Test 9	10
6.10 Data Test 10	11
6.11 Data Test 11	11
6.12 Data Test 12	11
6.13 Data Test 13	12
6.14 Data Test 14	12
6.15 Data Test 15	13
6.16 Data Test 16	13
6.17 Data Test 17	13
7 Test Script	14
8 Pembagian Kerja dalam Kelompok	17
9 Lampiran	18
9.1 Deskripsi Tugas Besar 2	18
9.2 Notulen Rapat	19
9.3 Log Activity Anggota Kelompok	21

1 Ringkasan

BNMO merupakan sebuah robot video *game console* yang rusak dan berhasil diperbaiki. Setelah diperbaiki, BNMO mendapat banyak bug sehingga dicari *programmer* yang dapat mengulang program robot *video console* tersebut. BNMO memiliki beberapa fitur di antaranya adalah :

1. Memainkan *game*
2. Menambahkan *game*
3. Menghapus *game*
4. Mengurutkan *game*

Berdasarkan deskripsi tersebut, kami membuat program *game* dengan beberapa komponen di antaranya adalah: *main menu* yang berisi *start* dan *load*, *create game*, *list game*, *delete game*, *queue game*, *play game*, *skip game*, *quit*, *help*, dan *command* lain, serta dua buah *game* yaitu RNG dan *Diner Dash*.

Laporan mencakup deskripsi umum dari persoalan *game*, penjelasan tambahan spesifikasi tugas, penjelasan tentang struktur data (ADT), program utama, algoritma menarik yang ditemukan selama mengerjakan tugas besar, penjelasan untuk menguji keberjalanan *game*, skenario *test* yang memungkinkan untuk semua fitur yang ada, pembagian tugas, dan lampiran tambahan seperti deskripsi tugas besar, notulensi rapat, *log activity*, dan lain-lain.

Pada tugas besar ini, kami membuat sebuah permainan berbasis CLI atau *command-line interface* dalam bahasa C dengan memanfaatkan ADT yang sudah dipelajari pada mata kuliah IF2210 – Algoritma dan Struktur Data. *Game* akan dimulai dengan menampilkan *main menu* yang berisi *start game* dan *load game*. Setelah pemain memilih, pemain dapat memilih beberapa *command* di antaranya *create game* untuk menambah *game*, *list game* untuk melihat daftar *game*, *delete game* untuk menghapus *game*, *queue game* untuk menambah *game* pada antrian yang akan digunakan ketika memainkan *game*, *play game* untuk memainkan *game* yang sudah didaftar di *queue game*, *skip game* untuk melewati *game* pada *queue game* sebanyak *n* kali, *quit* untuk keluar dari *game*, *help* untuk melihat daftar *command*, dan *command* lain jika terjadi kesalahan dalam penginputan *command*. Terdapat dua *game* yang tersedia. Yang pertama adalah RNG. *Game* ini akan menguji keberuntungan kita dengan menebak angka yang sudah di-*random* oleh *game*. *Game* akan mengeluarkan output “lebih kecil” atau “lebih besar” tergantung dari angka yang kita input. Permainan kedua adalah *Diner Dash*. *Game* ini mengantarkan makanan yang terurut berdasarkan prioritas dengan 3 buah *command*, yaitu : *cook*, *serve*, dan *skip*.

2 Penjelasan Tambahan Spesifikasi Tugas

2.1 Spesifikasi Fungsi Pertambahan

Fungsi ini digunakan untuk *game Math Quiz*. Fungsi akan meminta *user* untuk memilih tingkat kesulitan. Jika *user* memasukkan tingkat kesulitan “Mudah”, program akan mengeluarkan soal pertambahan dengan batas angka dari 0 sampai 100. Jika *user* memasukkan

tingkat kesulitan “Normal”, program akan mengeluarkan soal penambahan dengan batas angka dari 101 sampai 500. Dan jika *user* memasukkan tingkat kesulitan “Sulit”, program akan mengeluarkan soal penambahan dengan batas angka dari 501 sampai 1000. Lalu program akan melakukan *looping*. *User* diminta untuk menjawab soal penambahan dengan angka yang berdasarkan tingkat kesulitan yang sudah dipilih. Program akan mengeluarkan *integer random* dan mengecek kevalidan *input* menggunakan fungsi *isInteger*. Jika soal dijawab dengan benar, skor akan bertambah 10 dan program akan mengembalikan nilai skor.

2.2 Spesifikasi Fungsi Pengurangan

Fungsi ini digunakan untuk *game Math Quiz*. Sama seperti fungsi sebelumnya, fungsi akan meminta *user* untuk memilih tingkat kesulitan. Jika *user* memasukkan tingkat kesulitan “Mudah”, program akan mengeluarkan soal pengurangan dengan batas angka dari 0 sampai 100. Jika *user* memasukkan tingkat kesulitan “Normal”, program akan mengeluarkan soal pengurangan dengan batas angka dari 101 sampai 500. Dan jika *user* memasukkan tingkat kesulitan “Sulit”, program akan mengeluarkan soal pengurangan dengan batas angka dari 501 sampai 1000. Lalu program akan melakukan *looping*. *User* diminta untuk menjawab soal pengurangan dengan angka yang berdasarkan tingkat kesulitan yang sudah dipilih. Program akan mengeluarkan *random integer* dan mengecek kevalidan *input* menggunakan fungsi *isInteger*. Jika angka pertama lebih besar dari angka kedua, maka program akan mengeluarkan soal angka pertama dikurang angka kedua dan sebaliknya. Jika jawaban benar, skor akan bertambah 10 dan program akan mengembalikan nilai skor.

2.3 Spesifikasi Fungsi Perkalian

Fungsi ini digunakan untuk *game Math Quiz*. Sama seperti kedua fungsi sebelumnya, fungsi akan meminta *user* untuk memilih tingkat kesulitan. Jika *user* memasukkan tingkat kesulitan “Mudah”, program akan mengeluarkan soal perkalian dengan batas angka dari 0 sampai 100. Jika *user* memasukkan tingkat kesulitan “Normal”, program akan mengeluarkan soal perkalian dengan batas angka dari 101 sampai 500. Dan jika *user* memasukkan tingkat kesulitan “Sulit”, program akan mengeluarkan soal perkalian dengan batas angka dari 501 sampai 1000. Lalu program akan melakukan *looping*. *User* diminta untuk menjawab soal perkalian dengan angka yang berdasarkan tingkat kesulitan yang sudah dipilih. Program akan mengeluarkan *random integer* dan mengecek kevalidan *input* menggunakan fungsi *isInteger*. Jika jawaban benar, skor akan bertambah 10 dan program akan mengembalikan nilai skor.

1 Struktur Data (ADT)

1.1 ADT Array Dinamis

- Sketsa struktur data : struktur data terdiri atas *ElType pointer to char*, *Capacity* yang bertipe *integer*, dan Panjang dari elemen efektif (Neff) yang bertipe *integer*. *Prototype* yang terdapat pada ADT ini berupa konstruktor untuk membuat tabel kosong dengan mengalokasi

elemen *array*, destruktorkan untuk dealokasi *array*, fungsi untuk mengirim elemen efektif *array* dan mengecek jika *array* kosong, prosedur untuk memasukkan elemen pada awal, akhir, dan tempat yang telah ditentukan posisinya pada *array*.

- Persoalan yang diselesaikan : *command creategame, deletgame, listgame, load, playgame, queuegame, save, dan start*
- Alasan pemilihan : daftar *game* yang berubah-ubah selama *game* berlangsung akan disimpan dalam *array*
- Diimplementasikan dengan nama *file* header “arrayOfString.h”

1.2 ADT Mesin Karakter

- Sketsa struktur data : struktur data terdiri atas *variable extern* dengan tipe *char* dengan nama *currentChar*, *variable extern* dengan tipe *boolean* dengan nama *EOP* dan *extern file* pita. *MARK* didefinisikan dengan ‘.’ (titik) dan *ENTER* didefinisikan dengan ‘\n’ (*new line*). *Protoype* pada ADT ini berfungsi untuk membaca dari *file* dan dari terminal. Terdapat prosedur *START* untuk memulai pembacaan karakter dan *ADV* untuk menggeser pembacaan karakter. Juga terdapat fungsi *GetCC* untuk mengirim kembali karakter yang sudah dibaca, dan *IsEOP* untuk mengecek jika pembacaan karakter telah selesai. Dan untuk pembacaan dari terminal terdapat prosedur *COMMAND* untuk memulai pembacaan *command* dan prosedur *ADVC* untuk menggeser pembacaan *command*.
- Persoalan yang diselesaikan : ADT mesin kata
- Alasan pemilihan : ADT mesin kata yang membutuhkan fungsi-fungsi dari mesin karakter
- Diimplementasikan dengan nama *file* header “mesinkar2.h”

1.3 ADT Mesin Kata

- Sketsa struktur data : struktur terdiri atas *TabWord* dengan tipe *array of char* dan *Length* dengan tipe *integer*. *BLANK* didefinisikan dengan ‘ ’ (spasi) dan *ENTER* yang didefinisikan dengan \n atau *new line*. *Prototype* pada ADT ini terdiri dari prosedur untuk membaca *file* antara lain mengabaikan *BLANK*, mengabaikan *ENTER*, memulai pembacaan kata, menggeser pembacaan kata, menyalin kata, menulis kata ke layar, membuat *line* baru, dan menyalin *line*. Pada prosedur untuk membaca *command* dari terminal antara lain mengabaikan *BLANK* pada *command*, memulai *command*, menggeser pembacaan *command*, dan menyalin *command*. Terdapat prosedur untuk memanipulasi input *command*. Juga terdapat fungsi untuk mengkonversi tipe data antara lain mengakuisisi *char*, mengubah tipe *char* ke *integer* dan sebaliknya, mengubah tipe *word* ke *integer*, mengubah tipe *word* ke *string* dan sebaliknya, menyalin kata, mengembalikan Panjang *string*, menyalin *string*, dan mengeluarkan *random integer*. Yang terakhir terdapat fungsi untuk membandingkan kata antara lain mengecek jika *word* dan *char* adalah kata yang sama, mengecek jika dua buah *word* merupakan kata yang sama, membandingkan dua buah *string*, mengecek jika suatu kata merupakan tipe *integer*, mengembalikan Panjang kalimat, dan terdapat prosedur *delay* untuk *loading* dan menggabungkan dan memisahkan dua buah kata.

- Persoalan yang diselesaikan : membaca informasi dari *file* eksternal, membaca dan membandingkan *command* dari *user*, membaca *state* dari *game* yang sudah disimpan, mengecek tipe data sebuah input.
- Alasan pemilihan : pembacaan informasi dari *file* eksternal yang memerlukan akuisisi kata dan menerjemahkan dalam tipe lain., mengecek kevalidan sebuah input.
- Diimplementasikan dengan nama *file* header “mesinkata2.h”

1.4 ADT Queue

- Sketsa struktur data : struktur terdiri atas EType buffer (*pointer to char* dengan kapasitas maksimum 100 karakter), index HEAD yang bersifat *integer*, dan index TAIL yang bersifat *integer*. *Prototype* yang terdapat pada ADT ini terdiri dari konstruktor untuk membuat *queue* kosong, primitif untuk menambah (*enqueue*) dan menghapus (*dequeue*) elemen pada *queue*, penulisan isi *queue* secara traversal, pengecekan jika *queue* kosong atau *full*, pengiriman banyaknya elemen pada *queue*, dan pengecekan jika sebuah kata merupakan bagian dari *queue*
- Persoalan yang diselesaikan : *command queuegame, playgame, dan skipgame.*
- Alasan pemilihan : *command* ini akan menambah dan menghapus daftar game yang terdaftar pada list game yang akan dimainkan pada fitur play game dengan berurutan (*first in first out/FIFO*)
- Diimplementasikan dengan nama *file* header “queue.h”

2 Program Utama

Program ini dijalankan menggunakan *file* bernama main.c. Pada *file* ini, dilakukan include *file* header “console.h” yang meng-include seluruh ADT yang diperlukan dan mendeklarasi prosedur yang diperlukan pada main.c. Ketika program dijalankan, akan ditampilkan opening message. Lalu program mendeklarasikan beberapa variable yang dibutuhkan seperti Queue QueueGame dan ArrayDin ListGames. Lalu program akan meminta *user* untuk memilih *command* start atau load diikuti dengan nama *file*. Program akan melakukan *looping* untuk memasukkan *command* kedua. *Command* akan dicek menggunakan fungsi yang membandingkan antara *command* dengan input dari *user*. Jika *user* memasukkan *command* “START” maka prosedur STARTGAME dipanggil dan mengeluarkan prosedur HELP. Jika *user* memasukkan *command* LOAD, program akan menggeser pembacaan *command* lalu membaca nama *file* konfigurasi. Program akan mengambil data dari *file* yang telah dibaca dan mengeluarkan prosedur HELP. Jika bukan keduanya, program akan meminta *command* masukan kembali. program akan melakukan *looping* kembali.

Program mendeklarasi pointer to *char* bernama GAME. Lalu program akan meminta masukkan *command* dari *user*. Jika *command* masukan adalah SAVE, maka program akan menggeser pembacaan *command* dan membaca nama *file* konfigurasi. Program akan mengubah nama *file* masukan dengan fungsi wordToString dan menyimpan data dengan prosedur SAVEBNMO. Jika *command* masukan adalah CREATE, program akan menggeser pembacaan *command* dan membaca nama game yang dimasukkan. Program akan mengubah nama game dengan fungsi wordToString dan menggunakan prosedur CREATEGAME dan memasukkannya

pada array ListGames. Jika *user* memasukkan *command* LIST, program akan menggeser pembacaan *command* dan menggunakan prosedur LISTGAME dan memanggil array ListGames. Jika *user* memasukkan *command* DELETE, program akan menggeser pembacaan *command*. Program akan membandingkan *string* nama *game* yang dimasukkan dan kata “GAME” lalu program akan menghapus *game* dari array ListGames dan *queue* QueueGame. Jika *user* memasukkan *command* QUEUE, program akan mengeluarkan prosedur QUEUEGAME dengan memanggil *queue* QueueGame dan array ListGames. Jika *user* memasukkan *command* PLAY maka program akan mengeluarkan prosedur PLAYGAME. Jika *user* memasukkan *command* SKIPGAME, program akan menggeser pembacaan *command* dan mengecek jika *command* lanjutan merupakan *integer* dan bukan EndWord. Jika semua syarat terpenuhi, program akan mendefinisikan *command* lanjutan sebagai *integer* bernama skip dan memanggil prosedur SKIPGAME dengan jumlah skip yang sudah didefinisikan dari *command* lanjutan. Jika *user* memasukkan *command* HELP maka program akan mengeluarkan prosedur HELP. Jika *user* memasukkan *command* selain *command* yang tersedia, maka program akan mengeluarkan output “Command tidak dikenali. Silahkan masukkan command yang valid”. Program akan melakukan *looping* hingga *user* memasukkan *command* QUIT. Jika *user* mengeluarkan *command* QUIT maka program akan mengeluarkan prosedur QUIT yang menandakan program telah selesai dimainkan.

3 Algoritma-Algoritma Menarik

Algoritma ini dinilai menarik karena berbeda dari algoritma lainnya yang pernah dipelajari

3.1 Konversi Tipe Data

Algoritma ini digunakan untuk mengubah tipe data *word* ke *string* menggunakan fungsi wordToString dan sebaliknya menggunakan fungsi stringToWord. Fungsi ini mengirimkan kata yang elemen *array*-nya berasal dari *command*

```
Word stringToWord(char* command) ;
```

```
void wordToString(Word currentWord, char *string);
```

Terdapat juga algoritma untuk mengubah tipe data *word* ke *integer* menggunakan fungsi wordToInt dan sebaliknya menggunakan fungsi IntToWord.

```
int WordToInt(Word kata);
```

```
Word IntToWord(int X);
```

Dan terdapat algoritma untuk mengubah tipe *char* ke *integer* dan sebaliknya menggunakan fungsi charToInt dan intToChar

```
int charToInt(char c);
```

```
char intToChar(int c);
```

Seluruh algoritma di atas didefinisikan pada ADT mesin kata.

3.2 Algoritma Delay

Algoritma ini didefinisikan sebagai *delay* pada ADT mesin kata

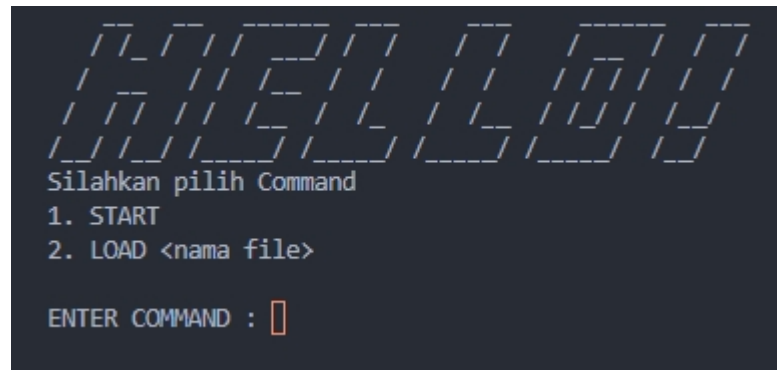
```
void delay(int number_of_seconds);
```

Algoritma ini digunakan pada *game* Math Quiz. Algoritma ini menarik karena algoritma ini akan memperlambat *output* yang selanjutnya dikeluarkan. Algoritma ini akan mengubah masukan *integer* menjadi *millisecond* lalu menyimpan waktu mulai dalam sebuah fungsi `clock()`. Lalu algoritma akan melakukan *looping* sampai waktu saat ini kurang dari waktu yang dibutuhkan tercapai. Algoritma ini digunakan setelah *output* “Loading”.

4 Data Test

4.1 Data Test 1

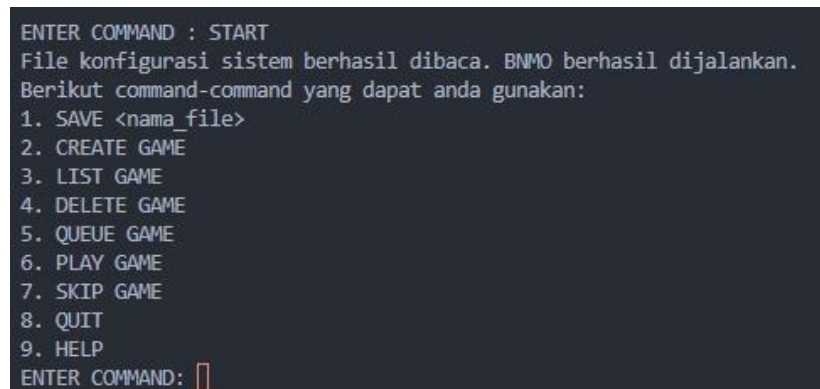
Test ini dilakukan untuk memeriksa jika program dapat berjalan dan menampilkan message serta *main menu*. Cara untuk melakukan kompilasi adalah dengan menggunakan perintah `gcc main.c ADT/mesinkata2.c ADT/mesinkar2.c ADT/arrayOfString.c ADT/queue.c console.c`



Gambar 1 : Main Menu

4.2 Data Test 2

Test ini dilakukan untuk memeriksa jika program dapat dijalankan.



Gambar 2 : Start

4.3 Data Test 3

Test ini dilakukan untuk memeriksa jika program dapat me-load file eksternal


```
ENTER COMMAND : LOAD savefile.txt
Save file berhasil dibaca. BNMO berhasil dijalankan.
Berikut command-command yang dapat anda gunakan:
1. SAVE <nama_file>
2. CREATE GAME
3. LIST GAME
4. DELETE GAME
5. QUEUE GAME
6. PLAY GAME
7. SKIP GAME
8. QUIT
9. HELP
ENTER COMMAND: 
```

Gambar 3 : Load

4.4 Data Test 4

Test ini dilakukan untuk memeriksa jika program dapat menyimpan permainan

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

Gambar 4 : Save

4.5 Data Test 5

Test ini dilakukan untuk menguji *command* CREATEGAME

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

Gambar 5 : Create Game

4.6 Data Test 6

Test ini dilakukan untuk menguji *command* LISTGAME

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

Gambar 6 : List Game

4.7 Data Test 7

Test ini dilakukan untuk menguji *command* DELETEDGAME

```
ENTER COMMAND: DELETE GAME
Masukkan nomor game yang akan dihapus: 6
Game berhasil dihapus
ENTER COMMAND: 
```

Gambar 7 : Delete Game

4.8 Data Test 8

Test ini dilakukan untuk menguji *command* QUEUEGAME

```
ENTER COMMAND: QUEUE GAME
Antrian game kosong.

Berikut adalah daftar game yang tersedia
1. RNG
2. Diner DASH
3. DINOSAUR IN EARTH
4. RISEWOMAN
5. EIFFEL TOWER
Nomor Game yang mau ditambahkan ke antrian : 1
Game berhasil ditambahkan kedalam daftar antrian.
ENTER COMMAND: 
```

Gambar 8 : Queue Game

4.9 Data Test 9

Test ini dilakukan untuk menguji *command* PLAYGAME

```

ENTER COMMAND: PLAY GAME
Loading RNG ...
RNG Telah dimulai. Uji keberuntungan Anda dengan menebak sebuah angka X yang rentangnya berada diantara 1 - 50.
Tebakan : 12
Lebih Besar
Tebakan : 20
Lebih Besar
Tebakan : 40
Lebih Kecil
Tebakan : 30
Lebih Kecil
Tebakan : 35
Lebih Kecil
Tebakan : 25
Lebih Besar
Tebakan : 23
Lebih Besar
Tebakan : 22
Lebih Besar
Tebakan : 21
Lebih Besar
Tebakan : 18
Lebih Besar
Tebakan : 28
Lebih Besar
Tebakan : 30
Lebih Kecil
Tebakan : 29
YA , X Adalah 29
skor = 38
ENTER COMMAND: 

```

Gambar 9 : Play Game

4.10 Data Test 10

Test ini dilakukan untuk menguji *command* SKIPGAME

```

ENTER COMMAND: SKIPGAME 2
Berikut adalah daftar antrian game-mu.
1. RNG
2. Diner DASH
3. DINOSAUR IN EARTH
4. RISEWOMAN
Game DINOSAUR IN EARTH masih dalam maintenance, belum dapat dimainkan. Silahkan pilih game lain.
ENTER COMMAND: 

```

Gambar 10 : Skip Game

4.11 Data Test 11

Test ini dilakukan untuk menguji *command* QUIT

```

ENTER COMMAND: QUIT
Anda keluar dari game BNMO.
Bye bye ...

```

Gambar 11 : Quit

4.12 Data Test 12

Test ini dilakukan untuk menguji *command* HELP

STEI- ITB	IF2111-TBI-09	Halaman 11 dari 23 halaman
Template dokumen ini dan informasi yang dimilikinya adalah milik Sekolah Teknik Elektro dan Informatika ITB dan bersifat rahasia. Dilarang me-reproduksi dokumen ini tanpa diketahui oleh Sekolah Teknik Elektro dan Informatika ITB.		

```

ENTER COMMAND: HELP
Berikut command-command yang dapat anda gunakan:
1. SAVE <nama_file>
2. CREATE GAME
3. LIST GAME
4. DELETE GAME
5. QUEUE GAME
6. PLAY GAME
7. SKIP GAME
8. QUIT
9. HELP
ENTER COMMAND: 

```

Gambar 12 : Help

4.13 Data Test 13

Test ini dilakukan untuk menguji kevalidan *command*

```

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

```

Gambar 13 : Command Lain

4.14 Data Test 14

Test ini dilakukan untuk menguji game RNG

```

ENTER COMMAND: PLAY GAME
Loading RNG ...
RNG Telah dimulai. Uji keberuntungan Anda dengan menebak sebuah angka X yang rentangnya berada diantara 1 - 50.
Tebakan : 12
Lebih Besar
Tebakan : 20
Lebih Besar
Tebakan : 40
Lebih Kecil
Tebakan : 30
Lebih Kecil
Tebakan : 35
Lebih Kecil
Tebakan : 25
Lebih Besar
Tebakan : 23
Lebih Besar
Tebakan : 22
Lebih Besar
Tebakan : 21
Lebih Besar
Tebakan : 18
Lebih Besar
Tebakan : 28
Lebih Besar
Tebakan : 30
Lebih Kecil
Tebakan : 29
YA , X Adalah 29
skor = 38
ENTER COMMAND: 

```

Gambar 14 : Game RNG

4.15 Data Test 15

Test ini dilakukan untuk menguji *game* Diner Dash.

```
ENTER COMMAND: PLAY GAME
Loading RNG ...
RNG Telah dimulai. Uji keberuntungan Anda dengan menebak sebuah angka X yang rentangnya berada diantara 1 - 50.
Tebakan : 12
Lebih Besar
Tebakan : 20
Lebih Besar
Tebakan : 40
Lebih Kecil
Tebakan : 30
Lebih Kecil
Tebakan : 35
Lebih Kecil
Tebakan : 25
Lebih Besar
Tebakan : 23
Lebih Besar
Tebakan : 22
Lebih Besar
Tebakan : 21
Lebih Besar
Tebakan : 18
Lebih Besar
Tebakan : 28
Lebih Besar
Tebakan : 30
Lebih Kecil
Tebakan : 29
YA , X Adalah 29
skor = 38
ENTER COMMAND: 
```

Gambar 15 : Game Diner Dash

4.16 Data Test 16

Test ini dilakukan untuk menguji *game* tambahan

```
ENTER COMMAND: PLAY GAME
16571
ENTER COMMAND: 
```

Gambar 16 : Game Tambahan

Game akan mengeluarkan *integer random* sebagai skor akhir

4.17 Data Test 17

Test ini dilakukan untuk menguji *game* Math Quiz

```

ENTER COMMAND: SKIPGAME 2
Berikut adalah daftar antrian game-mu.
1. DINOSAUR IN EARTH
2. RISEWOMAN
3. MATH QUIZ
Loading MATH QUIZ ...
<SELAMAT DATANG DI GAME MATHQUIZ>
GAME RULE :
1.Pilih Mode Quiz Matematika
2.Pilih Tingkat Kesulitan Mode , Semakin Tinggi Tingkat Kesulitannya, Semakin Besar Angka Yang Akan Dihitung
3.Jawaban Kuis Tidak Perlu Dispasi
4.Setiap Mode terdiri dari 10 soal
5.Skor Maksimal Dari Setiap Permainan Adalah 100

SELAMAT BERMAIN!
Pilih Mode Quiz Matematika !
1.Pertambahan
2.Pengurangan
3.Perkalian
ENTER COMMAND : pert
Masukan Command Anda Salah , Silahkan Ulangi
ENTER COMMAND : Pertambahan
Pilih Tingkat Kesulitan !
1.Mudah
2.Normal
3.Sulit
ENTER COMMAND : Mudah
Loading Mode pertambahan...
83 + 83 = 166
Jawabannya adalah 166
Skor kuis kamu sekarang adalah 10

83 + 11 = 

```

Gambar 17 : Math Quiz

5 Test Script

Table 1. 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 main menu dapat dijalankan	Melakukan compile <i>file</i> main.c di terminal	Data Test 1	Program menampilkan message "Hello!" dan pilihan <i>command</i>	Sesuai yang diharapkan
2.	Fitur Start Game	Memeriksa apakah permainan dapat dijalankan	Memasukkan input STARTGAME	Data Test 2	Permainan berhasil dimulai dan program menampilkan pilihan <i>command</i>	Sesuai yang diharapkan

3.	Fitur Load Game	Memeriksa apakah permainan dapat me-load <i>file</i> eksternal	Memasukkan input LOADGAME diikuti nama <i>file</i> yang akan di-load	Data Test 3	Permainan berhasil melakukan load dari <i>file</i> eksternal	Sesuai yang diharapkan
4.	Fitur Save Game	Memeriksa apakah permainan dapat disimpan	Memasukkan input SAVEGAME	Data Test 4	Permainan berhasil disimpan	Sesuai yang diharapkan
5.	Fitur Create Game	Memeriksa apakah permainan dapat menambah game baru	Memasukkan <i>command</i> CREATEGAME	Data Test 5	Permainan berhasil menambah permainan baru	Sesuai yang diharapkan
6.	Fitur List Game	Memeriksa apakah permainan dapat mengeluarkan daftar permainan	Memasukkan <i>command</i> LISTGAME	Data Test 6	Permainan berhasil mengeluarkan daftar permainan	Sesuai yang diharapkan
7.	Fitur Delete Game	Memeriksa apakah permainan dapat dihapus dari daftar permainan	Memasukkan <i>command</i> DELETGAME	Data Test 7	Permainan berhasil menghapus game dari daftar permainan	Sesuai yang diharapkan
8.	Fitur Queue Game	Memeriksa apakah permainan dapat menambah antrian permainan	Memasukkan <i>command</i> QUEUEGAME	Data Test 8	Permainan berhasil menambah game di antrian permainan	Sesuai yang diharapkan

9.	Fitur Play Game	Memeriksa apakah permainan dapat dimainkan	Memasukkan <i>command</i> PLAYGAME	Data Test 9	Permainan yang terdaftar di antrian dapat dimainkan	Sesuai yang diharapkan
10.	Fitur Skip Game	Memeriksa apakah permainan dapat melewati permainan dari antrian	Memasukkan <i>command</i> SKIPGAME diikuti dengan <i>integer</i> sebagai jumlah permainan yang akan dilewati	Data Test 10	Permainan berhasil melewati game dari antrian	Sesuai yang diharapkan
11.	Fitur Quit	Memeriksa apakah permainan dapat menjalankan <i>command</i> quit	Memasukkan <i>command</i> QUIT	Data Test 11	Program dapat keluar dari permainan	Sesuai yang diharapkan
12.	Fitur Help	Memeriksa apakah permainan dapat mengeluarkan daftar <i>command</i> yang ada	Memasukkan <i>command</i> HELP	Data Test 12	Permainan berhasil mengeluarkan daftar <i>command</i>	Sesuai yang diharapkan
13.	Fitur <i>Command</i> Lain	Memeriksa apakah permainan dapat mengecek kevalidan <i>command</i>	Memasukkan input <i>command</i> sembarang	Data Test 13	Permainan berhasil mengecek kevalidan <i>command</i>	Sesuai yang diharapkan
14.	Fitur RNG	Memeriksa apakah permainan RNG dapat dijalankan	Memasukkan <i>game</i> RNG dalam <i>queue</i> dan memainkannya menggunakan PLAYGAME	Data Test 14	Permainan RNG dapat dijalankan	Sesuai yang diharapkan

15.	Fitur Diner Dash	Memeriksa apakah permainan Diner Dash dapat dijalankan	Memasukkan <i>game</i> Diner Dash dalam <i>queue</i> dan memainkannya menggunakan PLAYGAME	Data Test 15	Permainan Diner Dash dapat dijalankan	
16.	Fitur Game tambahan	Memeriksa apakah permainan <i>game</i> tambahan dapat dijalankan	Memasukkan <i>command</i> nama <i>game</i> yang didaftar di CREATEGAME	Data Test 16	Permainan tambahan dapat dijalankan dan langsung ke tahap game over dengan skor yang berupa <i>integer random</i>	Sesuai yang diharapkan
17.	Fitur Game Math Quiz	Memeriksa apakah permainan Math Quiz dapat dijalankan	Memasukkan <i>game</i> Math Quiz dalam <i>queue</i> dan memainkannya menggunakan PLAYGAME	Data Test 17	Permainan Math Quiz dapat dijalankan	Sesuai yang diharapkan

6 Pembagian Kerja dalam Kelompok

Table 2. Pembagian Tugas

NIM	Nama Lengkap	Deskripsi Tugas
18221044	Kinanti Wening Asih	<ul style="list-style-type: none"> • Membuat prosedur SKIPGAME • Membuat game Diner DASH

18221058	Marvel Subekti	<ul style="list-style-type: none"> • Membuat MoM • Membuat LOAD • Membuat game RNG • Membuat fungsi SAVE • Membuat game MATHQUIZ • Menyatukan fungsi/prosedur dalam file “main.c”
18221094	Raka Admiharfan Fatihah	<ul style="list-style-type: none"> • Membuat fungsi DELETE GAME • Membuat fungsi PLAY GAME • Membuat fungsi HELP
18221120	Carissa Tabina Rianda	<ul style="list-style-type: none"> • Membuat laporan • Membuat fungsi QUEUE GAME • Membuat fungsi QUIT
18221170	Amjad Adhie Prasetyo	<ul style="list-style-type: none"> • Membuat fungsi START • Membuat fungsi CREATE GAME • membuat fungsi LIST GAME • Menyatukan fungsi/prosedur dalam file “main.c”

7 Lampiran

7.1 Deskripsi Tugas Besar 2

BNMO (dibaca: Binomo) adalah sebuah robot *video game console* yang dimiliki oleh Indra dan Doni. Dua bulan yang lalu, ia mengalami kerusakan dan telah berhasil diperbaiki. Sayangnya, setelah diperbaiki ia justru mendapatkan lebih banyak *bug* dalam sistemnya. Oleh karena itu, Indra dan Doni mencari *programmer* lain yang lebih andal untuk ulang memprogram robot *video game console* kesayangannya. 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**

7.2 Notulen Rapat

Notulensi Pertemuan 1 Tugas Besar Alstrukdat 29 Oktober 2022

Membahas spesifikasi tugas dan membagi tugas untuk masing-masing anggota kelompok

Amjad : *start, creategame, listgame*

Marvel : *load, RNG*

Carissa : *save, queuegame, quit*

Raka : *deletgame, playgame*

Asih : *command lain, skipgame, Diner Dash*

Note :

- Asumsi history permainan yang terakhir dimainkan (eiffel tower kurang jelas)
- Ketahanan di Diner Dash kurang jelas



Asistensi 1 Jumat 4 November 2022





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

No. Kelompok/Kelas : 9 / K02
Nama Kelompok :
Anggota Kelompok (Nama/NIM) :
1. Kinanti Wening Asih/18221044
2. Marvel Subekti/18221058
3. Amjad Adhie Prasetyo/18221170
4. Raka Admiharfan Fatihah/18221094
5. Carissa Tabina Rianda/18221120

Asisten Pembimbing : Afif Fahreza

Asistensi I

Tanggal : Jumat, 4 November 2022	Catatan Asistensi: 1. <i>File</i> konfigurasi boleh ditambahin MARK 2. MARK enter itu “\n” 3. Contoh kode unik adalah <code>adt</code> untuk <code>compare string / word</code>
Tempat : Zoom	
Kehadiran Anggota Kelompok: <div style="text-align: center;">No NIM Tanda tangan</div> <div style="text-align: center;">1 18221058 </div> <div style="text-align: center;">2 18221120 </div> <div style="text-align: center;">3</div>	

<p>18221170</p>  <p>4 18221094</p>  <p>5 18221044</p> 	
	<p>Tanda Tangan Asisten:</p> 

Asistensi II

Tanggal : Kamis, 10 November 2022	Catatan Asistensi:
Tempat : Zoom	

1. Jangan lupa rapihin output

Kehadiran Anggota Kelompok:

No
NIM
Tanda tangan

1
18221058



2
18221120



3
18221170




4
18221094



5
18221044



2. Tidak test case untuk setiap ADT yang dibuat.

	Tanda Tangan Asisten: 

7.3 Log Activity Anggota Kelompok

Waktu	NIM	Keterangan
29 Oktober 2022	18221044, 18221058, 18221170, 18221094, 18221120	Rapat perdana pembagian tugas kelompok
4 November 2022	18221044, 18221058, 18221170, 18221094, 18221120	Asistensi 1
8 November 2022	18221044, 18221058, 18221170, 18221094, 18221120	Menyelesaikan dan testing seluruh fungsi/prosedur
9 November 2022	18221044, 18221058, 18221170, 18221094, 18221120	Menyatukan seluruh fungsi dalam satu program bernama main.c
10 November 2022	18221044, 18221058, 18221170, 18221094, 18221120	Asistensi 2

