

# **LAPORAN TUGAS BESAR**

## **IF2111 Algoritma dan Struktur Data STI**

### **PURRMART**

Dipersiapkan oleh:

Kelompok 11

Mochamad Ikhbar A / 18223050

Derick Amadeus Budiono / 18223090


Indana Aulia Ayundazulfa / 18223100

Wilson / 18223012

Naila Selvira Budiana / 18223018

Sekolah Teknik Elektro dan Informatika - Institut Teknologi Bandung

Jl. Ganesha 10, Bandung 40132

	<b>Sekolah Teknik Elektro dan Informatika ITB</b>	<b>Nomor Dokumen</b>		<b>Halaman</b>
		<i>IF2111-TB-K02-11</i>		<i>&lt;jml hlm&gt;</i>
		<i>Revisi</i>	<i>&lt;no revisi&gt;</i>	<i>25-11-2024</i>

# Daftar Isi

1	Ringkasan	3
2	Penjelasan Tambahan Spesifikasi Tugas	4
2.1	Quantum Wordl	4
3	Struktur Data (ADT)	4
3.1	Mesin Karakter	4
3.1.1	Sketsa Struktur Data	4
3.1.2	Persoalan yang Diselesaikan	5
3.1.3	Alasan Pemilihan	5
3.1.4	Implementasi	6
3.2	Mesin Kata	6
3.2.1	Sketsa Struktur Data	6
3.2.2	Persoalan yang Diselesaikan	8
3.3	List Dinamis	9
3.3.1	Sketsa Struktur Data	9
3.3.2	Persoalan yang Diselesaikan	11
3.3.3	Alasan Pemilihan	11
3.4	Queue	12
3.4.1	Sketsa Struktur Data	12
3.4.2	Persoalan yang Diselesaikan	14
3.4.3	Alasan Pemilihan	14
3.5.	Array	14
3.5.1.	Sketsa Struktur Data	14
3.5.2.	Persoalan yang Diselesaikan	17
3.5.3.	Alasan Pemilihan	17
4	Program Utama	18
5	Algoritma-Algoritma Menarik	18
5.1	Algoritma Random	18
6	Data Test	19
6.1	Data Test START	19
6.2	Data Test LOAD	19
6.3	Data Test LOAD	20
6.4	Data Test REGISTER	20
6.5	Data Test LOGIN	20
6.6	Data Test STORE LIST	21

6.7 Data Test STORE REQUEST	21
6.8 Data Test STORE SUPPLY	22
6.9 Data Test STORE REMOVE	23
6.10 Data Test WORK	23
6.11 Data Test WORK CHALLENGE	24
7 Test Script	24
8 Pembagian Kerja dalam Kelompok	26
9 Lampiran	27
9.1 Deskripsi Tugas Besar	28
9.2 Notulen Rapat	30
9.3 Log Activity Anggota Kelompok	30

# 1 *Ringkasan*

Kesulitan yang dialami Agen Purry ketika harus menyuplai segala kebutuhan yang sumbernya sangat sulit dijangkau, borma bojongsoang, membuat agen purry serta OWCA kewalahan. Maka dari itu, tim OWCA menghubungi kelompok 11 untuk dibuatkan sebuah sistem jual beli dengan nama PURRMART.

PURRMART adalah sebuah aplikasi berbasis CLI ( Command-Line Interface ) yang dibuat dengan bahasa C dengan bantuan struktur data terkait list ( statis dan dinamis ), mesin karakter, mesin kata, dan queue. Tak hanya itu, user yang masuk tentunya terautentikasi dan dapat melakukan minigames lainnya selain melihat toko, seperti work dan work challenge. Yang spesial dari tugas besar ini adalah tantangan untuk murni menggunakan mesin kata serta mesin karakter dan tidak diperbolehkannya implementasi scanf dan fgets.

Laporan ini berisikan mengenai penjelasan lebih detail mengenai fitur fitur atau ADT yang kami gunakan saat pengerjaan program PURRMART ini, tes data, dan script yang dilakukan, pembagian kerja, serta lampiran yang terkait.

## 2 *Penjelasan Tambahan Spesifikasi Tugas*

### 2.1 *Quantum Wordl3*

Quantum Wordl3 adalah fitur tambahan berupa *work challenge* ekstra bagi pemain. Berbeda dengan WORDL3 yang terdapat pada spesifikasi wajib, fitur Quantum WORDL3 memungkinkan pemain untuk menebak empat kata sekaligus dalam satu waktu. Seperti halnya WORDL3, Quantum WORDL3 mengusung konsep Random Number Generator. Perbedaan spesifik dengan WORDL3 adalah cara permainan itu berlangsung. User bisa melakukannya dengan hanya 9 kesempatan.

## 3 *Struktur Data (ADT)*

### 3.1 *Mesin Karakter*

#### 3.1.1 *Sketsa Struktur Data*

```
#ifndef MESINKARAKTER_H
#define MESINKARAKTER_H

#include <stdio.h>
#include "../boolean.h"
```

```

#define MARK '.'           // Define the MARK character for end
of processing
#define MAX_LENGTH 1000 // Maximum length of the input string
extern char currentChar;
extern boolean EOP;

void START();
// I.S. : sembarang
// F.S. : CC adalah karakter pertama pita
void END();
// I.S. : sembarang
// F.S. : CC adalah karakter terakhir yang terbaca dari pita
void ADV();
// I.S. : CC != MARK
// F.S. : CC adalah karakter berikutnya dari CC yang lama,
mungkin MARK
char GetCC();
// Mengembalikan karakter saat ini (currentChar)

boolean IsEOP();
// Mengembalikan true jika EOP tercapai
int panjangKarakter(char s[]);
// Menghitung panjang string hingga EOP atau batas karakter
tercapai

#endif

```

### 3.1.2 Persoalan yang Diselesaikan

Mesin Karakter digunakan sebagai pengganti input yang tidak diperbolehkan, sehingga mesin karakter berguna untuk menerima input dari user. Selain itu, mesin karakter memiliki beberapa fungsi tambahan untuk melihat panjang karakter yang digunakan secara khusus untuk word13 sebagai *constraint* input dari user.

### 3.1.3 Alasan Pemilihan

Dalam implementasinya, Mesin Karakter berperan sangat penting dalam menerima user input, dan juga primitif-primitif yang tersedia sangat berguna untuk digunakan sebagai pengecekan/iterasi lebih lanjut mengenai input yang dimasukkan.

### 3.1.4 Implementasi

Implementasi dari mesin karakter terdapat pada hampir keseluruhan program yang dibuat. Terdapat fungsi yang ditambahkan yaitu panjang Karakter yang diimplementasikan pada wordl3 untuk memerikan batasan kepada user bahwa kata yang dimasukan harus memiliki 5 karakter.

## 3.2 *Mesin Kata*

### 3.2.1 Sketsa Struktur Data

```
/* File: mesinkata.h */
/* Definisi Mesin Kata: Model Akuisisi Versi I */

#ifndef __MESINKATA_H__
#define __MESINKATA_H__

#include "../boolean.h"
#include "../mesinkarakter/mesinkarakter.h"

#define NMax 50
#define BLANK ' '

typedef struct
{
    char TabWord[NMax]; /* container penyimpan kata, indeks
yang dipakai [0..NMax-1] */
    int Length;
} Word;

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

void IgnoreBlanks();
/* Mengabaikan satu atau beberapa BLANK
I.S. : currentChar sembarang
```

```

        F.S. : currentChar ≠ BLANK atau currentChar = MARK */

void STARTWORD();
/* I.S. : currentChar sembarang
   F.S. : EndWord = true, dan currentChar = MARK;
          atau EndWord = false, currentWord adalah kata yang
          sudah diakuisisi,
          currentChar karakter pertama sesudah karakter
          terakhir kata */

void ADVWORD();
/* I.S. : currentChar adalah karakter pertama kata yang akan
   diakuisisi
   F.S. : currentWord adalah kata terakhir yang sudah
   diakuisisi,
          currentChar adalah karakter pertama dari kata
          berikutnya, mungkin MARK
          Jika currentChar = MARK, EndWord = true.
   Proses : Akuisisi kata menggunakan procedure SalinWord */

void CopyWord();
/* Mengakuisisi kata, menyimpan dalam currentWord
   I.S. : currentChar adalah karakter pertama dari kata
   F.S. : currentWord berisi kata yang sudah diakuisisi;
          currentChar = BLANK atau currentChar = MARK;
          currentChar adalah karakter sesudah karakter
          terakhir yang diakuisisi.
          Jika panjang kata melebihi NMax, maka sisa kata
          "dipotong" */

boolean isEndWord();
/* Mengembalikan true jika EndWord = true */

void printWord(Word Kata);
/* I.S. : Kata terdefinisi
   F.S. : Kata tercetak di layar tanpa karakter tambahan di
          awal maupun di akhir */

```

```

boolean StringCompare(Word kata1, Word kata2);
/* Mengembalikan true jika kata1 sama dengan kata2 */

Word str2Word(char* String);
/* Mengubah string menjadi Word */

char* Word2str(Word Kata);
/* Mengubah Word menjadi string */

int Word2int(Word Kata);
/* Mengubah Word menjadi integer */

Word int2Word(int Angka);
/* Mengubah integer menjadi Word */

Word CloneWord(Word kata);
/* Mengembalikan salinan dari kata */

char isOnWord(Word kata, char c);
/* Mengembalikan true jika c terdapat pada kata */

void STARTINPUT ();
/* fungsi dapat memulai sebuah input */

void IgnoreRest();
/* membuat batasan akuisisi kata berdasarkan batasan tertentu
*/

#endif

```

### 3.2.2 Persoalan yang Diselesaikan

Mesin Kata merupakan pengembangan dari mesin karakter. Tujuan digunakannya mesin kasta adalah untuk menerima input user yang berupa sebuah kalimat. Berbeda dengan



mesin karakter, pada ADT mesinkata, hal yang menjadi elemen utamanya adalah sebuah kata yang terdiri dari beberapa karakter.

### 3.3 *List Dinamis*

#### 3.3.1 Sketsa Struktur Data

```
/* File : list_dinamis.h */
/* Definisi ADT list_dinamis dengan representasi array secara
eksplisit dan alokasi dinamis */

#ifndef LIST_DINAMIS_H
#define LIST_DINAMIS_H

#include "../mesinkata/mesinkata.h"
#include "../barang/barang.h"

#define InitialSize 10
#define MAX_LEN 20
typedef int IdxType;
typedef CurrentBarang ArrayElType;

typedef struct {
    ArrayElType *A;
    int Capacity;
    int Neff;
} ArrayDin;

#define A(AD) (AD).A
#define Cap(AD) (AD).Capacity
#define Neff(AD) (AD).Neff

ArrayDin MakeArrayDin();
/* Membuat sebuah array dinamis kosong dengan kapasitas awal
```

```

InitialSize.
I.S.: Tidak ada.
F.S.: Array dinamis kosong terbuat, siap digunakan. */

void DeallocateArrayDin(ArrayDin *array);
/*Menghapus alokasi memori yang digunakan oleh array dinamis.
I.S.: Array dinamis valid.
F.S.: Memori untuk array dinamis telah dibebaskan.
*/

boolean IsEmptyList(ArrayDin array);
/*Mengecek apakah array dinamis kosong (Neff = 0).
return: true jika kosong, false jika tidak.*/

int Lengtharray(ArrayDin array);
/*Mengembalikan jumlah elemen efektif dalam array. */

int GetCapacity(ArrayDin array);
/*Mengembalikan kapasitas maksimum array. */

void InsertAt(ArrayDin *array, ArrayElType el, IdxType i);
/* Menyisipkan elemen el pada indeks i.
I.S.: Array valid,  $0 \leq i \leq \text{Neff}$ , array belum penuh.
F.S.: Elemen baru dimasukkan pada indeks i, elemen lainnya bergeser. */

void InsertLast(ArrayDin *array, ArrayElType el);
/*Menambahkan elemen el di akhir array.
I.S.: Array valid dan tidak penuh.
F.S.: Elemen baru ditambahkan sebagai elemen terakhir. */

void InsertFirst(ArrayDin *array, ArrayElType el);
/*Menambahkan elemen el di awal array.
I.S.: Array valid dan tidak penuh.
F.S.: Elemen lainnya bergeser, elemen baru ditambahkan di indeks 0.*/

```

```

void DeleteAt(ArrayDin *array, IdxType i);
/*Menghapus elemen pada indeks i.
I.S.: Array tidak kosong,  $0 \leq i < \text{Neff}$ .
F.S.: Elemen di indeks i dihapus, elemen lainnya bergeser. */

void DeleteLast(ArrayDin *array);
/* Menghapus elemen terakhir dari array.
I.S.: Array tidak kosong.
F.S.: Elemen terakhir dihapus.
*/

void DeleteFirst(ArrayDin *array);
/*Menghapus elemen pertama dari array.
I.S.: Array tidak kosong.
F.S.: Elemen pertama dihapus, elemen lainnya bergeser.
*/

void PrintArrayDin(ArrayDin array);
/* Menampilkan elemen-elemen array ke layar.
Format: Elemen ditampilkan dalam urutan dari indeks pertama hingga
terakhir.
*/

IdxType SearchArrayDin(ArrayDin array, ArrayElType item);
/* Mencari elemen item dalam array.
Return: Indeks elemen jika ditemukan, atau nilai tertentu (misal
-1) jika tidak ditemukan.*/

#endif

```

### 3.3.2 Persoalan yang Diselesaikan

List dinamis digunakan pada fitur store. List dinamis berfungsi sebagai media penyimpanan barang ( CurrentBarang ) yang akan senantiasa bertambah atau berkurang sesuai keinginan user yang masuk.

### 3.3.3 Alasan Pemilihan

Karena elemen dalam sebuah list bersifat dinamis ( bertambah dan berkurang tanpa tahu berapa frekuensi spesifik perubahannya ), maka list dinamis akan lebih unggul dibanding list statik yang sedari awal sudah ditentukan jumlah elemen maksimalnya berapa.

## 3.4 Queue

### 3.4.1 Sketsa Struktur Data

Queue yang digunakan pada program PURRMART adalah queue biasa ( bukan circular queue ) yang berbasis array. Tipe data yang digunakan pada ADT queue ini adalah tipe Word dengan CurrentBarang sebagai elemen dalam queue ( berisikan nama barang dan harga barang ).

```
/* File : queue.h */
/* Definisi ADT Queue dengan representasi array secara eksplisit
dan alokasi statik */

#ifndef QUEUE_H
#define QUEUE_H

#include "../boolean.h"
#include "../mesinkata/mesinkata.h"
#include "../barang/barang.h"

#define IDX_UNDEF -1
#define CAPACITY 100

/* Definisi elemen dan address */
typedef CurrentBarang BarangElType;
typedef struct {
    BarangElType buffer[CAPACITY];
    int idxHead;
    int idxTail;
} Queue;

/* ***** AKSES (Selektor) ***** */
```

```

/* Jika q adalah Queue, maka akses elemen : */
#define IDX_HEAD(q) (q).idxHead
#define IDX_TAIL(q) (q).idxTail
#define HEAD(q) (q).buffer[(q).idxHead]
#define TAIL(q) (q).buffer[(q).idxTail]

/* *** Kreator *** */
void CreateQueue(Queue *q);
/* I.S. sembarang */
/* F.S. Sebuah q kosong terbentuk dengan kondisi sbb: */
/* - Index head bernilai IDX_UNDEF */
/* - Index tail bernilai IDX_UNDEF */
/* Proses : Melakukan alokasi, membuat sebuah q kosong */

/* ***** Prototype ***** */
boolean queueisEmpty(Queue q);
/* Mengirim true jika q kosong: lihat definisi di atas */
boolean queueisFull(Queue q);
/* Mengirim true jika tabel penampung elemen q sudah penuh */
/* yaitu IDX_TAIL akan selalu di belakang IDX_HEAD dalam buffer
melingkar*/

int queuelength(Queue q);
/* Mengirimkan banyaknya elemen queue. Mengirimkan 0 jika q
kosong. */

/* *** Primitif Add/Delete *** */
/* PRAPRAKTIKUM
void enqueue(Queue *q, BarangElType val); */
/* Proses: Menambahkan val pada q dengan aturan FIFO */
/* I.S. q mungkin kosong, tabel penampung elemen q TIDAK penuh */
/* F.S. val menjadi TAIL yang baru, IDX_TAIL "mundur" dalam buffer
melingkar. */

void enqueue(Queue *q, BarangElType val);
/* Proses: Menambahkan val pada q dengan aturan FIFO */
/* I.S. q mungkin kosong, tabel penampung elemen q TIDAK penuh */

```

```

/* F.S. val menjadi TAIL yang baru, IDX_TAIL "mundur".
    Jika q penuh semu, maka perlu dilakukan aksi penggeseran
    "maju" elemen-elemen q
    menjadi rata kiri untuk membuat ruang kosong bagi TAIL
    baru */

void dequeue(Queue *q, BarangElType *val);
/* Proses: Menghapus val pada q dengan aturan FIFO */
/* I.S. q tidak mungkin kosong */
/* F.S. val = nilai elemen HEAD pd I.S., IDX_HEAD "mundur";
    q mungkin kosong */

/* *** Display Queue *** */
void displayQueue(Queue q);
/* Proses : Menuliskan isi Queue dengan traversal, Queue ditulis
    di antara kurung
    siku; antara dua elemen dipisahkan dengan separator "koma",
    tanpa tambahan
    karakter di depan, di tengah, atau di belakang, termasuk spasi
    dan enter */
/* I.S. q boleh kosong */
/* F.S. Jika q tidak kosong: [e1,e2,...,en] */
/* Contoh : jika ada tiga elemen bernilai 1, 20, 30 akan dicetak:
    [1,20,30] */
/* Jika Queue kosong : menulis [] */

#endif

```

### 3.4.2 Persoalan yang Diselesaikan

ADT Queue sangat terpakai di salah satu function store, yaitu store\_request dimana function tersebut memungkinkan user memasukkan barang ke sebuah antrean. Antrean ini adalah tahap pemrosesan sebelum barang dimasukkan ke store list.

### 3.4.3 Alasan Pemilihan

Alasan Queue digunakan dibanding ADT list biasanya karena sejatinya fitur store\_request adalah simulasi antrean barang yang juga mengharuskan queue digunakan dalam implementasinya.

## 3.5. Array

### 3.5.1. Sketsa Struktur Data

Array yang digunakan pada fitur register adalah array statik.

```
#ifndef ARRAY_H
#define ARRAY_H

#include "boolean.h"
#include "mesinkata.h"
#include <stdio.h>
/* Kamus Umum */

#define IdxMax 100
#define IdxMin 1
#define IdxUndef -999 /* indeks tak terdefinisi*/

/* Definisi elemen dan koleksi objek */
typedef int IdxType;
typedef Word ElType;

typedef struct
{
    ElType TI [IdxMax-IdxMin+1]; /* memori tempat penyimpan elemen
(container) */
    int Neff; /* banyaknya elemen efektif */
} TabKata;

/* Indeks yang digunakan [IdxMin..IdxMax] */
/* Jika T adalah TabKata, cara deklarasi dan akses: */
/* Deklarasi : T : TabKata */
/* Maka cara akses:
* T.Neff untuk mengetahui banyaknya elemen
* T.TI untuk mengakses seluruh nilai elemen tabel
* T.TI[i] untuk mengakses elemen ke-i */
/* Definisi :
* Tabel kosong: T.Neff = 0
* Definisi elemen pertama : T.TI[i] dengan i=1
* Definisi elemen terakhir yang terdefinisi: T.TI[i] dengan i=T.Neff */

/* ***** KONSTRUKTOR ***** */
/* Konstruktor : create tabel kosong */
void MakeEmpty (TabKata *T);
/* I.S. sembarang */
/* F.S. Terbentuk tabel T kosong dengan kapasitas IdxMax-IdxMin+1 */

/* ***** SELEKTOR ***** */
/* *** Banyaknya elemen *** */
int NbElmt (TabKata T);
```

```

/* Mengirimkan banyaknya elemen efektif tabel */
/* Mengirimkan nol jika tabel kosong */
/* *** Daya tampung container *** */
int MaxNbEl (TabKata T);
/* Mengirimkan maksimum elemen yang dapat ditampung oleh tabel */
/* *** Selektor INDEKS *** */
IdxType GetFirstIdx (TabKata T);
/* Prekondisi : Tabel T tidak kosong */
/* Mengirimkan indeks elemen pertama */
IdxType GetLastIdx (TabKata T);
/* Prekondisi : Tabel T tidak kosong */
/* Mengirimkan indeks elemen terakhir */
/* *** Menghasilkan sebuah elemen *** */
ElType GetElmt (TabKata T, IdxType i);
/* Prekondisi : Tabel tidak kosong, i antara FirstIdx(T)..LastIdx(T) */
/* Mengirimkan elemen tabel yang ke-i */

/* *** Selektor SET : Mengubah nilai TABEL dan elemen tabel *** */
/* Untuk type private/limited private pada bahasa tertentu */
void SetTab (TabKata Tin, TabKata *Tout);
/* I.S. Tin terdefinisi, sembarang */
/* F.S. Tout berisi salinan Tin */
/* Assignment THsl -> Tin */
void SetEl (TabKata *T, IdxType i, ElType v);
/* I.S. T terdefinisi, sembarang */
/* F.S. Elemen T yang ke-i bernilai v */
/* Mengeset nilai elemen tabel yang ke-i sehingga bernilai v */
void SetNeff (TabKata *T, IdxType N);
/* I.S. T terdefinisi, sembarang */
/* F.S. Nilai indeks efektif T bernilai N */
/* Mengeset nilai indeks elemen efektif sehingga bernilai N */

/* ***** Test Indeks yang valid ***** */
boolean IsIdxValid (TabKata T, IdxType i);
/* Prekondisi : i sembarang */
/* Mengirimkan true jika i adalah indeks yang valid utk ukuran tabel */
/* yaitu antara indeks yang terdefinisi utk container*/
boolean IsIdxEff (TabKata T, IdxType i);
/* Prekondisi : i sembarang */
/* Mengirimkan true jika i adalah indeks yang terdefinisi utk tabel */
/* yaitu antara FirstIdx(T)..LastIdx(T) */

/* ***** TEST KOSONG/PENUH ***** */
/* *** Test tabel kosong *** */
boolean IsEmpty (TabKata T);
/* Mengirimkan true jika tabel T kosong, mengirimkan false jika tidak */
/* *** Test tabel penuh *** */
boolean IsFull (TabKata T);
/* Mengirimkan true jika tabel T penuh, mengirimkan false jika tidak */

```



```

/* ***** BACA dan TULIS dengan INPUT/OUTPUT device ***** */
void TulisIsi (TabKata T);
/* Proses : Menuliskan isi tabel dengan traversal */
/* I.S. T boleh kosong */
/* F.S. Jika T tidak kosong : indeks dan elemen tabel ditulis berderet
ke bawah */
/* Jika isi tabel [1,2,3] maka akan diprint
0:1
1:2
2:3
*/
/* Jika T kosong : Hanya menulis "Tabel kosong" */

/* ***** OPERATOR ARITMATIKA ***** */
/* *** Aritmatika tabel : Penjumlahan, pengurangan, perkalian, ... ***
*/
TabKata PlusTab (TabKata T1, TabKata T2);
/* Prekondisi : T1 dan T2 berukuran sama dan tidak kosong */
/* Mengirimkan T1 + T2 */
TabKata MinusTab (TabKata T1, TabKata T2);
/* Prekondisi : T1 dan T2 berukuran sama dan tidak kosong */
/* Mengirimkan T1 - T2 */

/* ***** NILAI EKSTREM ***** */
int ValMax (TabKata T);
/* Prekondisi : Tabel T tidak kosong */
/* Mengirimkan nilai maksimum tabel */

int ValMin (TabKata T);
/* Prekondisi : Tabel T tidak kosong */
/* Mengirimkan nilai minimum tabel */

/* *** Mengirimkan indeks elemen bernilai ekstrem *** */
IdxType IdxMaxTab (TabKata T);
/* Prekondisi : Tabel T tidak kosong */
/* Mengirimkan indeks i dengan elemen ke-i adalah nilai maksimum pada
tabel */

IdxType IdxMinTab (TabKata T);
/* Prekondisi : Tabel tidak kosong */
/* Mengirimkan indeks i */
/* dengan elemen ke-i nilai minimum pada tabel */

boolean isMember(TabKata T, ElType X);

void AddElmt(TabKata *T, ElType X);

#endif

```

### 3.5.2. Persoalan yang Diselesaikan

ADT Array sangat terpakai di salah satu function register, yaitu registerUser dimana function tersebut menyimpan nama user dan passwordnya di tab array. Register nantinya akan berhubungan dengan fungsi login dan logout.

### 3.5.3. Alasan Pemilihan

Alasan Array digunakan dibanding ADT list biasanya karena sejatinya fitur register perlu menyimpan username dan password.

## 4 Program Utama

Program utama dimulai dengan include modul modul yang diperbolehkan serta include semua ADT yang terpakai di program ini. Terdapat 2 modul utama yang terpakai yaitu <stdio.h> dan <stdlib.h> dan semua ADT file yang berjumlah 6 ADT, yaitu barang.h, list\_dinamis.h, mesinkarakter.h, mesinkata.h, queue.h, random.h, dan store.h.

Masuk ke bagian utama program, pertama tama dimulai dengan deklarasi queue, list, dan list dinamis sebagai container pertama terhadap segala elemen, baik mengenai user ataupun barang dalam store. Lalu dilanjut dengan looping *while(true)* agar sebuah looping terus berjalan hingga user ingin menyudahinya, di dalam blok loop terdapat pengkondisian dengan command yang mungkin diinput oleh user beserta pemanggilan masing masing fungsi yang terkait dengan command.

Terdapat beberapa command utama, yaitu START untuk memulai sesi, LOAD untuk menginput file konfigurasi, REGISTER untuk membuat akun baru dengan username yang unik, LOGIN untuk masuk ke program sebagai sebuah akun untuk menggunakan fitur fitur spesifik lainnya, STORE LIST untuk menampilkan barang yang tersedia di toko, STORE REQUEST untuk input barang baru ke antrean supply barang, STORE SUPPLY untuk memasukkan barang baru ke list store, WORK memungkinkan user mendapatkan uang sebagai imbalan dari melakukan sebuah pekerjaan, WORK CHALLENGE memungkinkan user memainkan mini games yang imbalannya adalah uang, terdapat 3 mini games utama yaitu WORDL3, TEBAK ANGKA, dan QUANTUM WORDL3, LOGOUT untuk keluar dari program sebagai sebuah akun, SAVE untuk menyimpan segala konfigurasi yang terjadi, baik update barang atau update user, dan QUIT untuk benar benar keluar dari program.

## 5 Algoritma-Algoritma Menarik

### 5.1 Algoritma Random

Algoritma Random merupakan salah satu algoritma yang menarik untuk diterapkan karena algoritma random ini menggunakan fungsi time yang merujuk pada waktu saat ini sebagai seed dari random yang akan digunakan. Algoritma random ini digunakan dalam beberapa spesifikasi seperti spesifikasi tebak angka dan spesifikasi wordl3.

**Algoritma :**

STEI- ITB	<nomor dokumen>	Halaman 18 dari 39 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.		

```

#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include "random.h"
#include "../mesinkata/mesinkata.h"

int Random(){
    int max_num = 100;
    srand(time(0));

    int random = (rand() % max_num) + 1;

    return random;
}

char* katarandom(const char *words[], int wordCount) {
    // Map the random number (1 to 100) to a valid index (0 to wordCount
- 1)
    int randomIndex = (Random() - 1) % wordCount;

    // Explicitly cast away the const qualifier
    return (char*)words[randomIndex];
}

```

## 6 Data Test

### 6.1 Data Test START

Fitur yang ditest : START  
 Hasil yang diharapkan :

```

>> START
====[Welcome To PURRMART]====
1. REGISTER
2. LOGIN
3. LOGOUT

```

#### 4. QUIT

Ketik "HELP" for more information

Hasil :

```
>> START
====[Welcome To PURRMART]====
  1. REGISTER
  2. LOGIN
  3. LOGOUT
  4. QUIT
Ketik "HELP" for more information
```

## 6.2 Data Test LOAD

Fitur yang ditest : LOAD

Hasil yang diharapkan :

```
>> LOAD <file.txt>
File konfigurasi tidak ditemukan
>> LOAD <file.txt>
File konfigurasi berhasil dibaca. PurrMart dijalankan
```

Hasil :

```
>> LOAD
D.TXT
Save file berhasil dibaca. PURRMART berhasil dijalankan.
```

## 6.3 Data Test REGISTER

Fitur yang ditest : REGISTER

Hasil yang diharapkan :

```
>> REGISTER
// test username berada di database
username : AgenPerry
password : 123

Akun dengan username AgenPerry sudah terdaftar. Silakan lakukan REGISTER
ulang.

>> REGISTER
```

```
//test username unik dan baru
```

```
username : AgenPerry
```

```
password : 123
```

Akun dengan username AgenPerry telah berhasil dibuat. Silakan LOGIN untuk melanjutkan.

Hasil :

```
>> REGISTER
```

```
Username: LMFAO
```

```
Password: AWOK
```

```
Akun dengan username 'LMFAO' telah berhasil dibuat. Silakan LOGIN untuk melanjutkan.
```

```
Daftar Pengguna:
```

```
1. Username: IBAY, Password: 12, Uang: 0
```

```
2. Username: LMFAO, Password: AWOK, Uang: 0
```

## 6.4 Data Test LOGIN

Fitur yang dites : LOGIN

Hasil yang diharapkan :

```
>> LOGIN
```

```
//test case username dan password benar
```

```
username : AgenPerry
```

```
password : 123
```

```
Anda telah login ke PURRMART sebagai AgenPerry
```

```
>> LOGIN
```

```
//test case username dan password salah
```

```
username : AgenPerry
```

```
password : 1010101
```

```
Username atau password salah.
```

```
>> LOGIN
```

```
//test case terdapat akun yang masih dalam keadaan IsLoggedIn
```

```
username : AgenGanteng
```

```
password : 123
```

```
Anda masih tercatat sebagai AgenPerry. Silakan LOGOUT terlebih dahulu.
```

Hasil :

```
>> LOGIN
Username: LMFAO
Password: AWOK
Anda telah login ke PURRMART sebagai LMFAO.
```

## 6.5 Data Test STORE LIST

Fitur yang ditest : STORE LIST  
Hasil yang diharapkan :

```
>> STORE LIST
//test case store kosong
TOKO KOSONG.

>> STORE LIST
//test case store tidak kosong
List barang yang ada di toko :
- Platypus Laser
- Ambalabu
```

Hasil :  
//test case kosong

```
>>STORE LIST
TOKO KOSONG
```

//test case store tidak kosong

```
>>STORE LIST
List barang yang ada di toko:
- PLATYPUS LASER
```

## 6.6 Data Test STORE REQUEST

Fitur yang ditest : STORE REQUEST  
Hasil yang diharapkan :

```
>> STORE REQUEST
//test case barang yang ingin dimasukkan sudah ada di list toko
Masukkan barang yang ingin diminta : AdaDiToko
Barang AdaDiToko sudah ada di toko !

>> STORE REQUEST
//test case barang yang ingin dimasukkan sudah ada di antrian
Masukkan barang yang ingin diminta : AdaDiAntrian
```

```
Barang AdaDiAntrian sudah ada di antrian!
```

```
>> STORE REQUEST
//test case barang tidak ada di toko dan tidak ada di antrian
Masukkan barang yang ingin diminta : Platypus Laser
Permintaan untuk Platypus Laser ditambahkan ke antrian.
```

Hasil :  
//test case barang tidak ada di toko dan tidak ada di antrian

```
>>STORE REQUEST
Masukkan nama barang yang ingin diminta: PLATYPUS LASER
Permintaan untuk PLATYPUS LASER telah ditambahkan ke antrian.
```

//test case barang ada di toko

```
>>STORE LIST
List barang yang ada di toko:
- PLATYPUS LASER
>>STORE REQUEST
Masukkan nama barang yang ingin diminta: PLATYPUS LASER
Barang PLATYPUS LASER sudah ada di toko!
```

//test case barang ada diantrian

```
>>STORE REQUEST
Masukkan nama barang yang ingin diminta: AdaDiAntrian
Permintaan untuk AdaDiAntrian telah ditambahkan ke antrian.

>>STORE REQUEST
Masukkan nama barang yang ingin diminta: AdaDiAntrian
Barang AdaDiAntrian sudah ada di antrian!
```

## 6.7 Data Test STORE SUPPLY

Fitur yang dites : STORE SUPPLY

Hasil yang diharapkan :

```
>> STORE SUPPLY
// test case jika antrean barang kosong
```

Antrean kosong ! tidak ada barang yang disupply

**>> STORE SUPPLY**

**// test case jika antrean barang di terima, misal barang bernama Platypus Laser ( sebagai head of queue )**

Barang dalam antrean : AdaDiAntrian

Apakah kamu ingin menambahkan barang AdaDiAntrian : Terima

Masukkan harga barang : 100

Barang berhasil dimasukkan ke store.

**>> STORE SUPPLY**

**// test case jika antrean barang di tunda, misal barang bernama Platypus Laser ( sebagai head of queue )**

Barang dalam antrean : Platypus Laser

Apakah kamu ingin menambahkan barang Platypus Laser : Tunda

Platypus Laser dikembalikan ke antrian.

**>> STORE SUPPLY**

**// test case jika antrean barang di tolak, misal barang bernama Platypus Laser ( sebagai head of queue )**

Barang dalam antrean : Ayam Rebus

Apakah kamu ingin menambahkan barang Platypus Laser : Tolak

Ayam Rebus dihapuskan dari antrian.

Hasil :

//test case menerima barang dari antrean

**>>STORE SUPPLY**

Barang dalam antrean: AdaDiAntrian

Apakah kamu ingin menambahkan barang AdaDiAntrian : Terima

Masukkan harga barang: 100

AdaDiAntrian dengan harga 100 telah ditambahkan ke toko.

**>>STORE LIST**

List barang yang ada di toko:

- PLATYPUS LASER

- AdaDiAntrian

//test case antrean kosong



```
>>STORE SUPPLY
Antrean kosong! Tidak ada barang untuk disuplai.
```

//test case menolak dan menunda barang dari antrean

```
>>STORE REQUEST
Masukkan nama barang yang ingin diminta: AYAM GORENG
Permintaan untuk AYAM GORENG telah ditambahkan ke antrian.

>>STORE REQUEST
Masukkan nama barang yang ingin diminta: AYAM REBUS
Permintaan untuk AYAM REBUS telah ditambahkan ke antrian.

>>STORE SUPPLY
Barang dalam antrean: AYAM GORENG
Apakah kamu ingin menambahkan barang AYAM GORENG : Tunda

AYAM GORENG dikembalikan ke antrian.
>>STORE SUPPLY
Barang dalam antrean: AYAM REBUS
Apakah kamu ingin menambahkan barang AYAM REBUS : Tolak

AYAM REBUS dihapuskan dari antrian.
```

//test case command = purry

```
>>STORE SUPPLY
Barang dalam antrean: AYAM GORENG
Apakah kamu ingin menambahkan barang AYAM GORENG : Purry

<Kembali ke menu>
```

## 6.8 Data Test STORE REMOVE

Fitur yang ditest : STORE REMOVE

Hasil yang diharapkan :

```
>> STORE REMOVE
// test case jika tidak ada barang bernama "Ayam Goreng" di toko

Masukkan nama barang yang ingin dihapus dari toko: Ayam Goreng
Toko tidak menjual Ayam Goreng

>> STORE REMOVE
// test case jika ada barang bernama "PLatypus Laser" di toko

Masukkan nama barang yang ingin dihapus dari toko: Platypus Laser
Platypus Laser berhasil dihapus.
```

Hasil :

STEI- ITB	<nomor dokumen>	Halaman 25 dari 39 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.		

```

>>STORE LIST
List barang yang ada di toko:
- PLATYPUS LASER
- AdaDiAntrian
>>STORE REMOVE
Masukkan nama barang yang ingin dihapus dari toko: PLATYPUS LASER
PLATYPUS LASER telah berhasil dihapus.

>>STORE LIST
List barang yang ada di toko:
- AdaDiAntrian
>>STORE REMOVE
Masukkan nama barang yang ingin dihapus dari toko: AYAM GORENG
Toko tidak menjual AYAM GORENG.

>>STORE LIST
List barang yang ada di toko:
- AdaDiAntrian
>>

```

## 6.9 Data Test WORK

Fitur yang dites : WORK  
 Hasil yang diharapkan :

```

>> WORK
Daftar Pekerjaan:
  1. Evil Lab Assistant (Pendapatan=100, durasi=14s)
  2. OWCA Hiring Manager (Pendapatan=4200, durasi=21s)
  3. Cikapundunginator Caretaker (Pendapatan=7000, durasi=30s)
  4. Mewing Specialist (Pendapatan=10000, durasi=22s)
  5. Inator Connoisseur (Pendapatan=997, durasi=15s)

Masukkan Pekerjaan yang dipilih: Evil Lab Assistant
Anda sedang bekerja sebagai Evil Lab Assistant... harap tunggu.
Pekerjaan selesai, +100 rupiah telah ditambahkan ke akun anda.

```

Hasil :

```

>> WORK
Daftar Pekerjaan:
  1. Evil Lab Assistant (Pendapatan=100, durasi=14s)
  2. OWCA Hiring Manager (Pendapatan=4200, durasi=21s)
  3. Cikapundunginator Caretaker (Pendapatan=7000, durasi=30s)
  4. Mewing Specialist (Pendapatan=10000, durasi=22s)
  5. Inator Connoisseur (Pendapatan=997, durasi=15s)

Masukkan Pekerjaan yang dipilih: Evil Lab Assistant
Anda sedang bekerja sebagai Evil Lab Assistant... harap tunggu.
Pekerjaan selesai, +100 rupiah telah ditambahkan ke akun anda.

```

## 6.10 Data Test WORK CHALLENGE

Fitur yang dites : WORK CHALLENGE

Hasil yang diharapkan :

```
>> WORK CHALLENGE
Daftar challenge ayng tersedia:
1. Tebak Angka (biaya main=200)
2. WORDL399 (biaya main=500)
3. QuantumWordl3 (biaya main = 1000)

Masukkan challenge yang hendak dimainkan:
```

Hasil :

```
>> WORK CHALLENGE
Daftar challenge ayng tersedia:
1. Tebak Angka (biaya main=200)
2. WORDL399 (biaya main=500)
3. QuantumWordl3 (biaya main = 1000)

Masukkan challenge yang hendak dimainkan: Tebak Angka

Tebak angka: 10
Tebakanmu lebih kecil!

Tebak angka: 11
Tebakanmu lebih kecil!

Tebak angka: 50
Tebakanmu lebih kecil!

Tebak angka: 70
Tebakanmu lebih besar!

Tebak angka: 60
Tebakanmu lebih kecil!

Tebak angka: 65
Tebakanmu lebih kecil!

Tebak angka: 67
Tebakanmu lebih kecil!

Tebak angka: 68
Tebakanmu Benar! +150 rupiah telah ditambahkan ke akun anda.
```

```
Masukkan challenge yang hendak dimainkan: WORDL399
=====WORDLE=====
Welcome to WORDLE.
You have 5 chance to guess the word.
Good luck!
!char! is Right placed
(char) is contains the character
|char| is false
Input WORD (max 5 char) :
>> TESTS
(T) (E) (S) (T) (S)
Input WORD (max 5 char) :
>> ADIEU
(A) |D| |I| (E) |U|
Input WORD (max 5 char) :
>> BERRY
|B| (E) |R| |R| |Y|
Input WORD (max 5 char) :
>> HORSE
|H| |O| |R| (S) (E)
Input WORD (max 5 char) :
>> TEST
The length of the string must be 5. Please try again.
>> STATE
!S! !T! (A) (T) (E)
Input WORD (max 5 char) :
>> START
!S! !T! (A) |R| (T)

The correct string is: STEAK
Uh-oh You lose, better luck next time!
=====
```

## 7 Test Script

Isi dengan skenario test yang dimungkinkan untuk semua fitur yang ada. Bisa dibuat dalam bentuk tabel sebagai berikut:

No.	Fitur yang Dites	Tujuan Testing	Langkah-Langkah Testing	Input Data Test	Hasil yang Diharapkan	Hasil yang Keluar
1	START	Memeriksa apakah file konfigurasi default berhasil dibaca dan menampilkan	Pengguna memasukkan command "START".	TEST START	Program berjalan.	Program berjalan.

		tampilan / pesan apabila file konfigurasi berhasil dibuka				
2	LOAD	Memeriksa apakah file konfigurasi berhasil dibaca dan menampilkan tampilan / pesan apabila file konfigurasi berhasil dibuka	Pengguna memasukkan command "LOAD".	TEST LOAD	file .txt terdeteksi dan isi file tersebut dapat dibaca.	file .txt terdeteksi dan isi file tersebut dapat dibaca.
3	REGISTER	Memeriksa input implementasi mesin kata dan edge case dari register seperti username yang harus unik serta melihat langsung isi database username yang sementara disimpan dari list statik	Pengguna memasukkan command "REGISTER"	<a href="#">TEST REGISTER</a>	User baru terdaftar dan dapat digunakan untuk login.	User baru terdaftar dan dapat digunakan untuk login.
4	LOGIN	Memeriksa apakah username dan password yang di assign user dapat diperiksa dengan tepat oleh fungsi yang membandingkan data di list database user dan input user	Memasukkan command "LOGIN"	<a href="#">TEST LOGIN</a>	User dapat melakukan login dan menggunakan command-command lain yang sudah disediakan.	User dapat melakukan login dan menggunakan command-command lain yang sudah disediakan.
5	STORE LIST	Memeriksa apakah function dapat benar benar menampilkan seluruh isi toko, baik saat terdapat barang di toko maupun tidak	Memasukkan command "STORE LIST"	<a href="#">TEST STORE LIST</a>	Fungsi akan menampilkan barang yang ada di store list ( array dinamis ). Cek edge case seperti ketika toko kosong ( list kosong ) dan toko tidak kosong.	Fungsi akan menampilkan barang yang ada di store list ( array dinamis ). Cek edge case seperti ketika toko kosong ( list kosong ) dan toko tidak kosong.
6	STORE REQUEST	Memeriksa apakah function dapat benar benar	Memasukkan command "STORE REQUEST"	<a href="#">TEST STORE REQUEST</a>	Fungsi menampilkan teks yang	Fungsi menampilkan teks yang

		memasukkan barang baru ke dalam antrian serta mengecek edge case nya			berupa pertanyaan barang apa yang ingin dimasukkan ke antrian barang. Setelah barang ditambahkan, tentunya terdapat pesan yang menandakan barang telah masuk ke antrian.	berupa pertanyaan barang apa yang ingin dimasukkan ke antrian barang. Setelah barang ditambahkan, tentunya terdapat pesan yang menandakan barang telah masuk ke antrian.
7	STORE SUPPLY	Memeriksa apakah function dapat benar benar memasukkan barang baru di antrian dengan beberapa command spesifik seperti TERIMA, TUNDA, TOLAK, dan PURRY	Memasukkan command “STORE SUPPLY”	<a href="#">TEST STORE SUPPLY</a>	Fungsi dapat menampilkan HEAD OF QUEUE dari barang yang akan disupply ke store list. Pengecekan dimulai secara bertahap dari command ke command ( Terima, Tolak, Purry, dan Tunda ).	Fungsi dapat menampilkan HEAD OF QUEUE dari barang yang akan disupply ke store list. Pengecekan dimulai secara bertahap dari command ke command ( Terima, Tolak, Purry, dan Tunda ).
8	WORK	Memeriksa apakah function dapat benar benar memproses input berupa pilihan pekerjaan yang mana dan memastikan bahwa fitur waktu sebagai pembeda masing masing jenis dari work dapat bekerja	Pengguna memasukkan perintah “WORK” lalu memilih pekerjaan yang tertera.	<a href="#">TEST WORK</a>	Pengguna menunggu sesuai dengan keterangan waktu yang tertera, kemudian atribut uang dari pengguna bertambah.	Pengguna menunggu hampir sesuai dengan keterangan waktu yang tertera, kemudian atribut uang dari pengguna bertambah.
9	WORK CHALLENGE	memeriksa berjalannya masing masing minigames yang ada dan memastikan fitur RNG dapat berjalan	Pengguna akan memasukkan command “WORK CHALLENGE” lalu memasukkan nama minigames yg ingin dimainkan.	<a href="#">TEST WORK CHALLENGE</a>	Permainan yang berjalan sempurna dan minimum dengan adanya bug, lalu dengan pengembalian uang sesuai	Permainan yang berjalan sempurna dan minimum dengan adanya bug, lalu dengan pengembalian uang sesuai

					dengan isi permainan	dengan isi permainan
10	LOGOUT					
11	SAVE					
12	QUIT					

## 8 Pembagian Kerja dalam Kelompok

Nama Lengkap - NIM	Deskripsi Tugas
Mochamad Ikhbar Adiwinangun - 18223050	<ul style="list-style-type: none"> <li>- Mengerjakan fitur store dan membuat struct barang</li> <li>- Mengerjakan ADT queue, list dinamis, dan membuat tambahan fungsi minor di mesin kata</li> <li>- Mengerjakan laporan</li> </ul>
Derick Amadeus Budiono - 18223090	<ul style="list-style-type: none"> <li>- Mengerjakan fitur wordl3 dan quantum wordl3</li> <li>- Mengerjakan, merombak dan menambah berbagai fitur pada ADT mesinkarakter dan mesinkata.</li> <li>- Mengerjakan laporan</li> </ul>
Indana Aulia Ayundazulfa - 18223100	<ul style="list-style-type: none"> <li>- Mengerjakan load dan save</li> <li>- mengerjakan laporan</li> </ul>
Wilson - 18223012	<ul style="list-style-type: none"> <li>- Mengerjakan fitur Work, tebak angka dan melakukan merging serta debugging terhadap kode keseluruhan</li> <li>- mengerjakan laporan</li> </ul>
Naila Selvira Budiana - 18223018	<ul style="list-style-type: none"> <li>- membuat repositori github</li> <li>- mengerjakan adt array</li> <li>- mengerjakan fitur login, logout</li> <li>- mengerjakan laporan</li> </ul>
satu lagi gatau kemana	

## 9 Lampiran

### 9.1 Deskripsi Tugas Besar

## Spesifikasi Umum

Buatlah sebuah aplikasi simulasi 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. Daftar ADT yang wajib digunakan dapat dilihat pada bagian Daftar ADT. *Library* yang boleh digunakan hanya **stdio.h**, **stdlib.h**, **time.h**, dan **math.h**.

## System Mechanic

### 1. About the System

PURRMART adalah sebuah aplikasi yang dapat mensimulasikan aktivitas beli barang pada *e-commerce*. PURRMART memiliki beberapa fitur utama, yaitu:

- Menampilkan barang toko
- Meminta dan menyuplai barang baru ke toko
- Menyimpan dan membeli barang dalam keranjang
- Menampilkan barang yang sudah dibeli
- Membuat dan menghapus *wishlist*
- Bekerja untuk menghasilkan uang

### 2. Menu Program

Ketika program pertama kali dijalankan, PURRMART akan memperlihatkan *main menu* yang berisi **welcome menu** dan beberapa *command* yaitu **START**, **LOAD**, dan juga **HELP**.

Setelah itu, program akan memasuki *login menu* yang memiliki command **LOGIN**, **REGISTER**, dan juga **HELP**. Jika pengguna berhasil memasuki kredensial suatu akun, maka mereka akan masuk ke menu selanjutnya.

**Main menu** menerima masukan berupa *command* yang akan dijelaskan pada bagian berikutnya. Program akan terus menerima *command* sampai diberikan *command* **QUIT** yang berlaku pada seluruh menu.

STEI- ITB	<nomor dokumen>	Halaman 31 dari 39 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.		

### 3. Command

Pengguna dapat memasukkan *command-command* berikut.

**a. START**

START merupakan salah satu *command* yang dimasukkan pertama kali dalam Toko PURRMART. Setelah menekan Enter, dibaca file konfigurasi *default* yang berisi daftar barang pada toko.

**b. LOAD <filename>**

LOAD merupakan salah satu *command* yang dimasukkan pertama kali dalam PURRMART. Command ini memiliki satu argumen yaitu *filename* yang merepresentasikan suatu *save file* yang ingin dibuka. *File* didapatkan dari *folder* tertentu, contohnya *save*. Setelah menekan *Enter*, akan dibaca *save file <filename>* yang berisi daftar barang pada toko. Lebih detailnya bisa dilihat pada Konfigurasi Aplikasi.

**c. LOGIN**

Login merupakan *command* yang baru dapat dipanggil setelah pengguna memulai sesi. *Login* berguna untuk masuk ke akun di sistem PURRMART yang sudah didaftarkan sebelumnya.

**d. LOGOUT**

LOGOUT merupakan salah satu *command* yang baru dapat digunakan setelah pengguna telah memasuki sebuah sesi.

**e. REGISTER**

Register merupakan *command* yang baru dapat dipanggil setelah pengguna memulai sesi. *Register* berguna untuk mendaftarkan akun baru ke dalam sistem PURRMART. Sebuah akun setidaknya memiliki atribut *username* dan *password*. **Username dan password hanya terdiri dari 1 kata.**

**f. WORK**

WORK merupakan *command* yang digunakan pengguna untuk mendapatkan uang. Terdapat sejumlah pekerjaan yang bisa dipilih. Setiap pekerjaan memiliki waktu tunggu yang berbeda-beda dan dengan nominal pendapatan yang berbeda-beda pula. Selama pengguna sedang bekerja, maka sistem tidak bisa digunakan hingga pekerjaan selesai dilakukan.

**g. WORK CHALLENGE**



WORK CHALLENGE merupakan *command* alternatif sebagai cara mendapatkan uang dengan melakukan *challenge-challenge* di OWCA. Pemain membutuhkan uang dengan jumlah tertentu untuk bisa memainkan challenge. Uang yang dibayarkan untuk bermain *challenge* tidak akan dikembalikan, meskipun pemain kalah dalam permainan. Terdapat dua *challenge* yang dapat dipilih:

**a) Tebak Angka**

Challenge Tebak Angka merupakan permainan yang meminta pemain menebak sebuah angka yang ditentukan oleh program. Pemain memiliki 10 (sepuluh) kesempatan untuk menebak angka yang benar. Program akan memberikan *feedback* apakah angka tebakannya lebih besar, lebih kecil, atau sama dengan angka target. Jumlah kesempatan yang dipakai oleh pengguna akan mempengaruhi uang yang didapatkan.

**b) WORDL3**

*Challenge* WORDL3 merupakan permainan tebak kata berjumlah lima karakter. Pemain memiliki 6 (enam) kesempatan untuk menebak kata yang benar. Kata harus berupa kata valid, tidak boleh sekadar *string* acak, bahasa dibebaskan (disarankan bahasa Indonesia/Inggris). Pada setiap giliran, program akan mencetak ulang kata yang dimasukkan, tetapi dengan penanda tertentu. Huruf yang benar dan berada pada tempat yang tepat diberi tanda “!char!”. Huruf yang benar, tetapi berada di tempat yang salah diberi tanda “(char)” setelah hurufnya. Huruf yang tidak ada sama sekali pada kata diberi tanda “|char|” setelah hurufnya.

**h. STORE LIST**

STORE LIST adalah *command* yang digunakan untuk melihat barang-barang apa saja yang ada di dalam toko. **Setiap barang yang ditampilkan haruslah bersifat *unique*.**

**i. STORE REQUEST**

STORE REQUEST adalah *command* yang digunakan untuk meminta penambahan barang baru ke dalam toko. Barang-barang yang diminta akan disimpan di dalam sebuah antrian dan akan dimasukkan ke toko menggunakan *command* selanjutnya. **Nama barang yang masuk tidak boleh sama dengan nama barang yang sudah ada di toko atau di antrian.**

**j. STORE SUPPLY**

STORE SUPPLY adalah *command* yang digunakan untuk menambahkan barang baru ke dalam toko berdasarkan antrian permintaan. Barang yang berada pada antrian paling

depan akan dimasukkan ke toko. Pengguna dapat menerima, menunda, atau menolak permintaan.

- Jika diterima, maka program akan meminta harga dari barang dan dimasukkan ke toko.
- Jika ditunda, maka barang akan kembali masuk ke antrian
- Jika ditolak, maka barang akan dihapus dari antrian

Harus terdapat validasi agar harga barang merupakan angka yang valid (berupa angka dan bernilai lebih dari nol).

#### k. **STORE REMOVE**

STORE REMOVE adalah *command* yang dapat menghapus barang yang ada di toko. Akan dilakukan *input* akan barang yang akan dihapus. Beri tahu apabila proses berhasil (barang terdapat pada toko dan berhasil dihapus) ataupun tidak (barang tidak terdapat di toko).

#### l. **HELP**

HELP merupakan *command* yang digunakan menampilkan daftar *command* yang mungkin untuk dieksekusi dengan deskripsinya. Penjelasan dari deskripsi dibebaskan selama masih mendeskripsikan *command* sesuai dengan spek.

#### m. **SAVE <filename>**

SAVE merupakan *command* yang digunakan untuk menyimpan *state* aplikasi terbaru ke dalam suatu *file*. Command SAVE memiliki satu argumen yang merepresentasikan nama *file* yang akan disimpan. Penyimpanan dilakukan pada *folder* tertentu, misal *folder save*.

#### n. **QUIT**

QUIT merupakan *command* yang digunakan untuk keluar dari sesi aplikasi PURRMART.

## 9.2 *Notulen Rapat*







**Form Asistensi Tugas Besar**  
**IF2111/Algoritma dan Struktur Data STI**  
**Sem. 1 2024/2025**

No. Kelompok/Kelas : 11/K2  
Nama Kelompok :  
Anggota Kelompok (Nama/NIM) :  
1. Derick Amadeus B / 18223090  
2. Wilson / 18223012  
3. Mochamad Ikhbar A / 18223050

STEI- ITB	<nomor dokumen>	Halaman 34 dari 39 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.		







4. Indana Aulia Ayundazulfa / 18223100  
 5. Naila Selvira Budiana / 18223018

Asisten Pembimbing : Jonathan Arthurito Aldi Sinaga

Tanggal : 19 November 2024			<b>Catatan Asistensi:</b>  Rapikan hierarki file di github. Pastikan ADT yang terpakai di masing-masing fitur sudah di commit di branch khusus supaya fungsi yang digunakan sama dan tidak menyulitkan saat merge.
Tempat : Daring via Gmeets			
Kehadiran Anggota Kelompok:			
No	NIM	Tanda Tangan	
1	18223050		
2	18223090		
3	18223100		
4	18223012		
5	18223018		
			<b>Tanda Tangan Asisten:</b> 

## Asistensi II

<b>Tanggal : 23 November 2024</b>			<b>Catatan Asistensi:</b>  Beri komentar yang sama, rapikan github, dan pastikan tidak ada conflict saat melakukan merge.
<b>Tempat : Daring via Gmeets</b>			
<b>Kehadiran Anggota Kelompok:</b>			
No	NIM	Tanda Tangan	

1	18223050		
2	18223090		
3	18223100		
4	18223012		
5	18223018		
			<b>Tanda Tangan Asisten:</b> 

### 9.3 Log Activity Anggota Kelompok

<u>No</u>	<u>Tanggal</u>	<u>NIM</u>	<u>Nama</u>	<u>Aktivitas</u>
1	13/11/2024	18223050 18223090 18223100 18223012 18223018	M. Ikhbar A Derick Amadeus Wilson Indana Aulia Naila Selvira	Membuat repository Github sekaligus diskusi pembagian tugas
2	17/11/2024	18223050	M. Ikhbar A	Commit pertama ADT mesinkata, ADT mesinkarakter, ADT queue, ADT list dinamis, dan ADT barang serta store sebagai fitur utama di branch store
	18/11/2024	18223012	Wilson	Commit pertama file random dan tebak angka di branch pribadi
	19/11/2024	18223050 18223090 18223100 18223012 18223018	M. Ikhbar A Derick Amadeus Wilson Indana Aulia Naila Selvira	Asistensi ke-1
	19/11/2024	18223012	Wilson	Fix random
	20/11/2024	18223012	Wilson	Fix bug, fix work, dan beberapa update di branch
	20/11/2024	18223090	Derick Amadeus	Update ADT mesinkarakter di branch ADT

	21/11/2024	18223090 18223050	Derick Amadeus M Ikhbar A	Update ADT dan driver di branch ADT
	15/11/2024	18223018	Naila Selvira B	debugging ADT mesin kata, membuat fitur login logout register
	21/11/2024	18223018	Naila Selvira B	commit fitur login logout register
	22/11/2024	18223050	M. Ikhbar A	update ADT di branch store
	22/11/2024	18223090	Derick Amadeus	menambah 2 driver di branch ADT
	23/11/2024	18223050	M. Ikhbar A	Fix bug store
	23/11/2024	18223018	Naila Selvira B	debugging fitur login logout register dan commit ke github
	23/11/2024	18223090	Derick Amadeus	Menambah driver dan deskripsi fungsi di branch ADT
	23/11/2024	18223050 18223090 18223100 18223012 18223018	M. Ikhbar A Derick Amadeus Wilson Indana Aulia Naila Selvira	Asistensi ke-2
	23/11/2024	18223100	Indana Aulia	Commit pertama feature load, user, dan save di branch
	24/11/2024	18223100	Indana Aulia	Update load dan commit beberapa ADT ke branch
	24/11/2024	18223090 18223100	Derick Amadeus Wilson Indana Aulia	Debugging bersama dan merge masing masing branch ke

		18223012 18223018	Naila Selvira	main
	25/11/2024	18223100	Indana Aulia	Fix feature save
	25/11/2024	18223050 18223090 18223100 18223012 18223018	M. Ikhbar A Derick Amadeus Wilson Indana Aulia Naila Selvira	Debugging bersama dan mengerjakan laporan