

**LAPORAN TUGAS BESAR**  
**IF2111 Algoritma dan Struktur Data STI**


**PURRMART**

Dipersiapkan oleh:  
Kelompok 01:

Bagas Noor Fadhilah	(18223115)
Khairunnisa Azizah	(18223117)
Maria Vransiska P. C. T. D. P	(18223119)
Theresia Ivana M. S	(18223126)
Aulia Azka Azzahra	(18223131)

Sekolah Teknik Elektro dan Informatika - Institut Teknologi Bandung

Jl. Ganesha 10, Bandung 40132

	<b>Sekolah Teknik Elektro dan Informatika ITB</b>	Nomor Dokumen		Halaman
		IF2111-TB-03-01		40
		Revisi	-	25/11/2024

## Daftar Isi

<b>Daftar Isi.....</b>	<b>2</b>
<b>1 Ringkasan.....</b>	<b>4</b>
2 Struktur Data (ADT).....	4
2.1 ADT Array.....	4
2.1.1 Sketsa Struktur Data.....	4
2.1.2 Persoalan yang Diselesaikan.....	6
2.1.3 Alasan Pemilihan.....	6
2.1.4 Implementasi.....	7
2.2 ADT arraydin.....	7
2.2.1 Sketsa Struktur Data.....	7
2.2.2 Persoalan yang Diselesaikan.....	8
2.2.3 Alasan Pemilihan.....	8
2.2.4 Implementasi.....	8
2.3 ADT boolean.....	8
2.3.1 Sketsa Struktur Data.....	8
2.3.2 Persoalan yang Diselesaikan.....	9
2.3.3 Alasan Pemilihan.....	9
2.3.4 Implementasi.....	9
2.4 ADT mesinkarakter.....	9
2.4.1 Sketsa Struktur Data.....	9
2.4.2 Persoalan yang Diselesaikan.....	10
2.4.3 Alasan Pemilihan.....	10
2.4.4 Implementasi.....	10
2.5 ADT mesinkata.....	10
2.5.1 Sketsa Struktur Data.....	10
2.5.2 Persoalan yang Diselesaikan.....	11
2.5.3 Alasan Pemilihan.....	11
2.5.4 Implementasi.....	12
2.6 ADT queue.....	12
2.6.1 Sketsa Struktur Data.....	12
2.6.2 Persoalan yang Diselesaikan.....	13
2.6.3 Alasan Pemilihan.....	13
2.6.4 Implementasi.....	13
2.7 ADT misc.....	13
2.7.1 Sketsa Struktur Data.....	13
2.7.2 Persoalan yang Diselesaikan.....	15
2.7.3 Alasan Pemilihan.....	15

2.7.4 Implementasi.....	15
<b>3 Program Utama.....</b>	<b>15</b>
<b>4 Algoritma-Algoritma Menarik.....</b>	<b>16</b>
4.1 Implementasi Lain scanf() pada Bahasa C.....	16
<b>5 Data Test.....</b>	<b>17</b>
5.1 START.....	17
5.2 LOAD.....	18
5.3 LOGIN.....	19
5.4 LOGOUT.....	20
5.5 REGISTER.....	20
5.6 WORK.....	22
5.7 WORK CHALLENGE.....	23
5.8 STORE LIST.....	25
5.9 STORE REQUEST.....	26
5.10 STORE SUPPLY.....	27
5.11 STORE REMOVE.....	29
5.12 HELP.....	30
5.13 SAVE.....	31
5.14 QUIT.....	31
<b>6 Test Script.....</b>	<b>32</b>
<b>7 Pembagian Kerja dalam Kelompok.....</b>	<b>35</b>
<b>8 Lampiran.....</b>	<b>36</b>
8.1 Deskripsi Tugas Besar.....	36
8.2 Notulen Rapat.....	36
8.3 Log Activity Anggota Kelompok.....	38

# 1 Ringkasan

Persediaan senjata dan barang OWCA sudah mulai menipis setelah melalui pertarungan sengit melawan Dr. Asep Spakbor selama 3 bulan 13 hari 2 jam 47 menit dan 2 detik. OWCA pun menugaskan tim programmer andalannya untuk merancang aplikasi jual beli barang bernama PURRMART. PURRMART ini dibuat dengan tujuan agar dapat membantu Toko Borma, toko pemasok barang-barang perang, untuk melakukan sistem jual beli secara online.

PURRMART adalah sebuah *e-commerce* berbasis CLI (Command Line Interface) yang diimplementasikan menggunakan bahasa C yang dikembangkan untuk menyelesaikan tugas besar untuk mata kuliah IF2111 Algoritma dan Struktur Data. Fitur-fitur utama PURRMART adalah 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*, serta bekerja untuk menghasilkan uang. Implementasi ADT menjadi dasar dalam penerapan fitur-fitur tersebut.

Laporan ini berisi penjelasan secara rinci mengenai aplikasi *e-commerce* yang telah dibuat, beserta spesifikasi-spesifikasi yang terdiri dari fitur-fitur utama, implementasi ADT yang telah dimodifikasi, serta algoritma yang digunakan dalam aplikasi PURRMART. Pada bagian akhir laporan, dilampirkan hasil pengetesan program yang dilakukan untuk memastikan fungsionalitas aplikasi PURRMART. Dengan demikian, tugas besar ini membantu memahami dan mengimplementasikan ADT dalam konsep yang berbeda, yaitu sebagai sebuah aplikasi *e-commerce* berbasis *command-line*.

## 2 Struktur Data (ADT)

### 2.1 ADT Array

#### 2.1.1 Sketsa Struktur Data

```
#include "../boolean/boolean.h"

#ifndef ARRAY_H
#define ARRAY_H

/* Kamus Umum */

#define IdxMax 50
#define IdxMin 1
#define IdxUndef -999 /* indeks tak terdefinisi*/
#define integer int

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

typedef struct {
    char name[IdxMax];
    char password[IdxMax];
    integer money;
} User;

typedef struct
{
```

```

        User TC[IdxMax-IdxMin+1]; /* memori tempat menyimpan elemen (container) */
        int Neff; /* banyaknya elemen efektif */
    } TabUser;

typedef struct {
    char name[50];
    int income;
    int duration;
} job;

extern job listjob[];
extern int totaljob;

/* Indeks yang digunakan [IdxMin..IdxMax] */
/* Jika T adalah TabInt, cara deklarasi dan akses: */
/* Deklarasi : T : TabInt */
/* 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 (TabUser *T);
/* I.S. sembarang */
/* F.S. Terbentuk tabel T kosong dengan kapasitas IdxMax-IdxMin+1 */

/* Konstruktor :Reset isi dari tabel */
void ResetTabUser (TabUser *T);

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

int GetMoney (TabUser T, IdxType i);
/* Prekondisi : Tabel tidak kosong, i antara FirstIdx(T)..LastIdx(T) */
/* Mengirimkan elemen tabel yang ke-i */

```

```

char* GetPassword (TabUser T, IdxType i);
/* Prekondisi : Tabel tidak kosong, i antara FirstIdx(T)..LastIdx(T) */
/* Mengirimkan elemen tabel yang ke-i */

void SetNeff (TabUser *T, IdxType N);
/* I.S. T terdefinisi, sembarang */
/* F.S. Nilai indeks efektif T bernilai N */
/* Mengeset nilai indeks elemen efektif sehingga bernilai N */

void SetMoney(TabUser *T, IdxType i, int val);
/* I.S. T terdefinisi, sembarang */
/* F.S. Element uang di set */
/* Mengeset nilai indeks elemen efektif sehingga bernilai N */

/* ***** Test Indeks yang valid ***** */
boolean IsIdxValid (TabUser 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 (TabUser 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 IsUserEmpty (TabUser T);
/* Mengirimkan true jika tabel T kosong, mengirimkan false jika tidak */
/* *** Test tabel penuh *** */
boolean IsFull (TabUser T);
/* Mengirimkan true jika tabel T penuh, mengirimkan false jika tidak */

/* ***** BACA dan TULIS dengan INPUT/OUTPUT device ***** */
void TulisIsi (TabUser 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" */

#endif

```

### 2.1.2 Persoalan yang Diselesaikan

ADT array digunakan untuk mengelola data user yang ada di dalam sistem.

### 2.1.3 Alasan Pemilihan

Array statik dipilih untuk menyimpan data pengguna karena diasumsikan jumlah pengguna (ukuran data) tidak akan berubah drastis selama runtime. Lalu operasi seperti menambah elemen, memvalidasi indeks, atau mencetak isi tabel lebih efisien bila diimplementasikan dengan menggunakan array statik.

### 2.1.4 Implementasi

Implementasi array dalam program ini ada di dalam kode `array.c` pada direktori `src/ADT/array/array.c`

## 2.2 ADT arraydin

### 2.2.1 Sketsa Struktur Data

```
#ifndef __ARRAY_DINAMIK__
#define __ARRAY_DINAMIK__

// Boolean
#define boolean unsigned char
#define true 1
#define false 0

#define InitialSize 10
#define MAX_LEN 50
#define integer int

typedef struct {
    char name[MAX_LEN];
    integer price;
} Barang;

typedef int IdxType;
typedef int ElType;
typedef struct {
    Barang *A;
    int Capacity;
    int Neff;
} ArrayDinBarang;

/**
 * Konstruktor
 * I.S. sembarang
 * F.S. Terbentuk ArrayDin kosong dengan ukuran InitialSize
 */
void MakeArrayDinBarang(ArrayDinBarang *array);

/**
 * Destruktor
 * I.S. ArrayDinBarang terdefinisi
 * F.S. array→A terdealokasi
 */
void DeallocateArrayDinBarang(ArrayDinBarang *array);

/**
 * Fungsi untuk mengetahui apakah suatu array kosong.
 * Prekondisi: array terdefinisi
 */
boolean IsEmpty(ArrayDinBarang array);

/**
 * Fungsi untuk mendapatkan banyaknya elemen efektif array, 0 jika tabel kosong.
 * Prekondisi: array terdefinisi
 */
```

```

int Length(ArrayDinBarang array);

/**
 * Fungsi untuk mendapatkan kapasitas yang tersedia.
 * Prekondisi: array terdefinisi
 */
int GetCapacity(ArrayDinBarang array);

/**
 * Fungsi untuk menambahkan elemen baru di index ke-i
 * Prekondisi: array terdefinisi, i di antara 0..Length(array).
 */
void InsertBarang(ArrayDinBarang *array, Barang eI, IdxType i);

/**
 * Fungsi untuk menghapus elemen di index ke-i ArrayDinBarang
 * Prekondisi: array terdefinisi, i di antara 0..Length(array).
 */
void DeleteAt(ArrayDinBarang *array, IdxType i);

/**
 * Fungsi untuk menambah jumlah kapasitas array dinamis
 * Prekondisi: array terdefinisi, i di antara 0..Length(array).
 */
void GrowArray(ArrayDinBarang *array);

/**
 * Fungsi untuk mengurangi jumlah kapasitas array dinamis
 * Prekondisi: array terdefinisi, i di antara 0..Length(array).
 */
void DecreaseArray(ArrayDinBarang *array);

#endif

```

### 2.2.2 Persoalan yang Diselesaikan

ADT arraydin digunakan untuk mengelola data barang.

### 2.2.3 Alasan Pemilihan

Array dinamis dipilih sebagai struktur data karena memudahkan dalam pengelolaan memori secara otomatis dengan memperbesar atau memperkecil ukuran array sesuai kebutuhan sehingga lebih fleksibel tanpa harus menghapus data barang jika array sudah penuh.

### 2.2.4 Implementasi

Implementasi arraydin terdapat pada kode arraydin.c pada direktori src/ADT/arraydin.c

## 2.3 ADT boolean

### 3.3.1 Sketsa Struktur Data

```

#ifndef BOOLEAN_H
#define BOOLEAN_H

#define boolean unsigned char
#define true 1
#define false 0

```



```
#endif
```

### 2.3.2 Persoalan yang Diselesaikan

ADT boolean digunakan untuk menyelesaikan masalah terkait dengan penulisan dan pembacaan logika dalam kode. Dengan mendefinisikan tipe data boolean, serta nilai true dan false, program menjadi lebih mudah dibaca dan dipahami.

### 2.3.3 Alasan Pemilihan

ADT boolean dipilih supaya kode menjadi lebih konsisten, mudah dibaca, dan mudah dipahami.

### 2.3.4 Implementasi

Implementasi boolean terdapat pada kode `boolean.c` pada direktori `src/ADT/boolean.c`

## 2.4 ADT mesinkarakter

### 2.4.1 Sketsa Struktur Data

```
/* File: mesinkarakter.h */
/* Definisi Mesin Karakter */

#ifndef __MESIN_KAR_H_
#define __MESIN_KAR_H_

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

#define MARK '\n'
#define NEWLINE '\n'
/* State Mesin */
extern int retval;
extern char currentChar;
extern boolean EOP;
extern boolean input;

extern FILE *pita;
extern FILE *pitaFile;

void START(char *path, char *var);
/* Mesin siap dioperasikan. Pita disiapkan untuk dibaca.
   Karakter pertama yang ada pada pita posisinya adalah pada jendela.
   Pita baca diambil dari stdin.
   I.S. : sembarang
   F.S. : currentChar adalah karakter pertama pada pita
          Jika currentChar ≠ MARK maka EOP akan padam (false)
          Jika currentChar = MARK maka EOP akan menyala (true) */

void ADV();
/* Pita dimajukan satu karakter.
   I.S. : Karakter pada jendela = currentChar, currentChar ≠ MARK
   F.S. : currentChar adalah karakter berikutnya dari currentChar yang lama,
          currentChar mungkin = MARK
          Jika currentChar = MARK maka EOP akan menyala (true) */
```

```

char GetCC();
/* Mengirimkan currentChar */

boolean IsEOP();
/* Mengirimkan true jika currentChar = MARK */

#endif

```

#### 2.4.2 Persoalan yang Diselesaikan

ADT Mesin Karakter dapat digunakan untuk membaca file konfigurasi, memproses input pengguna, dan menjadi dasar untuk implementasi modul lanjutan seperti mesin kata.

#### 2.4.3 Alasan Pemilihan

ADT Mesin Karakter dipilih karena efisien dalam membaca input satu per satu, memberi kontrol lebih pada pengolahan data teks. Selain itu, pemisahan pembacaan karakter dari bagian lain kode membuat program lebih mudah dibaca dan dikelola, serta memastikan alur pembacaan data yang konsisten.

#### 2.4.4 Implementasi

Implementasi mesinkarakter terdapat pada kode mesinkarakter.c pada direktori src/ADT/mesinkarakter.c

### 2.5 ADT mesinkata

#### 2.5.1 Sketsa Struktur Data

```

#include<stdio.h>
#include "mesinkata.h"

boolean EndWord;
Word CurrentWord;

void Ignore(){
/* Mengabaikan satu atau beberapa NEWLINE
  I.S. : currentChar sembarang
  F.S. : currentChar ≠ NEWLINE atau currentChar = MARK */
  while (GetCC() == NEWLINE || GetCC() == BLANK){
    ADV();
  }
}

void STARTWORD(char *path, char *type){
/* 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 */
  START(path, type);
  if (pitaFile == NULL){
    return;
  } else {
    Ignore();
    if (IsEOP()){
      EndWord = true;
    } else {

```

```

        EndWord = false;
        CopyWord();
    }
}

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 */
    Ignore();
    if (IsEOP()){
        EndWord = true;
    } else {
        CopyWord();
        EndWord = false;
    }
}

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" */
    int i = 0;
    while (!IsEOP() && GetCC() != BLANK && GetCC() != MARK && GetCC() != NEWLINE){
        if (i < NMax)
        {
            CurrentWord.TabWord[i] = GetCC();
            i++;
        }
        ADV();
    }
    CurrentWord.Length = i;
    CurrentWord.TabWord[i] = '\0';
}

void ResetWord(){
    CurrentWord.TabWord[0] = '\0';
    CurrentWord.Length = 0;
}

```

### 2.5.2 Persoalan yang Diselesaikan

ADT mesinkata digunakan untuk mempermudah proses membaca kata demi kata dari suatu file atau input, dalam hal ini dikembangkan sebagai pengganti dari scanf.

### 2.5.3 Alasan Pemilihan

ADT mesinkata dipilih untuk menerima input dan commad dari user.

### 2.5.4 Implementasi

Implementasi mesinkata terdapat pada kode mesinkata.c pada direktori src/ADT/mesinkata.c

## 2.6 ADT queue

### 2.6.1 Sketsa Struktur Data

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

#ifndef QUEUE_H
#define QUEUE_H

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

#define IDX_UNDEF -1
#define CAPACITY 10

typedef struct {
    char name[50];
} nama_barang;

/* Definisi elemen dan address */
typedef struct {
    nama_barang 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 isEmpty(Queue q);
/* Mengirim true jika q kosong: lihat definisi di atas */
boolean isFull(Queue q);
/* Mengirim true jika tabel penampung elemen q sudah penuh */
/* yaitu IDX_TAIL akan selalu di belakang IDX_HEAD dalam buffer melingkar*/

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

```

/* *** Primitif Add/Delete *** */
void enqueue(Queue *q, nama_barang 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 dequeue(Queue *q, nama_barang *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

```

### 2.6.2 Persoalan yang Diselesaikan

ADT queue digunakan untuk menyelesaikan persoalan pengelolaan struktur data antrian (queue) untuk menyimpan dan memproses data barang yang masuk, keluar, dan dikelola secara FIFO (First In First Out).

### 2.6.3 Alasan Pemilihan

ADT queue berbasis circular buffer dipilih agar sesuai dengan kebutuhan pengelolaan data barang yang menggunakan aturan FIFO. Circular buffer dipilih untuk efisiensi memori dan kemudahan implementasi pada kasus dengan kapasitas tetap.

### 2.6.4 Implementasi

Implementasi queue terdapat pada kode `queue.c` pada direktori `src/ADT/queue.c`

## 2.7 ADT misc

### 2.7.1 Sketsa Struktur Data

```

#ifndef __MISC_H_
#define __MISC_H_

#include <stdlib.h>
#include "src/Header/load.h"
// #include "../ADT/array/array.h"
// #include "../ADT/arraydin/arraydin.h"
// #include "../ADT/queue/queue.h"
// #include "../ADT/boolean/boolean.h"
// #include "../ADT/mesinkarakter/mesinkarakter.h"
// #include "../ADT/mesinkata/mesinkata.h"
#include "src/ADT/array/array.h"
#include "src/ADT/arraydin/arraydin.h"
#include "src/ADT/queue/queue.h"
#include "src/ADT/boolean/boolean.h"

```

```

#include "src/ADT/mesinkarakter/mesinkarakter.h"
#include "src/ADT/mesinkata/mesinkata.h"

// Mengembalikan index sebuah barang pada toko menggunakan traversal
// menerima parameter array dinamis dan string
IdxType IndexItemInShop(ArrayDinBarang array, char *str);

// Mengembalikan index sebuah user pada konfigurasi file menggunakan traversal
// menerima parameter Tabuser dan string
IdxType IndexUserInFile(TabUser Users, char *str);

// Mengembalikan index sebuah job pada konfigurasi file menggunakan traversal
// menerima parameter job, total job dan string
IdxType IndexJobInList(job listjob[], int totaljob, char *str);

// Mengkonfirmasi apakah proses dalam store supply dan store remove di terminasi
// menerima parameter string
boolean isDone(char *input);

// Mencari nama barang pada toko menggunakan traversal
// menerima parameter array dinamis dan string
boolean isItemInShop(ArrayDinBarang array, char *str);

// Mencari nama barang pada antrian dengan traversal
// menerima parameter queue dan string
boolean isItemInQueue(Queue items_request, char *str);

// Mencari nama user pada file dengan traversal
// menerima parameter Tabuser dan string
boolean isUserInFile(TabUser Users, char *str);

// Memeriksa apakah input semuanya merupakan digit
// menerima parameter string
boolean isStrAllDigit (char *str);

// Memeriksa antara dua string apakah sama
// menerima parameter word dari mesin kata dan string
boolean isStrEqual(char *str1, char *str2);

// Mengembalikan panjang string
// menerima parameter string
int strLength(char *str);

// Fungsi menyalin isi dari string sumber ke string tujuan
// menerima parameter tujuan dan sumber
void copystrng(char *dest, const char *src);

// Validasi input agar tidak melebihi batas 49 Karakter menggunakan mesin kata
// menerima parameter buffer dan panjang maksimal
boolean readInput(char *buffer, int maxLen);

// Validasi command agar tidak melebihi batas 49 Karakter menggunakan mesin kata
// menerima parameter buffer dan panjang maksimal serta nama command
boolean readCommand(char *buffer, int maxLen);

```

```
// Memeriksa apakah input mempunyai spasi
boolean containsSpace(const char *str);

// Menerima input untuk dirubah ke integer
int strToInteger ();

// visual
void visual(char* path);

#endif
```

### 2.7.2 Persoalan yang Diselesaikan

ADT Misc digunakan untuk mencari data, memeriksa input pengguna, dan mengubah tipe data. Selain itu, misc juga berisi fungsi-fungsi yang mendukung program utama.

### 2.7.3 Alasan Pemilihan

ADT Misc dipilih karena menyediakan fungsi-fungsi penting seperti pencarian elemen, validasi input, dan konversi data yang mempermudah pengelolaan aplikasi. Dengan memisahkan fungsi-fungsi pendukung ini, kode menjadi lebih terstruktur, mudah dibaca, dan lebih mudah dikelola.

### 2.7.4 Implementasi

Implementasi misc terdapat pada kode `misc.c` pada direktori `misc.c`

## 3 Program Utama

Program utama terletak pada file `main.c` yang dapat dijalankan dengan menggunakan MakeFile `'mingw32-make run'`. Dalam program `main.c` ini dimulai dengan pendefinisian header yang diperlukan seperti `<stdlib.h>` dan header dari `main.h`. Dalam header `main.h` terdapat 18 header, yaitu ada `<stdlib.h>`, `help.h`, `misc.h`, `store_rmv.h`, `store_list.h`, `store_sup.h`, `store_req.h`, `save.h`, `quit.h`, `load.h`, `start.h`, `work.h`, `work_challenge.h`, `wordl399.h`, `tebakangka.h`, `login.h`, `register.h`, dan `logout.h`.

Program utama ini mengimplementasikan sistem simulasi e-commerce yang menggunakan struktur data seperti array, array dinamis, array static, queue, dan ADT lainnya untuk mengelola data pengguna dan barang. Program dimulai dengan inisialisasi struktur data, pengguna dapat memasukkan perintah seperti START, LOAD, HELP, dan QUIT setelah dilaksanakannya run `main.c`. Setelah itu jika kita mengetik START maka akan keluar tampilan “starting program” dan akan menampilkan nama user dan jumlah uang yang dimiliki user dalam loaded users, dan juga akan menampilkan loaded items beserta harga barangnya. Jika memasukkan command LOAD dan jika user memasukkan nama file yang valid maka program akan menampilkan loaded users dan loaded items, sama seperti ketika kita memasukkan command START pada program. Selain itu ada juga command HELP yang dapat dimasukkan. HELP ini akan menampilkan ‘Welcome Menu’, ‘Login Menu’, dan ‘Menu Help’. Jika command yang dimasukkan adalah command QUIT maka user akan keluar dari program jika memilih ‘Y’ dan akan tetap dalam program jika memilih ‘N’.

Secara umum, output yang dihasilkan merupakan informasi yang terkait dengan input command yang dimasukkan serta data yang tersedia seperti ITEMS dan informasi mengenai

USERS serta HARGA BARANG. Penggunaan fitur LOGIN dan REGISTER dapat digunakan user untuk menyimpan informasi seperti nama user. Users juga dapat melakukan WORK untuk menambah saldo dan menyelesaikan WORK CHALLENGE. Dalam program utama ini juga ada program untuk manajemen barang seperti STORE REQUEST, STORE SUPPLY, STORE LIST, dan STORE REMOVE. Program utama ini juga memungkinkan user untuk melakukan LOGOUT dimana dilakukannya validasi status login user, setelah users yang sudah login tervalidasi maka sesi user bisa diakhiri sehingga program akan menampilkan pesan LOGOUT.

Program memvalidasi setiap input untuk memastikan pengguna hanya mengakses fitur yang sesuai dengan status login users. Program akan terus berjalan hingga command QUIT diberikan, di mana data yang diubah dapat disimpan di SAVE.

## **4 Algoritma-Algoritma Menarik**

### **4.1 Implementasi Lain *scanf()* pada Bahasa C**

Merubah penggunaan fungsi *scanf()* dalam bahasa C dengan implementasi manual yang menggunakan kombinasi fungsi seperti *readInput()*, *isStrAllDigit()*, dan *isStrEqual()* membantu proses pembacaan input. Implementasi ini memanfaatkan ADT mesin karakter untuk membaca karakter satu per satu dan kemudian memprosesnya menggunakan algoritma berbasis string, seperti memeriksa apakah input berupa angka, membandingkan string, atau membaca input hingga panjang tertentu. Ini menarik karena memberikan fleksibilitas untuk memvalidasi dan mengelola input tanpa bergantung pada format bawaan *scanf()*, yang sering kali kurang fleksibel. Pendekatan ini serupa dengan cara kerja *scanf()*, namun memberikan kemampuan tambahan untuk menangani dan memvalidasi input secara lebih spesifik.



## 5.1 *START*

Data yang di test : start.c

[illegible]

Hasil yang seharusnya:

```
>>START
STARTING PROGRAM . . .
Loaded Users:
User 1: ligma (Money: 1000)
User 2: praktikan (Money: 25)
User 3: bagas (Money: 100)
User 4: tes (Money: 200)
Loaded Items:
Item 1: a (Price: 10)
Item 2: b (Price: 20)
Item 3: c d e (Price: 20)
Item 4: Doofincshzmert (Price: 4100)
```

## 5.2 LOAD

Data yang di test : load.c

```
>>LOAD
LOAD PROGRAM
Masukkan nama file yang akan di load :
aulia.txt
Tidak ada file dengan nama tersebut!
```

```
Tidak ada file dengan nama tersebut!
Masukkan nama file yang akan di load :
default.txt
Save file berhasil dibaca. PURRMART berhasil dijalankan.
LOADING PROGRAM . . .
Loaded Users:
User 1: ligma (Money: 1000)
User 2: praktikan (Money: 25)
User 3: bagas (Money: 100)
User 4: tes (Money: 200)

Loaded Items:
Item 1: a (Price: 10)
Item 2: b (Price: 20)
Item 3: c d e (Price: 20)
Item 4: Doofincshzmert (Price: 4100)
```

Hasil yang seharusnya:

```
>>LOAD
LOAD PROGRAM
Masukkan nama file yang akan di load :
// MeLoad /save/aulia.txt
aulia.txt
Tidak ada file dengan nama tersebut!
```

```
>>LOAD
LOAD PROGRAM
Masukkan nama file yang akan di load :
// MeLoad /save/default.txt
default.txt
Save file berhasil dibaca. PURRMART berhasil dijalankan.
LOADING PROGRAM . . .
Loaded Users:
User 1: ligma (Money: 1000)
User 2: praktikan (Money: 25)
User 3: bagas (Money: 100)
User 4: tes (Money: 200)

Loaded Items:
Item 1: a (Price: 10)
```

Item 2: b (Price: 20)  
Item 3: c d e (Price: 20)  
Item 4: Doofincshzmert (Price: 4100)

### 5.3 LOGIN

Data yang di test : login.c

```
>>LOGIN
Masukkan username: ligma
Masukkan password: alstrukdatkeren
Anda telah login ke PURRMART sebagai ligma.
```

```
>>LOGIN
Masukkan username: ligma
Masukkan password: alstrukdat
Username atau password salah. Coba lagi.
```

```
>>LOGIN
Masukkan username: Aulia
Masukkan password: 12345
Username atau password salah. Coba lagi.
```

```
>>LOGIN
Masukkan username: Aulia
Masukkan password: 12345
Username atau password salah. Coba lagi.
Masukkan username: Aulia
Masukkan password: 123456
Username atau password salah. Coba lagi.
Masukkan username: Aulia
Masukkan password: 123457
Username atau password salah. Coba lagi.
Sudah terlalu banyak percobaan, anda akan dikembalikan ke laman utama!
```

```
LOGIN
Anda masih berada di sesi ligma, silahkan logout terlebih dahulu!
```

Hasil yang seharusnya:

// Contoh login yang berhasil untuk username ligma dan password Alstrukdatkeren

```
>>LOGIN
Masukkan username: ligma
Masukkan password: alstrukdatkeren
Anda telah login ke PURRMART sebagai ligma.
```

// Contoh login yang gagal karena password salah

```
>>LOGIN
Masukkan username: ligma
Masukkan password: alstrukdat
Username atau password salah. Coba lagi.
```

// Contoh login yang gagal karena belum ada pengguna dengan username Aulia

>>LOGIN

Masukkan username: Aulia

Masukkan password: 12345

Username atau password salah. Coba lagi.

// Contoh login yang gagal karena terlalu banyak percobaan

>>LOGIN

Masukkan username: Aulia

Masukkan password: 12345

Username atau password salah. Coba lagi.

Masukkan username: Aulia

Masukkan password: 123456

Username atau password salah. Coba lagi.

Masukkan username: Aulia

Masukkan password: 1234567

Username atau password salah. Coba lagi.

Sudah terlalu banyak percobaan, anda akan dikembalikan ke laman utama!

// Contoh login yang gagal karena pengguna belum LOGOUT

>>LOGIN

Anda masih berada di sesi login, silahkan logout terlebih dahulu!

## 5.4 LOGOUT

Data yang di test : logout.c

>>LOGOUT

ligma telah logout dari sistem PURRMART. Silakan REGISTER/LOGIN kembali untuk melanjutkan.

Hasil yang seharusnya:

>>LOGOUT

ligma telah logout dari sistem PURRMART. Silakan REGISTER/LOGIN kembali untuk melanjutkan.

## 5.5 REGISTER

Data yang di test : register.c

>>REGISTER

Masukkan username (maksimum 49 karakter, tanpa spasi): ivanaazizah

Masukkan password (maksimum 20 karakter): hahaha

Masukkan jumlah uang awal: 100

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

```
>>REGISTER
```

```
Masukkan username (maksimum 49 karakter, tanpa spasi): bagas  
Akun dengan username bagas gagal dibuat karena sudah ada. Silakan lakukan REGISTER ulang.
```

```
>>REGISTER
```

```
Masukkan username (maksimum 49 karakter, tanpa spasi): ivana azizah  
Username tidak boleh mengandung spasi.
```

```
Masukkan username (maksimum 49 karakter, tanpa spasi): maria
```

```
Masukkan password (maksimum 20 karakter): uhuy
```

```
Password tidak valid! Panjang password harus 6-20 karakter.
```

```
>>REGISTER
```

```
Masukkan username (maksimum 49 karakter, tanpa spasi): maria
```

```
Masukkan password (maksimum 20 karakter): uhuybanget
```

```
Masukkan jumlah uang awal: aaa
```

```
Input jumlah uang tidak valid. Masukkan angka positif.
```

```
Masukkan jumlah uang awal: -50
```

```
Input jumlah uang tidak valid. Masukkan angka positif.
```

Hasil yang seharusnya:

```
// Contoh register yang berhasil
```

```
>>REGISTER
```

```
Masukkan username (maksimum 49 karakter, tanpa spasi): ivanaazizah
```

```
Masukkan password (maksimum 20 karakter): hahaha
```

```
Masukkan jumlah uang awal: 100
```

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

```
// Contoh register yang gagal karena username sudah ada
```

```
>>REGISTER
```

```
Masukkan username (maksimum 49 karakter, tanpa spasi): bagas
```

```
Akun dengan username bagas gagal dibuat karena sudah ada. Silakan lakukan REGISTER  
ulang.
```

```
// Contoh register yang gagal karena username mengandung spasi
```

```
>>REGISTER
```

```
Masukkan username (maksimum 49 karakter, tanpa spasi): ivana azizah
```

```
Username tidak boleh mengandung spasi.
```

```
//Contoh register yang gagal karena password tidak sesuai
```

```
>>REGISTER
```

```
Masukkan username (maksimum 49 karakter, tanpa spasi): maria
```

```
Masukkan password (maksimum 20 karakter): uhuy
```

```
Password tidak valid! Panjang password harus 6-20 karakter.
```

```
//Contoh register yang gagal karena uang bukan interger dan bernilai negatif
```

```
>>REGISTER
```

Masukkan username (maksimum 49 karakter, tanpa spasi): maria  
Masukkan password (maksimum 20 karakter): uhuybanget  
Masukkan jumlah uang awal: aaa  
Input jumlah uang tidak valid. Masukkan angka positif.  
Masukkan jumlah uang awal: -50  
Input jumlah uang tidak valid. Masukkan angka positif.

## 5.6 WORK

Data yang di test : work.c

```
>>WORK
Saldo sekarang : Rp 1000
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 nama pekerjaan yang dipilih: Evil Lab
Pekerjaan tidak ditemukan. Silakan coba lagi.
Masukkan nama pekerjaan yang dipilih: Evil Lab Assistant
Anda sedang bekerja sebagai Evil Lab Assistant... harap tunggu.
Pekerjaan selesai, +100 rupiah telah ditambahkan ke akun Anda.
Saldo sekarang setelah bekerja : Rp 1100
```

Hasil yang seharusnya:

```
>>WORK
Saldo sekarang : Rp 1000
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 nama pekerjaan yang dipilih: Evil Lab
Pekerjaan tidak ditemukan. Silakan coba lagi.

Masukkan nama pekerjaan yang dipilih: Evil Lab Assistant
Anda sedang bekerja sebagai Evil Lab Assistant... harap tunggu.
Pekerjaan selesai, +100 rupiah telah ditambahkan ke akun Anda.
Saldo sekarang setelah bekerja : Rp 1100
```

## 5.7 WORK CHALLENGE

Data yang di test : tebakangka.c

```
>> WORK CHALLENGE

Untuk keluar ketik (Purry)
Daftar challenge yang tersedia:
1. Tebak Angka (biaya main=200)
2. WORDL3 (biaya main=500)

Masukan challenge yang hendak dimainkan (1/2): 1
Mulai challenge Tebak Angka!
30
Tebakanmu lebih besar!
40
Tebakanmu lebih besar!
23
Tebakanmu lebih besar!
7
Tebakanmu lebih kecil!
10
Tebakanmu lebih kecil!
19
Tebakanmu lebih besar!
15
Tebakanmu lebih besar!
14
Tebakanmu benar! +150 rupiah telah ditambahkan ke akun anda.
```

Hasil yang seharusnya:

```
>> WORK CHALLENGE

Untuk keluar ketik (Purry)
Daftar challenge yang tersedia:
1. Tebak Angka (biaya main=200)
2. WORDL3 (biaya main=500)

Masukan challenge yang hendak dimainkan (1/2): 1
Mulai challenge Tebak Angka!
30
Tebakanmu lebih besar!
40
Tebakanmu lebih besar!
23
Tebakanmu lebih besar!
7
Tebakanmu lebih kecil!
10
Tebakanmu lebih kecil!
19
Tebakanmu lebih besar!
15
Tebakanmu lebih besar!
14
Tebakanmu benar! +150 rupiah telah ditambahkan ke akun anda.
```

Data yang di test : wordl399.c

```
>> WORK CHALLENGE
```

```
Untuk keluar ketik (Purry)
Daftar challenge yang tersedia:
1. Tebak Angka (biaya main=200)
2. W0RDL3 (biaya main=500)
```

```
Masukan challenge yang hendak dimainkan (1/2): 2
```

```
WELCOME TO W0RDL3! YOU HAVE 5 CHANCES TO GUESS THE WORD!
```

```
- - - - -
- - - - -
- - - - -
- - - - -
- - - - -
```

```
Masukkan kata tebakan Anda (input dalam lowercase): aiueo
Hasil:
a*i%u%e%o%
```

```
Masukkan kata tebakan Anda (input dalam lowercase): galon
Hasil:
g%a1%o%n%
```

```
Masukkan kata tebakan Anda (input dalam lowercase): katak
Hasil:
katak
```

```
Selamat, Anda menang!
+1500 rupiah telah ditambahkan ke akun Anda.
Apakah kamu ingin bekerja kembali? (YA/Purry)
```

```
>> WORK CHALLENGE
```

```
Untuk keluar ketik (Purry)
Daftar challenge yang tersedia:
1. Tebak Angka (biaya main=200)
2. W0RDL3 (biaya main=500)
```

```
Masukan challenge yang hendak dimainkan (1/2): 2
Saldo tidak mencukupi untuk memainkan challenge ini.
```

Hasil yang seharusnya:

```
//Contoh Work Challenge jika uang cukup
```

```
>> WORK CHALLENGE
```

```
Untuk keluar ketik (Purry)
Daftar challenge yang tersedia:
1. Tebak Angka (biaya main=200)
2. W0RDL3 (biaya main=500)
```

```
Masukan challenge yang hendak dimainkan (1/2): 2
```



WELCOME TO W0RDL3! YOU HAVE 5 CHANCES TO GUESS THE WORD!

- - - - -  
- - - - -  
- - - - -  
- - - - -  
- - - - -

Masukkan kata tebakan Anda (input dalam lowercase): aiueo

Hasil:

a\*i%u%e%o%

Masukkan kata tebakan Anda (input dalam lowercase): galon

Hasil:

g%aL%o%n%

Masukkan kata tebakan Anda (input dalam lowercase): katak

Hasil:

katak

Selamat, Anda menang!

+1500 rupiah telah ditambahkan ke akun Anda.

//Contoh Work Challenge jika uang tidak cukup

>> WORK CHALLENGE

Untuk keluar ketik (Purry)

Daftar challenge yang tersedia:

1. Tebak Angka (biaya main=200)
2. W0RDL3 (biaya main=500)

Masukan challenge yang hendak dimainkan (1/2): 2

Saldo tidak mencukupi untuk memainkan challenge ini.

## 5.8 STORE LIST

Data yang di test : store\_list.c

```
>>STORE LIST
```

```
List barang yang ada di toko:
```

```
- a  
- b  
- c d e  
- Doofincshzmert
```

```
STORE LIST
```

```
>>STORE LIST
```

```
TOKO KOSONG
```

Hasil yang seharusnya:

>>STORE LIST

List barang yang ada di toko:

- a
- b
- c d e
- Doofincshzmert

>>STORE LIST

TOKO KOSONG

## **5.9 STORE REQUEST**

Data yang di test : store\_req.c

## STORE REQUEST

/ \_ ) ( \_ ) ( \_ ) ( \_ \ ( \_ ) ( \_ \ ( \_ ) ( \_ ) ( ) ( ) ( \_ ) / \_ ) ( \_ )  
 \ \ ) ( ) ( ) ( ) / \_ ) ( ) / \_ ) ( ) ( ) ( ) ( \_ ) \ \ ) ( )  
 ( \_ / ( \_ ) ( \_ ) ( ) \ ( \_ ) ( ) \ ( \_ ) ( \_ / \ ( \_ ) ( \_ ) ( \_ / ( \_ )

Write "Purry" to exit

```
>>STORE REQUEST
===== [STORE REQUEST] =====
Nama barang yang diminta:
bom
Barang berhasil dimasukkan ke dalam antrian
Nama barang yang diminta:
bom
Barang sudah berada di dalam antrian!
Nama barang yang diminta:
a
Barang sudah ada di toko!
```

Hasil yang seharusnya:

Write "Purry" to exit

## >>STORE REQUEST

```
===== [STORE REQUEST] =====
```

Nama barang yang diminta:

bom

Barang berhasil dimasukkan ke dalam antrian

Nama barang yang diminta:

bom

Barang sudah berada di dalam antrian!

Nama barang yang diminta:

a

Barang sudah ada di toko!

## 5.10 STORE SUPPLY

Data yang di test : store\_sup.c

```

/___)(___)/ \(_ \(_) /___)/ )(\(_ \(_) ( \_ ) ( \_ )
\___ \ )( ( o )) / ) \___ \) \ ( ) _/ ) _/ ( \_ ) /
(____/ ( _ ) \_/ ( _ \ ) (____) (____/ \____/ ( _ ) ( _ ) \____/ ( _/

```

-----

Write "Purrry" to exit

>>STORE SUPPLY

===== [STORE SUPPLY] =====

Apakah kamu ingin menambahkan barang bom? (Terima/Tolak/Tunda)

Terima

Harga Barang :

150

bom dengan harga 150 telah ditambahkan ke toko.

Uangmu sekarang : Rp 850

Apakah kamu ingin menambahkan barang paku? (Terima/Tolak/Tunda)

Tunda

paku dikembalikan ke antrian.

Apakah kamu ingin menambahkan barang paku? (Terima/Tolak/Tunda)

Tolak

paku dibuang dari antrian.

Tidak ada barang di antrian

```
>>>STORE SUPPLY
===== [STORE SUPPLY] =====
Apakah kamu ingin menambahkan barang bom? (Terima/Tolak/Tunda)
Terima
Harga Barang :
2000
Uang kamu tidak mencukupi!
Apakah kamu ingin menambahkan barang bom? (Terima/Tolak/Tunda)
Terima
Harga Barang :
aaaa
Masukkan sebuah integer yang lebih besar dari 0!
```

Apakah kamu ingin menambahkan barang paku? (Terima/Tolak/Tunda)  
Purry  
Kamu keluar dari Store Supply!

Hasil yang seharusnya:

```

Write "Purry" to exit
>>STORE SUPPLY
===== [STORE SUPPLY] =====
Apakah kamu ingin menambahkan barang bom? (Terima/Tolak/Tunda)
Terima
Harga Barang :
150

bom dengan harga 150 telah ditambahkan ke toko.
Uangmu sekarang : Rp 850

```

Apakah kamu ingin menambahkan barang paku? (Terima/Tolak/Tunda)  
Tunda  
paku dikembalikan ke antrian.

Apakah kamu ingin menambahkan barang paku? (Terima/Tolak/Tunda)  
Tolak  
paku dibuang dari antrian.  
Tidak ada barang di antrian

>>STORE SUPPLY  
===== [STORE SUPPLY] =====  
Apakah kamu ingin menambahkan barang bom? (Terima/Tolak/Tunda)  
Terima  
Harga Barang :  
2000  
Uang kamu tidak mencukupi!  
  
Apakah kamu ingin menambahkan barang bom? (Terima/Tolak/Tunda)  
Terima  
Harga Barang :  
aaaa  
Masukkan sebuah integer yang lebih besar dari 0!

>>STORE SUPPLY  
===== [STORE SUPPLY] =====  
Apakah kamu ingin menambahkan barang paku? (Terima/Tolak/Tunda)  
Purry  
Kamu keluar dari Store Supply!

### 5.11 STORE REMOVE

Data yang di test : store\_rmv.c

```
( _ ) ( _ ) ( _ ) ( - \ ( _ )   ( _ \ ( _ ) ( V ) ( _ ) ( V ) ( _ )  
 \ \ ) ( ) ( ) / )      ) / )    ) ( ) ( \ / )  
( / ( ) ( _ ) ( ) \ ( _ )   ( ) \ ( _ ) ( \ \ ) ( _ ) V ( _ )
```

```
-----  
  
Write "Purrry" to exit  
  
>>STORE REMOVE  
===== [STORE REMOVE] =====  
Items in Store:  
Item 1: a (Price: 10)  
Item 2: b (Price: 20)  
Item 3: c d e (Price: 20)  
Item 4: Doofincshzmert (Price: 4100)  
Item 5: bom (Price: 77)  
Nama barang yang dihapus:  
bom  
Barang berhasil dihilangkan.  
Items in Store:  
Item 1: a (Price: 10)  
Item 2: b (Price: 20)  
Item 3: c d e (Price: 20)  
Item 4: Doofincshzmert (Price: 4100)
```

```
Nama barang yang dihapus:  
martabak  
Barang tidak ada di toko!  
Items in Store:  
Item 1: a (Price: 10)  
Item 2: b (Price: 20)  
Item 3: c d e (Price: 20)  
Item 4: Doofincshzmert (Price: 4100)
```

Hasil yang seharusnya:

```
Write "Purrry" to exit
//Contoh Store Remove jika barang ada
>>STORE REMOVE
===== [STORE REMOVE] =====
Items in Store:
Item 1: a (Price: 10)
Item 2: b (Price: 20)
Item 3: c d e (Price: 20)
Item 4: Doofincshzmert (Price: 4100)
Item 5: bom (Price: 77)
Nama barang yang dihapus:
bom
Barang berhasil dihilangkan.
Items in Store:
Item 1: a (Price: 10)
Item 2: b (Price: 20)
Item 3: c d e (Price: 20)
Item 4: Doofincshzmert (Price: 4100)

//Contoh Store Remove jika barang tidak ada
>>STORE REMOVE
```

```

===== [STORE REMOVE] =====
Nama barang yang dihapus:
martabak
Barang tidak ada di toko!
Items in Store:
Item 1: a (Price: 10)
Item 2: b (Price: 20)
Item 3: c d e (Price: 20)
Item 4: Doofincshzmert (Price: 4100)

```

## 5.12 HELP

Data yang di test : help.c

```

HELP
====[ Welcome Menu Help PURRMART ]====
1. START -> Untuk masuk sesi baru
2. LOAD -> Untuk memulai sesi berdasarkan file konfigurasi
3. QUIT -> Untuk keluar dari program

```

```

HELP
====[ Login Menu Help PURRMART ]====
1. REGISTER -> Untuk melakukan pendaftaran akun baru
2. LOGIN -> Untuk masuk ke dalam akun dan memulai sesi
3. QUIT -> Untuk keluar dari program

```

```

HELP
====[ Menu Help PURRMART ]====
1. WORK -> Untuk bekerja
2. WORK CHALLENGE -> Untuk mengerjakan challenge
3. STORE LIST -> Untuk melihat barang-barang di toko
4. STORE REQUEST -> Untuk meminta penambahan barang
5. STORE SUPPLY -> Untuk menambahkan barang dari permintaan
6. STORE REMOVE -> Untuk menghapus barang
7. LOGOUT -> Untuk keluar dari sesi
8. SAVE -> Untuk menyimpan state ke dalam file
9. QUIT -> Untuk keluar dari program

```

Hasil yang seharusnya:

```

// Ketika perintah dipanggil pada welcome menu
HELP
====[ Welcome Menu Help PURRMART ]====
1. START → Untuk masuk sesi baru
2. LOAD → Untuk memulai sesi berdasarkan file konfigurasi
3. QUIT → Untuk keluar dari program

```

```

// Ketika perintah dipanggil pada login menu
HELP
====[ Login Menu Help PURRMART ]====
1. REGISTER → Untuk melakukan pendaftaran akun baru
2. LOGIN → Untuk masuk ke dalam akun dan memulai sesi
3. QUIT → Untuk keluar dari program

```

```

// Ketika perintah dipanggil pada main menu

```

## HELP

=====[ Menu Help PURRMART ]=====

1. WORK → Untuk bekerja
2. WORK CHALLENGE → Untuk mengerjakan challenge
3. STORE LIST → Untuk melihat barang-barang di toko
4. STORE REQUEST → Untuk meminta penambahan barang
5. STORE SUPPLY → Untuk menambahkan barang dari permintaan
6. STORE REMOVE → Untuk menghapus barang
7. LOGOUT → Untuk keluar dari sesi
8. SAVE → Untuk menyimpan state ke dalam file
9. QUIT → Untuk keluar dari program

### 5.13 SAVE

Data yang di test : save.c

```
SAVE
Masukkan nama file untuk SAVE state program:
default.txt
save/default.txt
Save file berhasil disimpan
```

Hasil yang seharusnya:

```
>> SAVE
Masukkan nama file untuk SAVE state program:
default.txt
save/default.txt
Save file berhasil disimpan
// File disimpan pada /save/savefile.txt
```

### 5.14 QUIT

Data yang di test : quit.c

```
QUIT

Apakah kamu ingin menyimpan data sesi sekarang (Y/N)? N

Kamu keluar dari PURRMART.
Dadah ^_^/
```

QUIT

Apakah kamu ingin menyimpan data sesi sekarang (Y/N)? Y

Masukkan nama file untuk SAVE state program:

default.txt

save/default.txt

Save file berhasil disimpan

Kamu keluar dari PURRMART.

Dadah ^\_^/

Hasil yang seharusnya:

>> QUIT

Apakah kamu ingin menyimpan data sesi sekarang (Y/N)? N

Kamu keluar dari PURRMART.

Dadah ^\_^/

>> QUIT

Apakah kamu ingin menyimpan data sesi sekarang (Y/N)? Y

Masukkan nama file untuk SAVE state program:

default.txt

save/default.txt

Save file berhasil disimpan

Kamu keluar dari PURRMART.

Dadah ^\_^/

## 6 Test Script

No.	Fitur yang Dites	Tujuan Testing	Langkah-Langkah Testing	Input Data Test	Hasil yang Diharapkan	Hasil yang Keluar
1	F00-Start	Memastikan bahwa program PURRMART dapat dijalankan	Memasukkan <i>command START</i>	Data Test 6.1	PURRMART berhasil dimulai dan akan menampilkan tampilan awal yang dapat dilihat pada data test 6.1	sesuai yang diharapkan
2	F01-Load	Memastikan bahwa file konfigurasi dapat dibaca dan disimpan	Memasukkan <i>command LOAD</i> dilanjutkan dengan nama file yang sudah di <i>save</i>	Data Test 6.2	Program berhasil membaca file yang sudah di <i>save</i> dan menampilkan "Save file berhasil dibaca. PURRMART berhasil	sesuai yang diharapkan



					dijalankan” dan akan memuat user dan item yang tersimpan	
3	F02-Login	Memastikan pengguna dapat masuk ke akun yang sudah didaftarkan	Memasukkan <i>command LOGIN</i> dilanjutkan dengan memasukkan <i>username</i> dan <i>password</i> pengguna yang sudah didaftarkan sebelumnya	Data Test 6.3	Program berhasil memasukkan pengguna ke akun yang diinput saat melakukan <i>login</i> , tidak berhasil jika <i>username</i> atau <i>password</i> salah, telah <i>login</i> atau belum melakukan <i>logout</i> , terlalu banyak melakukan percobaan	sesuai yang diharapkan
4	F03-Logout	Memastikan pengguna dapat keluar dari akun	Memasukkan <i>command LOGOUT</i>	Data Test 6.4	Program berhasil mengeluarkan pengguna dari sistem PURRMART	sesuai yang diharapkan
5	F04-Register	Memastikan pengguna dapat melakukan pendaftaran akun baru dengan input yang valid	Memasukkan <i>command REGISTER</i> dilanjutkan dengan memasukkan <i>username</i> , <i>password</i> dan uang sesuai dengan ketentuan	Data Test 6.5	Program berhasil mendaftarkan akun baru yang dimasukkan oleh pengguna	sesuai yang diharapkan
6	F05-Work	Memastikan apakah <i>command WORK</i> dapat berjalan dengan baik	Memasukkan <i>command WORK</i> dilanjutkan dengan memasukkan nama pekerjaan yang dipilih	Data Test 6.6	Program berjalan dengan baik sehingga pengguna mendapatkan uang dari pekerjaan yang dipilih	sesuai yang diharapkan
7	F06-Work Challenge	Memastikan <i>command WORK CHALLENGE</i> dapat menjalankan <i>challenge</i> tebak angka dan word13	Memasukkan <i>command WORK CHALLENGE</i> dan memilih salah satu <i>challenge</i> yang tersedia. Jika pengguna memilih opsi 1, program tebak angka dimulai dan pengguna akan diminta untuk menginput angka hingga benar. Sedangkan pengguna memilih	Data Test 6.7	Program berjalan dengan baik pada semua fitur di <i>WORK CHALLENGE</i> . Program tebak angka dan word1399 berjalan dengan baik sehingga ketika tebakan angka atau huruf pengguna benar, uang akan bertambah secara otomatis ke akun pengguna	sesuai yang diharapkan

			opsi 2, program wordl3 dimulai dan pengguna diminta memasukkan 5 huruf hingga benar			
8	F07-Store List	Memastikan apakah <i>command STORE LIST</i> dapat menampilkan barang - barang yang ada di dalam toko	Memasukkan <i>command STORE LIST</i>	Data Test 6.8	Program berhasil menampilkan barang - barang yang ada di toko	sesuai yang diharapkan
8	F08-Store Request	Memastikan apakah <i>command STORE REQUEST</i> dapat melakukan permintaan penambahan barang baru ke dalam toko	Memasukkan <i>command STORE REQUEST</i> dilanjutkan dengan memasukkan nama barang yang diminta	Data Test 6.9	Program berhasil menyimpan barang yang diminta ke dalam antrian serta memastikan bahwa barang tersebut belum tersedia di toko maupun di antrian.	sesuai yang diharapkan
9	F09-Store Supply	Memastikan apakah <i>command STORE SUPPLY</i> dapat menambahkan barang dari antrian ke dalam toko, serta memungkinkan untuk memasukkan kembali atau menghapus barang dari antrian	Memasukkan <i>command STORE SUPPLY</i> dilanjutkan dengan memilih untuk menerima, menunda atau menolak barang tersebut	Data Test 6.10	Program berhasil menambahkan barang baru ke dalam toko serta menentukan harga barang. Selain itu, program juga dapat menghapus barang yang telah ditambahkan dan memasukkannya kembali	sesuai yang diharapkan
10	F10-Store Remove	Memastikan apakah <i>command STORE REMOVE</i> dapat menghapus barang pada toko	Memasukkan <i>command STORE REMOVE</i> dilanjutkan dengan memasukkan nama barang yang ingin dihapus	Data Test 6.11	Program berhasil menghapus barang dari toko dan menampilkan pesan bahwa barang tersebut telah berhasil dihapus	sesuai yang diharapkan
11	F11-Help	Mengetahui <i>command</i> -	Memasukkan <i>command HELP.</i>	Data Test 6.12	Program berhasil menampilkan	sesuai yang diharapkan

		command yang dapat diakses pada program tercetak pada layar	Untuk dapat mengakses menu utama, pengguna harus terlebih dahulu login melalui menu login		<i>command</i> - <i>command</i> yang dapat diakses pada layar	
12	F12-Save	Memastikan command SAVE dapat menyimpan keadaan terbaru aplikasi ke dalam file tertentu	Memasukkan command SAVE dan menuliskan nama file yang akan disimpan	Data Test 6.13	Program berhasil menyimpan keadaan terbaru aplikasi ke dalam suatu file	sesuai yang diharapkan
13	F13-Quit	Memastikan command QUIT dapat mengeluarkan pengguna dari program	Memasukkan command QUIT	Data Test 6.14	Program berhasil mengeluarkan pengguna dari program yang sedang berlangsung	sesuai yang diharapkan

## 7 Pembagian Kerja dalam Kelompok

Nama Lengkap - NIM	Deskripsi Tugas
Bagas Noor Fadhilah - 18223115	<ul style="list-style-type: none"> <li>- Membuat code start, store request, store supply, store remove, main</li> <li>- Melakukan debug untuk semua code</li> </ul>
Khairunnisa Azizah - 18223117	<ul style="list-style-type: none"> <li>- Membuat code store list, work, help, wordl3, work challenge</li> <li>- Membuat laporan</li> </ul>
Maria Vransiska P. C. T. D. P - 18223119	<ul style="list-style-type: none"> <li>- Membuat code tebak angka dan quit</li> <li>- Membuat laporan</li> </ul>
Theresia Ivana M. S - 18223126	<ul style="list-style-type: none"> <li>- Membuat code load dan save</li> <li>- Melakukan notulensi asistensi</li> <li>- Membuat laporan</li> </ul>
Aulia Azka Azzahra - 18223131	<ul style="list-style-type: none"> <li>- Membuat code login, logout, register</li> <li>- Membuat laporan</li> </ul>

## 8 Lampiran

### 8.1 Deskripsi Tugas Besar

Tugas besar ini bertujuan untuk membuat aplikasi simulasi berbasis CLI (*command-line interface*) menggunakan bahasa C dan berbagai ADT, yaitu ADT Kustom, ADT List, ADT Mesin Karakter, ADT Mesin Kata, dan ADT Queue, serta beberapa library seperti `stdio.h`, `stdlib.h`, `time.h`, dan `math.h`.

PURRMART adalah aplikasi yang mensimulasikan aktivitas beli barang pada *e-commerce* dengan fitur utama seperti menampilkan barang toko, meminta dan menyuplai barang baru, menyimpan dan membeli barang dalam keranjang, menampilkan barang yang sudah dibeli, membuat dan menghapus wishlist, serta bekerja untuk menghasilkan uang. Ketika dijalankan, aplikasi ini akan menampilkan main menu yang berisi welcome menu dan beberapa command seperti START, LOAD, dan HELP. Setelah itu, pengguna akan masuk ke login menu dengan command LOGIN, REGISTER, dan HELP. Program akan terus menerima command hingga pengguna memberikan command QUIT.

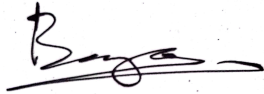
### 8.2 Notulen Rapat

#### Asistensi I

<b>Tanggal : 20 November 2024</b> <b>Tempat : Online (Google Meet)</b> 	<b>Catatan Asistensi:</b> <ul style="list-style-type: none"><li>- Pada mesin kata sempat ada stack overflow karena tidak mendeteksi <code>pitafile == NULL()</code> namun langsung <code>ADVWORD()</code></li><li>- Biasanya isi dari <code>STARTWORD()</code> itu hanya <code>ADVWORD()</code></li><li>- <code>misc.c</code> dipindahkan dari folder Command ke src</li><li>- Diperbolehkan menggunakan <code>atoi()</code> untuk mengubah string menjadi integer</li><li>- Enum atau Enumeration pada bahasa C untuk merepresentasikan tipe data yang berisi sekumpulan konstanta</li><li>- Perlu melakukan realokasi jika barang sudah dalam keadaan penuh</li><li>- Disarankan membuat driver per ADT untuk testing</li></ul>
---	---

**Kehadiran Anggota Kelompok:**

1  
18223115



2  
18223117



3  
18223119



4  
18223126



5  
18223131



Referensi:

<https://github.com/Safiqq/AlstrukdatSTI-Tubes>

<https://github.com/fawwazabrians/Tugas-Besar-IF2111>  
1

**Tanda Tangan Asisten:**



### 8.3 Log Activity Anggota Kelompok

Hari, tanggal	Nama Lengkap - NIM	Kegiatan
Senin, 11 November 2024	Bagas Noor Fadhilah 18223115	Inisialisasi Repository github
Selasa, 12 November 2024		Memasukkan ADT ke dalam repository
Sabtu, 16 November 2024		<ul style="list-style-type: none"> <li>- Mencocokkan ADT praktikum untuk dapat memproses kode ADT tubes</li> <li>- Memperbaiki mesinkata dan mesinkarakter agar dapat menerima input dan membaca file direktori</li> <li>- Membuat kode start</li> <li>- Memperbaiki array dinamis</li> <li>- Menstrukturkan direktori pengerjaan</li> </ul>
Minggu, 17 November 2024		<ul style="list-style-type: none"> <li>- Membuat queue menjadi circular queue</li> <li>- Membuat makefile untuk mempermudah kompilasi</li> <li>- Membuat kode store request</li> <li>- Membuat store supply</li> </ul>
Senin, 18 November 2024		<ul style="list-style-type: none"> <li>- Menambahkan folder misc.c untuk mempermudah pengerjaan</li> <li>- Membuat fungsi untuk menerima input pengguna dengan memperhatikan panjang input</li> </ul>
Selasa, 19 November 2024		Debugging/Review kode load, DoWork
Rabu, 20 November 2024		<ul style="list-style-type: none"> <li>- Debugging queue karena tidak dapat handle circular queue</li> <li>- Debugging mesinkata</li> <li>- Debugging store supply, DoWork, Register, TebakAngka, Load</li> <li>- Menambahkan prosedur baru pada array dinamis</li> </ul>
Kamis, 21 November 2024		<ul style="list-style-type: none"> <li>- Membuat main.c sebagai fungsi utama</li> <li>- Debugging save, work</li> </ul>

		challenge, quit - Menstruktur kembali file direktori untuk mengikuti arahan asisten
Selasa, 12 November 2024	Khairunnisa Azizah 18223117	Membuat tabel pembagian kerja
Rabu, 13 November 2024		- Membuat code store list dan help - Mengerjakan penjelasan tambahan spesifikasi tugas
Kamis, 14 November 2024		Melanjutkan code store list
Minggu, 17 November 2024		Membuat code work
Rabu, 20 November 2024		- Membuat code wordl3 - Membuat readme
Kamis, 21 November 2024		Membuat driver test array, arraydin, queue
Jumat, 22 November 2024		Membuat driver test mesin kata dan mesin karakter
Minggu, 24 November 2024		Membuat laporan bagian Test Script
Senin, 18 November 2024	Maria Vransiska P. C. T. D. P 18223119	Membuat code tebak angka
Selasa, 19 November 2024		Melanjutkan code tebak angka
Rabu, 20 November 2024		Membuat code quit
Kamis, 21 November 2024		Melanjutkan code quit dan membuat laporan
Minggu, 24 November 2024		- Membuat laporan bagian struktur data (ADT): mesinkarakter, misc - Membuat laporan bagian penjelasan program utama
Minggu, 17 November 2024	Theresia Ivana M. S 18223126	Membuat code load
Senin, 18 November 2024		Melanjutkan code load dan membuat code save

Selasa, 19 November 2024		Melanjutkan code load dan save
Rabu, 20 November 2024		Membuat laporan bagian lampiran dan melakukan notulensi asistensi
Kamis, 21 November 2024		Melanjutkan code save
Sabtu, 23 November 2024		<ul style="list-style-type: none"> <li>- Membuat laporan bagian ringkasan dan struktur data ADT array dan queue</li> <li>- Melengkapi bagian algoritma menarik</li> <li>- Merapikan notulen asistensi &amp; formatting laporan</li> </ul>
Senin, 11 November 2024	Aulia Azka Azzahra 18223131	Membuat link terpusat
Sabtu, 16 November 2024		Membuat code login dan logout
Minggu, 17 November 2024		Membuat code register
Selasa, 19 November 2024		Melanjutkan code login, register, dan logout
Rabu, 20 November 2024		<ul style="list-style-type: none"> <li>- Membuat laporan bagian lampiran</li> <li>- Revisi code login dan register</li> </ul>
Sabtu, 23 November 2024		<ul style="list-style-type: none"> <li>- Melanjutkan laporan bagian Struktur Data (ADT) : arraydin, boolean, dan mesin</li> <li>- Membuat laporan Data Test.</li> </ul>
Minggu, 24 November 2024		<ul style="list-style-type: none"> <li>- Melanjutkan laporan Data Test</li> <li>- Merapikan laporan</li> </ul>