

# LAPORAN TUGAS BESAR

## IF2111 Algoritma dan Struktur Data STI


### PURRMART

Dipersiapkan oleh:

Kelompok 6 - K02

Brandon Theodore Ferrinov	(18223020)
Ratukhansa Salsabila	(18223034)
Darryl Rayhananta Adenan	(18223042)
Raditya Zaki Athaya	(18223086)
Matthew Sebastian Kurniawan	(18223096)

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-K02-06</i>		37
		<i>Revisi</i>	0	22/12/2024

# Daftar Isi

1	Ringkasan	4
2	Penjelasan Tambahan Spesifikasi Tugas	5
	2.1 Spesifikasi Fitur globalAlignment	5
	2.2 Spesifikasi Fitur optimasiRute	5
3	Struktur Data (ADT)	5
	3.1 ADT List	5
	3.2 ADT Mesin Karakter	6
	3.3 ADT Mesin Kata	6
	3.4 ADT List Dinamis	6
	3.5 ADT Custom : User	7
	3.6 ADT Custom : Barang	7
	3.7 ADT Stack	7
	3.8 ADT Set dan Map	8
	3.9 ADT List Linier	8
4	Program Utama	9
	Penjelasan	9
	Menjalankan program	11
5	Algoritma-Algoritma Menarik	13
	5.1 Algoritma command, command2, command3 dalam fungsi main()	13
6	Data Test	14
	6.1 Data Test PROFILE	14
	6.2 Data Test CART ADD <nama> <n>	14
	6.3 Data Test CART REMOVE <nama> <n>	15
	6.4 Data Test CART SHOW	16
	6.5 Data Test CART PAY	17
	6.6 Data Test HISTORY <n>	20
	6.7 Data Test WISHLIST ADD	21
	6.8 Data Test WISHLIST SWAP <i> <j>	22
	6.9 Data Test WISHLIST REMOVE <i>	23
	6.10 Data Test WISHLIST REMOVE	24
	6.11 Data Test WISHLIST CLEAR	25
	6.12 Data Test WISHLIST SHOW	25
	6.13 Data Test STORE LIST	26
	6.14 Data Test Bonus GlobalAlignment	27
	6.15 Data Test Bonus OptimasiRute	28
7	Test Script	29
8	Pembagian Kerja dalam Kelompok	33

9	Lampiran	34
9.1	Deskripsi Tugas Besar	34
9.2	Notulen Rapat	34
9.3	Log Activity Anggota Kelompok	36
9.4	Lampiran Link Repository	37

# 1 Ringkasan

Agen Purry membantu OWCA mengatasi permasalahan yang disebabkan oleh Dr. Asep Spakbor. Dr. Asep Spakbor menciptakan suatu mesin penghancur bernama ‘Oppenheimer-inator’ yang akan menghancurkan wilayah tiga negara bagian, yang mengakibatkan pasokan barang semakin menipis. Untuk menghentikan ini, Agen Purry membuka tokonya untuk membantu OWCA dalam menghadapi kendala pasokan barang perang. Namun, OWCA tidak memiliki akses transportasi untuk mencapai lokasi toko tersebut. Sehingga diciptakan aplikasi bernama Purrmart yang memungkinkan OWCA mengakses barang-barang perang untuk bisa memesan dan mengelola kebutuhan perang secara *virtual*.

Purrmart merupakan sebuah aplikasi supermarket yang mensimulasikan aktivitas beli barang. Aplikasi ini berbasis CLI (*Command-line Interface*) dan diprogram menggunakan bahasa C. Purrmart menggunakan struktur data list, mesin karakter, mesin kata, queue, dan custom. Pengguna dapat melakukan berbagai operasi seperti melihat barang toko, meminta dan menyuplai barang ke toko, menyimpan dan membeli barang dalam keranjang, menampilkan barang yang sudah dibeli, hingga bekerja untuk menghasilkan uang. *Command* akan di input oleh pengguna.

Laporan ini berisikan penjelasan mengenai aplikasi PurrMart yang telah kami buat. Laporan ini mencakup penjelasan singkat spesifikasi PurrMart, Struktur Data dan Algoritma yang digunakan, Test Data dan Script yang dilakukan, Pembagian kerja, serta lampiran terkait aplikasi PurrMart.

Tugas besar ini disusun untuk menambah wawasan kami sebagai mahasiswa STI yang mengikuti mata kuliah Algoritma Struktur Data STI IF2111 mengenai penggunaan struktur data (ADT) dan bahasa C yang telah kami pelajari di kelas.

## 2 Penjelasan Tambahan Spesifikasi Tugas

### 2.1 Spesifikasi Fitur *globalAlignment*

Fitur *globalAlignment* merupakan implementasi algoritma untuk menyelaraskan dua sekuens DNA dengan masing-masing panjang maksimum 50 karakter, menggunakan algoritma *Needleman-Wunsch*. Sistem ini bertujuan mendeteksi adanya kebocoran senjata biologis berdasarkan perbandingan antara sekuens referensi dengan hasil metagenomik. Program akan memberikan skor dengan ketentuan scoring sebagai berikut: Match bernilai +1, Mismatch bernilai 0, dan Gap Penalty bernilai -1. Kesimpulan diambil berdasarkan skor akhir, di mana jika skor alignment mencapai atau melebihi 80% dari panjang sekuens yang lebih panjang, maka disimpulkan terjadi kebocoran. Sebaliknya, jika skor kurang dari 80%, maka disimpulkan tidak ada kebocoran. Selain itu, algoritma yang diimplementasikan harus efisien dengan kompleksitas tidak melebihi  $\sim O(2n)$ .

### 2.2 Spesifikasi Fitur *optimasiRute*

Fitur *optimasiRute* merupakan sistem yang dirancang untuk menentukan rute pengiriman barang paling efisien bagi perusahaan SiLambat, rekan toko PURRMART. Dalam sistem ini, setiap lokasi pengiriman harus dikunjungi satu kali, dengan jarak total yang seminimal mungkin. Program harus mampu menangani input jumlah lokasi (node), rute yang menghubungkan lokasi-lokasi tersebut (edge), dan jarak antar lokasi. Sistem akan menghitung jalur optimal menggunakan algoritma yang lebih efisien dibandingkan BFS.

## 3 Struktur Data (ADT)

### 3.1 ADT *List*

```
typedef struct {  
    EType A[MaxEl];  
} List;
```

ADT list bertipe integer. ADT List digunakan untuk melakukan penyimpanan data statis, melakukan operasi manipulasi list sederhana dan manajemen kumpulan elemen integer dengan batasan ukuran. ADT list dipilih karena dapat menyimpan data username,

mencatat transaksi, dan *logging* sederhana, serta mudah diimplementasikan. ADT List diimplementasikan dengan nama file “list.c” dan header “list.h”.

### 3.2 ADT Mesin Karakter

ADT Mesin Karakter bertipe character. ADT ini digunakan untuk melakukan *parsing* teks, Membaca file karakter per karakter, *Preprocessing* input sebelum diproses lebih lanjut, implementasi *scanner* sederhana. ADT mesin karakter juga digunakan sebagai dasar untuk membuat mesin kata. Mesin karakter digunakan untuk membaca karakter yang ada, input *command* dari pengguna, membaca isi file, dan diimplementasikan di mesin kata. Mesin karakter diimplementasikan sebagai ADT Mesin Karakter dengan nama file “mesinkarakter.c” dan header “mesinkarakter.h”.

### 3.3 ADT Mesin Kata

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

ADT Mesin Kata bertipe character, merupakan salah satu implementasi dari ADT Mesin Karakter. ADT ini membaca perintah dari input pengguna, mengurai konfigurasi file, mengekstrak informasi dari teks, validasi input, implementasi parser sederhana. Mesin kata digunakan untuk membaca kata yang ada, input *command* dari pengguna, dan membaca isi file. Mesin kata diimplementasikan sebagai ADT Mesin Kata dengan nama file “mesinkata.c” dan header “mesinkata.h”.

### 3.4 ADT *List* Dinamis

```
typedef struct {
    Barang *A;
    int Capacity;
    int Neff;
} ArrayDin;
```

ADT List Dinamis bertipe integer. ADT List Dinamis digunakan untuk melakukan penyimpanan data dinamis, melakukan operasi manipulasi list dinamis dan manajemen

kumpulan elemen integer tanpa batasan ukuran. ADT list dinamis dipilih karena dapat menyimpan data secara fleksibel dan ukuran yang dinamis, serta mudah diimplementasikan. ADT List dinamis diimplementasikan dengan nama file “dynamiclist.c” dan header “dynamiclist.h”.

### 3.5 ADT Custom : User

```
typedef struct {  
    char name[MAX_LEN];  
    char password[MAX_LEN];  
    int money;  
    char nickname[MAX_LEN];  
    int umur;  
    Map keranjang;  
    Stack riwayat_pembelian;  
    Listlinier wishlist;  
} User;
```

ADT User mendefinisikan struktur data *User* yang memiliki atribut *name* dan *password* masing-masing memiliki maksimal 100 karakter, serta *money* yang berupa integer. ADT *Custom User* dipilih karena sederhana dan efisien dalam menyimpan data *User*. ADT *User* diimplementasikan dalam file header bernama “custom.h”.

### 3.6 ADT Custom : Barang

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

ADT Barang mendefinisikan struktur data *Barang* yang memiliki atribut *name* yang memiliki maksimal 100 karakter dan atribut *price* yang bertipe integer. ADT *Custom Barang* dipilih karena sederhana dan efisien dalam menyimpan data *Barang*. ADT *Barang* diimplementasikan dalam file header bernama “custom.h”.

### 3.7 ADT Stack

```
typedef struct {  
    char* namaBarang;
```

```

    int totalHarga;
} infotypeStack;

typedef struct {
    infotypeStack T[MaxElStack];
    int Top;
    int totalBiaya;
} Stack;

```

ADT Stack bertipe integer. ADT Stack digunakan untuk melakukan pengelolaan data, melakukan operasi pada totalBiaya, totalHarga, dan HISTORY. ADT Stack dipilih karena dapat menyimpan data namabarang dan totalharga. ADT Stack diimplementasikan dengan nama file “stack.c” dan header “stack.h”.

### 3.8 ADT Set dan Map

```

typedef char* keytype;
typedef int valuetype;
typedef int address_map;

typedef struct {
    keytype Key;
    valuetype Value;
} ElmtMap;

typedef struct {
    ElmtMap Elements[MaxElMap];
    int Count;
} Map;

```

ADT Map bertipe integer. ADT ini digunakan untuk mengalokasi memori. ADT Map juga menyimpan pasangan key dan value, serta memungkinkan kita untuk melakukan pencarian, penyisipan, dan penghapusan value berdasarkan key yang ada. Map diimplementasikan sebagai ADT Map dengan nama file “map.c” dan header “map.h”.

### 3.9 ADT List Linier

```

typedef char* infotypelist;
typedef struct tElmtList *address_list;
typedef struct tElmtList {
    infotypelist Info;

```



```

    address_list Next;
} ElmtList;
typedef struct {
    address_list First;
} Listlinier;

```

ADT List linier digunakan untuk melakukan penyimpanan data linier dan melakukan operasi manipulasi list.. ADT list dipilih karena dapat digunakan dalam implementasi WISHLIST. ADT List linier diimplementasikan dengan nama file “listlinier.c” dan header “listlinier.h”.

## 4 Program Utama

<https://github.com/BrandonTheodore/TubesAlstrukdat/blob/main/src/main.c>

### Penjelasan

Dalam program utama, di dalam fungsi main() Program berjalan dalam loop while(true) sehingga terus berjalan hingga pengguna memilih untuk keluar (QUIT) dan melakukan clear terminal setiap input dengan memanggil fungsi clear\_terminal(). Di dalam fungsi main() juga memanggil seluruh fungsi yang ada dalam tugas besar ini. Berikut penjelasan dari seluruh fitur Purrmart :

#### A. Fitur REGISTER, LOGIN, dan PROFILE

- REGISTER : Mendaftarkan pengguna baru. Kemungkinan menyimpan data ke struktur User.
- LOGIN : Login pengguna dengan mencocokkan data pengguna di sistem.

#### B. Fitur WORK dan WORK CHALLENGE

- WORK : Menampilkan cara untuk mendapatkan uang dengan daftar pekerjaan yang bisa dipilih sesuai pendapatan dan durasi.
- WORK CHALLENGE : Menampilkan cara untuk mendapatkan uang dengan melakukan *challenge* yang terdiri atas 2 *challenge* yaitu tebak angka dan WORLD3.

### C. Fitur STORE

- STORE LIST : Menampilkan daftar barang yang tersedia di toko.
- STORE REQUEST : Memungkinkan pengguna meminta barang tertentu.
- BIOWEAPON : Memungkinkan menambahkan BIOWEAPON kedalam STORE REQUEST.
- STORE SUPPLY : Memungkinkan pengguna menambah stok barang ke toko.
- STORE REMOVE : Menghapus barang dari toko.

### C. Fitur CART

- CART ADD : Menambahkan barang ke keranjang belanja. Mendukung input jumlah barang. Jika jumlah tidak valid, maka akan diatur ke default.
- CART REMOVE : Menghapus barang dari keranjang belanja.
- CART SHOW : Menampilkan isi keranjang belanja.
- CART PAY : Memproses pembayaran barang di keranjang.

### D. Fitur WISHLIST

- WISHLIST ADD : Menambahkan barang ke wishlist.
- WISHLIST REMOVE: Menghapus barang dari wishlist berdasarkan nama atau indeks.
- WISHLIST CLEAR : Menghapus semua barang dari wishlist.
- WISHLIST SHOW : Menampilkan isi wishlist.
- WISHLIST SWAP : Menukar posisi dua barang di wishlist.

### E. Fitur Lainnya

- OPTIMASIRUTE : Mencari rute ekspedisi SiLambat yang paling efektif
- GLOBALALIGNMENT : Melakukan penyelarasan dua sekuens
- HISTORY : Menampilkan riwayat transaksi pengguna.
- LOGOUT : Keluar dari akun pengguna.
- SAVE : Menyimpan data ke file dengan nama tertentu (harus berakhiran .txt).
- HELP : Menampilkan panduan penggunaan aplikasi.
- QUIT : Keluar dari aplikasi dengan opsi menyimpan data terlebih dahulu.

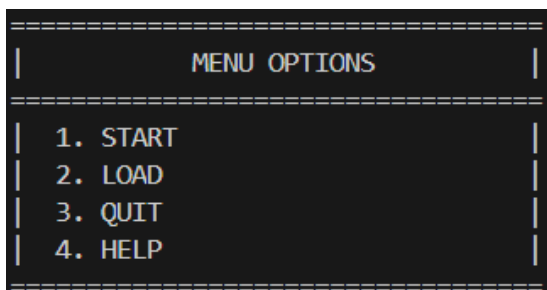
## F. Fungsi ASCII Art

Seluruh opsi di menu utama memiliki ASCII art yang dipanggil dengan fungsi seperti `ascii_quit()`, `ascii_work()`, `ascii_cartadd()`, dll.

## Menjalankan program

### A. Tampilan Awal

Program menampilkan ASCII art dan menu utama dengan opsi:



### B. Proses Input

Program mengambil input perintah menggunakan fungsi `STARTWORD2()`. Input diproses menjadi string (`command`) dan dibandingkan dengan perintah valid menggunakan fungsi `isEqual()`.

### C. Menu Utama

Program memiliki tiga kondisi utama berdasarkan status start dan login:

#### 1. Belum START dan LOGIN:

```
=====
|              MENU OPTIONS              |
|=====|
| 1. START |
| 2. LOAD  |
| 3. QUIT  |
| 4. HELP  |
|=====|
```

Menampilkan menu awal (START, LOAD, QUIT, HELP).

Jika memilih START, program memanggil fungsi MULAI().

Jika memilih LOAD, program memuat file menggunakan fungsi LOAD(txt).

#### 2. START tapi belum LOGIN:

```
=====
|              MENU OPTIONS              |
|=====|
| 1. REGISTER |
| 2. LOGIN    |
| 3. PROFILE  |
| 4. QUIT     |
| 5. HELP     |
|=====|
```

Menampilkan menu dengan opsi:

REGISTER: Mendaftarkan pengguna baru menggunakan fungsi REGISTER().

LOGIN: Login ke aplikasi menggunakan fungsi LOGIN().

PROFILE: Menampilkan profil pengguna.

QUIT: Keluar dari aplikasi (dengan opsi menyimpan data).

HELP: Menampilkan bantuan.

#### 3. Sudah LOGIN:



Menampilkan seluruh fitur yang dapat dipanggil oleh pengguna.

## 5 Algoritma-Algoritma Menarik

### 5.1 Algoritma command, command2, command3 dalam fungsi main()


Algoritma parsing perintah dalam kode menarik karena kegunaannya untuk menangani perintah/input dari user secara efisien. Algoritma ini secara memarsing perintah ke dalam tiga level: command, command2, dan command3, sehingga memungkinkan struktur perintah yang fleksibel. Hal ini dicapai melalui mekanisme parsing satu langkah yang memproses string masukan sambil memperhitungkan spasi dan pembatas perintah.

Algoritma ini digunakan di berbagai perintah, seperti "WORK", "STORE LIST", dan "STORE REQUEST BIOWEAPON", dengan cara memecah masukan menjadi

bagian-bagian yang lebih mudah dikelola dan mencocokkannya dengan perintah yang telah didefinisikan sebelumnya.

## 6 Data Test

### 6.1 Data Test *PROFILE*

Data tes:

Hasil yang diharapkan:
<pre>&gt;&gt; PROFILE Nama : Purry Saldo: 2000  (Silahkan kreasikan atribut yang ditampilkan) // Kembali ke menu utama</pre>

### 6.2 Data Test **CART ADD <nama> <n>**

Data tes:
<pre>// Contoh jika barang belum ada di toko &gt;&gt;&gt; CART ADD lolipop 23</pre>



// Contoh penambahan barang ke keranjang berhasil

```
>>> CART ADD bamburuncing 23
```



Hasil yang diharapkan:

```
>> CART ADD BebekKaliya 240
```

Barang tidak ada di toko!

```
// Perintah invalid; Kembali ke menu utama
```

```
>> CART ADD AK47 20
```

Berhasil menambahkan 20 AK47 ke keranjang belanja!

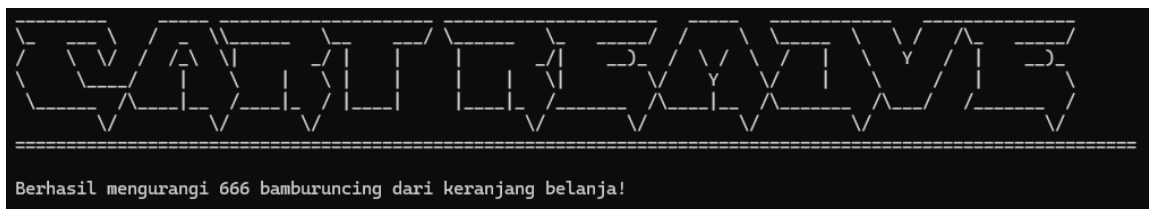
```
// Kembali ke menu utama
```

### 6.3 Data Test CART REMOVE <nama> <n>

Data tes:

// Contoh berhasil meremove barang

```
>>> CART REMOVE bamburuncing 666
```



// Contoh gagal meremove barang (kuantitas pengurangan lebih dari jumlah barang yang

ada)

```
>>> CART REMOVE bamburuncing 500|
```

Tidak berhasil mengurangi, hanya terdapat 333 bamburuncing pada keranjang!

// Contoh gagal meremove barang (barang tidak ada di keranjang)

```
>>> CART REMOVE gulagula 123|
```

Barang tidak ada di keranjang belanja!

Hasil yang diharapkan:

```
>> CART REMOVE AK47 10
```

Berhasil mengurangi 10 AK47 dari keranjang belanja!

```
// Kembali ke menu utama
```

```
>> CART REMOVE AK47 70
```

Tidak berhasil mengurangi, hanya terdapat 10 AK47 pada keranjang!

```
// Asumsi di keranjang belanja jumlah AK47 <70 sehingga perintah  
invalid; Kembali ke menu utama
```

```
>> CART REMOVE BintangSkibidi 70
```

Barang tidak ada di keranjang belanja!

```
// Asumsi tidak ada Bintang Skibidi di keranjang belanja; Kembali ke  
menu utama
```

## 6.4 Data Test CART SHOW

Data tes:

// Contoh keranjang memiliki isi

```
>>> CART SHOW|
```

```
=====
CART SHOW
=====
```

Berikut adalah isi keranjangmu.

Kuantitas	Nama	Total
-----------	------	-------

33	bamburuncing	2178
----	--------------	------

8	kuningmuda	160
---	------------	-----

Total biaya yang harus dikeluarkan adalah 2338.

// Contoh keranjang kosong

```
>>> CART SHOW|
```



```
Keranjang kamu kosong!
```

Hasil yang diharapkan:

```
// Contoh dimana keranjang memiliki isi
>> CART SHOW
Berikut adalah isi keranjangmu.
Kuantitas  Nama      Total
2           AK47      20
1           Lalabu    10
Total biaya yang harus dikeluarkan adalah 30.
```

```
// Command mati; Kembali ke menu utama
```

```
// Contoh yang kosong
>> CART SHOW
Keranjang kamu kosong!
```

## 6.5 Data Test CART PAY

Data tes:

```
>>> CART PAY|
```

// Contoh pembayaran berhasil (Pengguna memasukkan Ya)

```
=====
W A N T A R A N
=====
```

Kamu akan membeli barang-barang berikut.

```
Kuantitas  Nama      Total
33          bamburuncing 2178
8           kuningmuda 160
```

Total biaya yang harus dikeluarkan adalah 2338, apakah jadi dibeli? (Ya/Tidak): Ya

Selamat kamu telah membeli barang-barang tersebut!

// Contoh pembayaran gagal (pengguna tidak memiliki uang yang cukup)

```
Kamu akan membeli barang-barang berikut.  
Kuantitas Nama Total  
999 bamburuncing 65934  
Total biaya yang harus dikeluarkan adalah 65934, apakah jadi dibeli? (Ya/Tidak): Ya  
Uang kamu hanya 7762, tidak cukup untuk membeli keranjang!
```

// Contoh pembayaran gagal (pengguna memasukkan Tidak)

```
Kamu akan membeli barang-barang berikut.  
Kuantitas Nama Total  
33 bamburuncing 2178  
8 kuningmuda 160  
Total biaya yang harus dikeluarkan adalah 2338, apakah jadi dibeli? (Ya/Tidak): Tidak
```

// Contoh pembayaran gagal (pengguna memasukkan input aneh)

```
Kamu akan membeli barang-barang berikut.  
Kuantitas Nama Total  
33 bamburuncing 2178  
8 kuningmuda 160  
Total biaya yang harus dikeluarkan adalah 2338, apakah jadi dibeli? (Ya/Tidak): HAH  
Input tidak valid.
```

// Contoh pembayaran gagal (keranjang kosong)

```
Keranjang kamu kosong!
```

Hasil yang diharapkan:

// Contoh pembayaran yang berhasil (Pengguna memasukan Ya)

>> **CART PAY**

Kamu akan membeli barang-barang berikut.

```
Kuantitas Nama Total  
2 AK47 20  
1 Lalabu 10
```

Total biaya yang harus dikeluarkan adalah 30, apakah jadi dibeli?  
(Ya/Tidak): Ya

Selamat kamu telah membeli barang-barang tersebut!

// Command mati; Kembali ke main menu

// Contoh pembayaran yang gagal (Pengguna tidak memiliki uang yang cukup)

>> **CART PAY**

Kamu akan membeli barang-barang berikut.

```
Kuantitas Nama Total  
2 AK47 20  
1 Lalabu 20
```

Total biaya yang harus dikeluarkan adalah 40, apakah jadi dibeli?  
(Ya/Tidak): **Ya**

Uang kamu hanya 15, tidak cukup untuk membeli keranjang!

// Command mati; Kembali ke main menu

// Contoh pembayaran yang gagal (Pengguna memasukan Tidak)

>> **CART PAY**

Kamu akan membeli barang-barang berikut.

Kuantitas	Nama	Total
-----------	------	-------

2	AK47	20
---	------	----

1	Lalabu	10
---	--------	----

Total biaya yang harus dikeluarkan adalah 30, apakah jadi dibeli?  
(Ya/Tidak): **Tidak**

// Command mati; Kembali ke main menu

// Contoh pembayaran yang gagal (Pengguna memasukan masukan aneh)

>> **CART PAY**

Kamu akan membeli barang-barang berikut.

Kuantitas	Nama	Total
-----------	------	-------

2	AK47	20
---	------	----

1	Lalabu	10
---	--------	----

Total biaya yang harus dikeluarkan adalah 30, apakah jadi dibeli?  
(Ya/Tidak): **Purry**

// Command mati; Kembali ke main menu

// Contoh pembayaran yang gagal (Keranjang kosong)

>> **CART PAY**

Keranjang kamu kosong!

## 6.6 Data Test HISTORY <n>

Data tes:

// Contoh pengguna belum memiliki riwayat pembelian

>>> **HISTORY 5**



// Contoh riwayat pembelian terisi

```
Riwayat pembelian barang:
Barang: Meong, Harga: 500
Barang: Lalabu, Harga: 360
Barang: AK47, Harga: 220
```

Hasil yang diharapkan:

// Contoh menunjukkan riwayat pembelian  $N < \text{total riwayat}$

>> **HISTORY 3**

Riwayat pembelian barang:

1. AK47 40
2. AK47 100
3. Lalabu 35

// Command mati; Kembali ke main menu

// Contoh menunjukkan riwayat pembelian  $N \geq \text{total riwayat}$

>> **HISTORY 10**

Riwayat pembelian barang:

1. AK47 40
2. AK47 100
3. Lalabu 35
4. AK47 10
5. Meong 500
6. Ayam Goreng Crisbar 20

// Command mati; Kembali ke main menu

// Contoh riwayat pembelian kosong

Kamu belum membeli barang apapun!

## 6.7 Data Test WISHLIST ADD

Data tes:

// Berhasil menambahkan wishlist

```
>>> WISHLIST ADD|
```



Masukkan nama barang: bamburuncing  
Berhasil menambahkan bamburuncing ke wishlist!

// Barang sudah ada dalam wishlist

```
>>> WISHLIST ADD|
```

Masukkan nama barang: bamburuncing  
bamburuncing sudah ada di wishlist!

// Gagal menambahkan wishlist karena tidak ada barang dengan nama terkait

```
>>> WISHLIST ADD|
```

Masukkan nama barang: gundamrusak  
Tidak ada barang dengan nama gundamrusak!

Hasil yang diharapkan:

```
>> WISHLIST ADD
```

Masukkan nama barang: Ayam Geprek Bakar Crispy Besthal

Berhasil menambahkan Ayam Geprek Bakar Crispy Besthal ke wishlist!

```
>> WISHLIST ADD
```

Masukkan nama barang: Ayam Geprek Bakar Crispy Besthal

Ayam Geprek Bakar Crispy Besthal sudah ada di wishlist!

```
>> WISHLIST ADD
```

Masukkan nama barang: Ayam Geprek Sambalado Besthal

Tidak ada barang dengan nama Ayam Geprek Sambalado Besthal!

## 6.8 Data Test WISHLIST SWAP <i> <j>

Data tes:

// Contoh swap berhasil

```
>>> WISHLIST SWAP 1 2|
```

```
WISHLIST SWAP
=====
Berhasil menukar posisi 1 dengan 2 pada wishlist!
```

// Contoh swap gagal karena hanya terdapat satu barang dalam wishlist

```
>>> WISHLIST SWAP 1 2|
```

```
Gagal menukar posisi! Wishlist memiliki kurang dari 2 item.
```

Hasil yang diharapkan:

```
>> WISHLIST SWAP 1 2
```

Berhasil menukar posisi Ayam Geprek Bakar Crispy Besthal dengan Ayam Mangut Besthal pada wishlist!

// Urutan Ayam Geprek Bakar Crispy Besthal berubah dari 1 menjadi 2.  
Sebaliknya, urutan Ayam Mangut Besthal berubah dari 2 menjadi 1

```
>> WISHLIST SWAP 1 2
```

Gagal menukar posisi Ayam Geprek Bakar Crispy Besthal!

// Hanya terdapat satu barang (Ayam Geprek Bakar Crispy Besthal) pada  
wishlist sehingga posisinya tidak dapat ditukar

## 6.9 Data Test WISHLIST REMOVE <i>

Data tes:

// Contoh berhasil menghapus wishlist

```
>>> WISHLIST REMOVE 1|
```

```
WISHLIST REMOVE
```

```
====  
Berhasil menghapus barang posisi ke-1 dari wishlist!
```

// Contoh gagal menghapus wishlist karena urutan barang tidak ada

```
>>> WISHLIST REMOVE 45|
```

```
Penghapusan barang WISHLIST gagal dilakukan, Barang ke-45 tidak ada di WISHLIST!
```

// Contoh gagal karena wishlist kosong

```
>>> WISHLIST REMOVE 1|
```

```
Penghapusan barang WISHLIST gagal dilakukan, WISHLIST kosong!
```

//Contoh gagal karena command tidak valid

```
=====  
>>> WISHLIST REMOVE |
```

```
Penghapusan barang WISHLIST gagal dilakukan, command tidak valid!
```

Hasil yang diharapkan:

```
// Contoh menghapus barang ke-i dari WISHLIST
```

```
>> WISHLIST REMOVE 2
```

```
Berhasil menghapus barang posisi ke-2 dari wishlist!
```

```
// Command mati; Kembali ke main menu
```

```
// Contoh penghapusan wishlist yang gagal (Pengguna memasukkan urutan  
barang yang tidak ada)
```

```
//Misalnya dalam kasus ini, hanya terdapat 5 barang di dalam wishlist
```

```
>> WISHLIST REMOVE 10
```

```
Penghapusan barang WISHLIST gagal dilakukan, Barang ke-10 tidak ada di  
WISHLIST!
```

```
// Command mati; Kembali ke main menu
```

```
// Contoh penghapusan wishlist yang gagal (Wishlist kosong)
```

```
//Misalnya dalam kasus ini, tidak ada barang di dalam wishlist
```

```

>> WISHLIST REMOVE 1
Penghapusan barang WISHLIST gagal dilakukan, WISHLIST kosong!

// Command mati; Kembali ke main menu

// Contoh penghapusan wishlist yang gagal (Pengguna memasukkan perintah
yang tidak valid)
>> WISHLIST REMOVE XY
Penghapusan barang WISHLIST gagal dilakukan, command tidak valid!

// Command mati; Kembali ke main menu

```

## 6.10 Data Test WISHLIST REMOVE

Data tes:

// Contoh berhasil menghapus wishlist

```
>>> WISHLIST REMOVE|
```

```

=====
WISHLIST GAGAL DIHAPUS!
=====
Masukkan nama barang yang akan dihapus : AKU7
AKU7 berhasil dihapus dari WISHLIST!

```

// Contoh gagal menghapus karena barang tidak ada dalam wishlist

```
>>> WISHLIST REMOVE|
```

```

Masukkan nama barang yang akan dihapus : akamsi
Penghapusan barang WISHLIST gagal dilakukan, akamsi tidak ada di WISHLIST!

```

Hasil yang diharapkan:

// Contoh menghapus barang "LaLabu" dari WISHLIST

```
>> WISHLIST REMOVE
```

```

Masukkan nama barang yang akan dihapus : LaLabu
LaLabu berhasil dihapus dari WISHLIST!

```

// Command mati; Kembali ke main menu

// Contoh penghapusan wishlist yang gagal (Barang tidak ada di WISHLIST)



```
>> WISHLIST REMOVE
```

Masukkan nama barang yang akan dihapus : LoremIpsum

Penghapusan barang WISHLIST gagal dilakukan, LoremIpsum tidak ada di WISHLIST!

```
// Command mati; Kembali ke main menu
```

## 6.11 Data Test WISHLIST CLEAR

Data tes:

```
>>> WISHLIST CLEAR
```



Wishlist telah dikosongkan.

Hasil yang diharapkan:

```
>> WISHLIST CLEAR
```

Wishlist telah dikosongkan.

## 6.12 Data Test WISHLIST SHOW

Data tes:

```
// Contoh wishlist kosong
```

```
>>> WISHLIST SHOW
```



Wishlist kamu kosong!

```
// Contoh wishlist terisi
```

```
>>> WISHLIST SHOW
```

<pre> Berikut adalah isi wishlist-mu: 1. bamburuncing 2. Lalabu 3. AK47 </pre>
<p>Hasil yang diharapkan:</p>
<pre> &gt;&gt; WISHLIST SHOW Berikut adalah isi wishlist-mu: 1 Ayam Geprek Bakar Crispy Besthal 2 Ayam Mangut Besthal 3 Karaage Don 4 Torikatsu Don </pre>
<pre> &gt;&gt; WISHLIST SHOW Wishlist kamu kosong! </pre>

## 6.13 Data Test STORE LIST

<p>Data tes:</p>
<pre> &gt;&gt;&gt; STORE LIST </pre> 
<p>Hasil yang diharapkan:</p>
<pre> &gt;&gt; STORE LIST List barang yang ada di toko : - Platypus Laser - Harga: 100 - Shrink Ray - Harga: 500 - Net Shooter - Harga: 250 - Camouflage Cloak - Harga: 150 </pre>

- Sleep Dart Gun - Harga: 300
- Bubble Blaster - Harga: 200

>> **STORE LIST**  
 TOKO KOSONG

## 6.14 Data Test Bonus GlobalAlignment

Data tes:

//Contoh tidak ada kebocoran

```
Masukkan sekuens referensi: TAGTAGAATGGGAGAGGTT
Masukkan sekuens query: TAGTAGGGTTAATGTT
Skor: 9
Sekuens yang telah disejajarkan:
TAGTAGAATGGGAGAGGTT
TAGTAG---GGTTAATGTT

Woah! Tidak ada kebocoran
```

//Contoh ada kebocoran

```
Masukkan sekuens referensi: TAGTAGAATGGGAGAGGCC
Masukkan sekuens query: TAGTAGAATGGGTAAGTC
Skor: 15
Sekuens yang telah disejajarkan:
TAGTAGAATGGGAGAGGCC
TAGTAGAATGGGTAAGTC

Nawh! Ada kebocoran...
```

Hasil yang diharapkan:

>> **GLOBALALIGNMENT**

Masukan sekuens referensi: TAGTAGAATGGGAGAGGTT // Panjang: 19 karakter  
 Masukan sekuens query: TAGTAGGGTTAATGTT // Panjang: 16 karakter

Skor: 9

Sekuens yang telah disejajarkan:  
 TAGTAGAATGGGAGAGGTT  
 TAGTAG---GGTTAATGTT

Woah! Tidak ada kebocoran ( $\geq \cup \leq$ ) //  $19 \times 80\% = 15.20 > 9$  (lebih rendah)

>> GLOBALALIGNMENT

Masukan sekuens referensi: TAGTAGAATGGGAGAGGC // Panjang: 18 karakter

Masukan sekuens query: TAGTAGAATGGGTAAGTC // Panjang: 18 karakter

Skor: 15

Sekuens yang telah disejajarkan:

TAGTAGAATGGGAGAGGC

TAGTAGAATGGGTAAGTC

Nawh! Ada kebocoran... (ಥ\_ಥ) //  $18 \times 80\% = 14.4 < 15$  (lebih tinggi)

## 6.15 Data Test Bonus OptimasiRute

Data tes:

>> OPTIMASIRUTE

Masukkan jumlah lokasi pengiriman (node): 4

Masukkan jumlah rute (edge): 5

Masukkan jarak antarlokasi (weight) dengan format <from> <to> <weight> :

0 1 10

0 2 15

0 3 20

1 2 35

1 3 25

Data diterima, silakan tunggu!☺

A-ha! Rute paling efektif adalah 0-2-1-3.

Total jarak: 75

Hasil yang diharapkan:

>> OPTIMASIRUTE

Masukkan jumlah lokasi pengiriman (node): 4

Masukkan jumlah rute (edge): 5

Masukkan jarak antarlokasi (weight):

0 1 10

0 2 15

0 3 20

1 2 35

1 3 25

Data diterima, silakan tunggu...

A-ha! Rute paling efektif adalah 0-2-1-3 dengan total jarak 75.

## 7 Test Script

No.	Fitur yang Dites	Tujuan Testing	Langkah-Langkah Testing	Input Data Test	Hasil yang Diharapkan	Hasil yang Ditampilkan (Sesuai/Tidak)
1	PROFILE	Memastikan pengguna telah login untuk melihat data diri	Memasukkan <i>command</i> PROFILE	<a href="#">Data Test PROFILE</a>	<a href="#">Data Test PROFILE</a>	SESUAI
2	CART ADD <nama> <n>	Memeriksa fitur CART ADD <nama> <n> untuk menambahkan barang ke dalam keranjang belanja	Memasukkan <i>command</i> CART ADD <nama> <n> dengan dua case : 1. Barang berhasil ditambahkan 2. Barang tidak ada di toko	<a href="#">Data Test CART ADD</a>	<a href="#">Data Test CART ADD</a>	SESUAI
3	CART REMOVE <nama> <n>	Memeriksa fitur CART REMOVE <nama> <n> untuk mengurangi jumlah barang di dalam keranjang	Memasukkan <i>command</i> CART REMOVE <nama> <n> dengan dua case : 1. Barang berhasil dikurangi 2. Barang tidak berhasil dikurangi karena jumlah barang yang kurang dari stok yang ada 3. Barang tidak berhasil dikurangi karena barang tersebut tidak ada di keranjang	<a href="#">Data Test CART REMOVE</a>	<a href="#">Data Test CART REMOVE</a>	SESUAI
4	CART SHOW	Memeriksa fitur CART SHOW untuk melihat barang yang telah dimasukkan ke dalam keranjang	Memasukkan <i>command</i> CART SHOW dengan dua case : 1. List barang yang dimasukkan ke keranjang 2. Tidak ada barang karena	<a href="#">Data Test CART SHOW</a>	<a href="#">Data Test CART SHOW</a>	SESUAI

No.	Fitur yang Dites	Tujuan Testing	Langkah-Langkah Testing	Input Data Test	Hasil yang Diharapkan	Hasil yang Ditampilkan (Sesuai/Tidak)
1	PROFILE	Memastikan pengguna telah login untuk melihat data diri	Memasukkan <i>command</i> PROFILE	<a href="#">Data Test PROFILE</a>	<a href="#">Data Test PROFILE</a>	SESUAI
			keranjang kosong			
5	CART PAY	Memeriksa fitur CART PAY	Memasukkan <i>command</i> CART PAY dengan lima case : <ol style="list-style-type: none"> <li>1. Pembayaran berhasil</li> <li>2. Pembayaran gagal karena uang tidak cukup</li> <li>3. Pembayaran gagal karena mengkonfirmasi untuk tidak jadi membeli</li> <li>4. Pembayaran gagal karena konfirmasi pembayaran yang tidak sesuai</li> <li>5. Pembayaran gagal karena keranjang kosong</li> </ol>	<a href="#">Data Test CART PAY</a>	<a href="#">Data Test CART PAY</a>	SESUAI
6	HISTORY <n>	Memeriksa fitur HISTORY <n> untuk melihat riwayat pembelian	Memasukkan <i>command</i> HISTORY <n> dengan tiga case : <ol style="list-style-type: none"> <li>1. History pembelian kurang dari total riwayat</li> <li>2. History pembelian lebih besar atau sama dengan total riwayat</li> <li>3. History kosong karena belum ada membeli barang</li> </ol>	<a href="#">Data Test HISTORY</a>	<a href="#">Data Test HISTORY</a>	SESUAI

No.	Fitur yang Dites	Tujuan Testing	Langkah-Langkah Testing	Input Data Test	Hasil yang Diharapkan	Hasil yang Ditampilkan (Sesuai/Tidak)
1	PROFILE	Memastikan pengguna telah login untuk melihat data diri	Memasukkan <i>command</i> PROFILE	<a href="#">Data Test PROFILE</a>	<a href="#">Data Test PROFILE</a>	SESUAI
7	WISHLIST ADD	Memeriksa fitur WISHLIST ADD	Memasukkan <i>command</i> WISHLIST ADD dengan tiga case : 1. Penambahan berhasil 2. Penambahan gagal karena sudah ada di WISHLIST 3. Penambahan gagal karena tidak ada barang dengan nama tersebut	<a href="#">Data Test WISHLIST ADD</a>	<a href="#">Data Test WISHLIST ADD</a>	SESUAI
8	WISHLIST SWAP <i> <j>	Memeriksa fitur WISHLIST SWAP <i> <j>	Memasukkan <i>command</i> WISHLIST SWAP <i> <j> dengan dua case : 1. Berhasil menukar posisi 2. Gagal menukar posisi karena hanya ada satu barang dalam wishlist	<a href="#">Data Test WISHLIST SWAP</a>	<a href="#">Data Test WISHLIST SWAP</a>	SESUAI
9	WISHLIST REMOVE <i>	Memeriksa fitur WISHLIST REMOVE <i> untuk menghapus barang berdasarkan posisinya	Memasukkan <i>command</i> WISHLIST REMOVE <i> dengan 4 case : 1. Barang berhasil di remove 2. Barang gagal di remove karena kesalahan dalam urutan barang 3. Barang gagal di remove karena barang tidak ada 4. Barang gagal di remove karena	<a href="#">Data Test WISHLIST REMOVE &lt;i&gt;</a>	<a href="#">Data Test WISHLIST REMOVE &lt;i&gt;</a>	SESUAI

No.	Fitur yang Dites	Tujuan Testing	Langkah-Langkah Testing	Input Data Test	Hasil yang Diharapkan	Hasil yang Ditampilkan (Sesuai/Tidak)
1	PROFILE	Memastikan pengguna telah login untuk melihat data diri	Memasukkan <i>command</i> PROFILE	<a href="#">Data Test PROFILE</a>	<a href="#">Data Test PROFILE</a>	SESUAI
			kesalahan input barang			
10	WISHLIST REMOVE	Memeriksa fitur WISHLIST REMOVE untuk menghapus barang berdasarkan nama barangnya	Memasukkan <i>command</i> WISHLIST REMOVE dengan 2 case : 1. Barang berhasil di remove 2. Barang gagal di remove karena nama barang yang akan di remove tidak ada di keranjang	<a href="#">Data Test WISHLIST REMOVE</a>	<a href="#">Data Test WISHLIST REMOVE</a>	SESUAI
11	WISHLIST CLEAR	Memeriksa fitur WISHLIST CLEAR	Memasukkan <i>command</i> WISHLIST CLEAR	<a href="#">Data Test WISHLIST CLEAR</a>	<a href="#">Data Test WISHLIST CLEAR</a>	SESUAI
12	WISHLIST SHOW	Memeriksa fitur WISHLIST SHOW	Memasukkan <i>command</i> WISHLIST SHOW	<a href="#">Data Test WISHLIST SHOW</a>	<a href="#">Data Test WISHLIST SHOW</a>	SESUAI
13	STORE LIST	Memeriksa fitur STORE LIST	Memasukkan <i>command</i> STORE LIST dan akan menampilkan nama barang beserta harganya.	<a href="#">Data Test STORE LIST</a>	<a href="#">Data Test STORE LIST</a>	SESUAI
14	GLOBAL ALIGNMENT	Memeriksa fitur GLOBAL ALIGNMENT	Memasukkan <i>command</i> GLOBALALIGNMENT	<a href="#">Data Test Bonus GLOBALALIGNMENT</a>	<a href="#">Data Test Bonus GLOBALALIGNMENT</a>	SESUAI
15	OPTIMASIRUTE	Memeriksa fitur OPTIMASIRUTE	Memasukkan <i>command</i> OPTIMASIRUTE	<a href="#">Data Test Bonus OPTIMASIRUTE</a>	<a href="#">Data Test Bonus OPTIMASIRUTE</a>	SESUAI

## 8 Pembagian Kerja dalam Kelompok

Nama Lengkap (NIM)	Deskripsi Tugas
--------------------	-----------------



Brandon Theodore Ferrinov (18223020)	<ul style="list-style-type: none"> <li>- Membantu mengerjakan fungsi start</li> <li>- Membantu mengerjakan fungsi load</li> <li>- Membantu mengerjakan fungsi save</li> <li>- Melakukan data testing</li> </ul>
Ratukhansa Salsabila (18223034)	<ul style="list-style-type: none"> <li>- Mencatat notulensi selama asistensi</li> <li>- Mengisi dokumen form asistensi</li> <li>- Mengerjakan laporan bagian ringkasan</li> <li>- Melengkapi laporan bagian program utama</li> <li>- Mengerjakan laporan bagian data test</li> <li>- Mengerjakan laporan bagian test script</li> <li>- Mengisi laporan bagian lampiran</li> </ul>
Darryl Rayhananta Adenan (18223042)	<ul style="list-style-type: none"> <li>- Melakukan testing setiap fungsi</li> <li>- Membuat fungsi bonus Bioweapon</li> <li>- Membuat fungsi bonus globalAlignment</li> <li>- Membuat fungsi bonus optimasiRute</li> <li>- Melakukan formatting laporan</li> <li>- Mengerjakan laporan bagian struktur data, program utama, data test, test script, &amp; lampiran</li> <li>- Menambahkan dan merapikan driver milestone 2</li> </ul>
Raditya Zaki Athaya (18223086)	<ul style="list-style-type: none"> <li>- Melakukan data testing</li> <li>- Mengerjakan laporan bagian Data Test</li> <li>- Membantu merapikan laporan</li> </ul>
Matthew Sebastian Kurniawan (18223096)	<ul style="list-style-type: none"> <li>- Memperbaiki dan merapikan fungsi milestone 1</li> <li>- Mengerjakan koding bagian fungsi profile</li> <li>- Mengerjakan koding bagian fungsi cart add</li> <li>- Mengerjakan koding bagian fungsi cart remove</li> <li>- Mengerjakan koding bagian fungsi cart show</li> <li>- Mengerjakan koding bagian fungsi cart pay</li> <li>- Mengerjakan koding bagian fungsi wishlist add</li> <li>- Mengerjakan koding bagian fungsi wishlist swap</li> <li>- Mengerjakan koding bagian fungsi wishlist remove 1</li> <li>- Mengerjakan koding bagian fungsi wishlist remove 2</li> <li>- Mengerjakan koding bagian fungsi wishlist clear</li> <li>- Mengerjakan koding bagian fungsi wishlist show</li> <li>- Merubah fungsi bagian START untuk milestone 2</li> <li>- Merubah fungsi bagian LOAD untuk milestone 2</li> <li>- Merubah fungsi bagian SAVE untuk milestone 2</li> <li>- Membuat makefile untuk milestone 2</li> <li>- Merapikan dan menambahkan bagian main</li> <li>- Menambahkan bonus untuk ke main</li> </ul>

	<ul style="list-style-type: none"> <li>- Menambahkan ascii art</li> <li>- Melakukan dan memperbaiki error handling</li> <li>- Menambahkan dan memodifikasi adt untuk milestone 2</li> <li>- Mengontak dan sebagai sumber komunikasi asisten</li> <li>- Menambahkan algoritma menarik pada laporan</li> <li>- Menambahkan driver adt untuk milestone 2</li> </ul>
--	--

## 9 Lampiran

### 9.1 Deskripsi Tugas Besar

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**.

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

### 9.2 Notulen Rapat

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





No. Kelompok/Kelas : Kelompok 6 / K-02  
Nama Kelompok : Daemon  
Anggota Kelompok (Nama/NIM) : 1. Darryl Rayhananta Adenan (18223042)  
2. Matthew Sebastian Kurniawan (18223096)  
3. Brandon Theodore Ferrinov (18223020)  
4. Ratukhansa Salsabila (18223034)  
5. Raditya Zaki Athaya (18223086)



STEI- ITB	<nomor dokumen>	Halaman 34 dari 37 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.		

Asisten Pembimbing

: Rayhan Maheswara Pramanda

Asistensi II

<b>Tanggal : 19 Desember 2024</b>	<b>Catatan Asistensi:</b> <ol style="list-style-type: none"><li>1. ADT yang ditulis pada milestone 2 adalah ADT yang digunakan di milestone 2 tetapi jika terdapat ADT pada milestone 1 yang digunakan di milestone 2 ini, maka tetap dituliskan pada dokumen</li><li>2. Boleh tetap mengerjakan bonus pada milestone 2 walaupun saat milestone 1 bonusnya tidak dikerjakan</li><li>3. Jika ingin memperbaiki milestone 1 diperbolehkan karena penilaian akhir adalah secara keseluruhan</li><li>4. Repo masih sama, disambung dengan yang sebelumnya</li><li>5. Untuk laporan dipisah antara milestone 1 dengan laporan milestone 2</li><li>6. Jika masih ada buck di milestone 1 diperbolehkan untuk memperbaikinya</li><li>7. Utamakan dulu untuk menyelesaikan fitur utama sebelum lanjut ke bonus</li><li>8. Di quit ada pilihan save dan jika ingin ditambahkan implementasi untuk save dimana diperbolehkan namun tidak ada penambahan nilai karena hanya implementasi</li></ol>
<b>Tempat : Google Meet</b>	
<b>Kehadiran Anggota Kelompok:</b>	
No	
NIM	
Tanda tangan	
1	
18223042	
	
2	
18223096	
	
3	
18223020	
	
4	
18223034	
	
5	
18223086	

	
	Tanda Tangan Asisten: 

### 9.3 Log Activity Anggota Kelompok

Tanggal	Nama (NIM)	Aktivitas
8 Desember 2024	<ul style="list-style-type: none"> <li>• Ratukhansa Salsabila (18223034)</li> <li>• Darryl Rayhananta Adenan (18223042)</li> <li>• Matthew Sebastian Kurniawan (18223096)</li> <li>• Brandon Theodore Ferrinov (18223020)</li> <li>• Raditya Zaki Athaya (18223086)</li> </ul>	Pembagian tugas
8-19 Desember 2024	<ul style="list-style-type: none"> <li>• Darryl Rayhananta Adenan (18223042)</li> <li>• Matthew Sebastian Kurniawan (18223096)</li> </ul>	Pengerjaan Code
19 Desember 2024	<ul style="list-style-type: none"> <li>• Ratukhansa Salsabila (18223034)</li> <li>• Darryl Rayhananta Adenan (18223042)</li> <li>• Matthew Sebastian Kurniawan (18223096)</li> <li>• Brandon Theodore Ferrinov (18223020)</li> <li>• Raditya Zaki Athaya (18223086)</li> </ul>	Asistensi Milestone 2
19-20	<ul style="list-style-type: none"> <li>• Matthew Sebastian Kurniawan</li> </ul>	Finalisasi Code

Desember 2024	(18223096)	
19-20 Desember 2024	<ul style="list-style-type: none"> <li>• Ratukhansa Salsabila (18223034)</li> <li>• Darryl Rayhananta Adenan (18223042)</li> <li>• Matthew Sebastian Kurniawan (18223096)</li> <li>• Raditya Zaki Athaya (18223086)</li> </ul>	Pengerjaan Laporan

## 9.4 Lampiran Link Repository

<https://github.com/BrandonTheodore/TubesAlstrukdat.git>