

LAPORAN TUGAS BESAR
IF2111 Algoritma dan Struktur Data STI
Purrrmart 2



Dipersiapkan oleh:

Kelompok 09 - Kelas 02

Karunia Mega Lestari	/ 18221126
Mahesa Satria Prayata	/ 18223082
Rafli Dwi Nugraha	/ 18223038
M Khalfani Shaquille Indrajaya	/ 18223104
Alghan Pridanusuta	/ 18223058
Michael Ballard Isaiah Silaen	/ 18223080

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

	Sekolah Teknik Elektro dan Informatika ITB	Nomor Dokumen		Halaman
		IF2111-TB-02-09		37
	Revisi	0		Desember 1 2024

Daftar Isi

1	Ringkasan.....	3
2	Penjelasan Tambahan Spesifikasi Tugas.....	4
	2.1 File Konfigurasi.....	4
	2.2 <Spesifikasi Fitur Tambahan 2>.....	5
3	Struktur Data (ADT).....	5
	3.1 ADT Stack.....	5
	3.2 ADT Map.....	6
	3.3 ADT Linked List (linkedlist.h).....	8
4	Program Utama.....	10
5	Algoritma-Algoritma Menarik.....	10
	5.1 swapIdx.....	10
	5.2 isInList.....	11
	5.3 <Algoritma menarik 3>.....	11
6	Data Test.....	12
	6.1 <Data Test 1>.....	12
7	Test Script.....	12
8	Pembagian Kerja dalam Kelompok.....	12
9	Lampiran.....	12
	9.1 Deskripsi Tugas Besar.....	12
	9.2 Notulen Rapat.....	14
	9.3 Log Activity Anggota Kelompok.....	16

1 Ringkasan

Milestone ini merupakan lanjutan tugas kami sebagai programmer OWCA. Pada milestone kemarin terdapat fitur berupa store yang belum dapat digunakan, sehingga pada Milestone ini akan lebih terfokus pada fitur yang memanfaatkan store seperti CART, HISTORY, WISHLIST. Ada juga fitur tambahan berupa PROFILE untuk melihat statistik dari user.

Seperti namanya, fitur CART digunakan untuk mengumpulkan barang kemudian membelinya dengan harga total dari barang tersebut. Dapat menambah, menghapus, dan membeli barang di CART. HISTORY merupakan riwayat pembelian yang disimpan dengan sebuah stack, berisi barang yang baru dibeli user dan total uang yang dikeluarkan user. WISHLIST digunakan untuk mengetahui barang apa saja yang ingin dibeli agar tidak lupa di kemudian hari. PROFILE adalah fitur untuk melihat statistik dari pemain seperti nama, sisa uang user, dll.

Selain spesifikasi-spesifikasi diatas, kami juga membuat algoritma tambahan untuk membantu OWCA, yaitu dengan membuat prorgram Optimasi Rute, agar Purmart dengan mengirim barang melewati rute yang paling efisien dan Riwayat Maksimal agar detail setiap pembelian mudah dilihat.

2 Penjelasan Tambahan Spesifikasi Tugas

2.1 File Konfigurasi

4	# Banyaknya barang di dalam toko
10 AK47	
20 Lalabu	
20 Ayam Goreng Crisbar	
500 Meong	
2	# Banyaknya pengguna di dalam program
100 user1 alstrukdatkeren	# Data pengguna “user1”
6	# Banyaknya history pengguna “user1”
2 40	# Riwayat 1 dan total
20 2 AK47	
20 1 Lalabu	
2 100	# Riwayat 2 dan total
80 8 AK47	
20 1 Lalabu	
2 35	# Riwayat 3 dan total
15 2 M14	
20 1 Lalabu	
1 10	# Riwayat 4 dan total

10 1 AK47 1 500 500 1 Meong 1 20 20 1 Ayam Goreng Crisbar 3 Meong AK47 Ayam Goreng Crisbar 25 user2 kerenbangetkak 0 2 Lalabu AK47	# Riwayat 5 dan total # Riwayat 6 dan total # Banyaknya wishlist pengguna "user1" # Data pengguna "user2" # Banyaknya history pengguna "user2" # Banyaknya wishlist pengguna "user2"
---	---

2.2 Riwayat Maksimal

Riwayat maksimal ini dapat dikatakan sebagai fitur yang lebih baik dari history. Jadi tidak hanya menampilkan total dan satu barang saja, fitur history sekarang akan menampilkan nama barang, jumlah masing-masing barang, total harga dari barang, dan total harga dari keseluruhan barang yang dibeli.

/* history post-Riwayat Maksimal */ >> HISTORY 3 Riwayat pembelian barang: Pembelian 1 - Total 40 Kuantitas Nama Total 2 AK47 20 1 Lalabu 20 Pembelian 2 - Total 100 Kuantitas Nama Total 8 AK47 80 1 Lalabu 20 Pembelian 3 - Total 35 Kuantitas Nama Total 2 M14 15 1 Lalabu 20	/* history pre-Riwayat Maksimal */ >> HISTORY 3 Riwayat pembelian barang: 1. AK47 40 2. AK47 100 3. Lalabu 35
--	--

2.3 Optimasi Rute

```
/* Untuk code ada di bagian Algoritma Menarik */
```

Bonus optimasi rute ini menggunakan pemikiran Kruskal. Awalnya ketika data node dan edge dimasukkan, langsung di-sort menggunakan algoritma Bubble Sort berdasarkan cost dari edge yang ada, dan diurutkan dari termurah hingga termahal. Kemudian optimasirute() melakukan kalkulasi dengan approach algoritma Kruskal dengan mengambil edge-edge dengan harga yang lebih murah terlebih dahulu. Beberapa kasus di handling, sehingga tidak ada pengambilan set node dengan “parent” yang sama. Di beberapa kasus juga terdapat algoritma mini extra traversal untuk mengganti parent dari satu node-chain secara keseluruhan.

3 Struktur Data (ADT)

3.1 ADT Stack

```
/* File stackhistory.h */

#ifndef STACKHISTORY_H
#define STACKHISTORY_H

#define Nill -1
#define MaxEl 100
#define MaxItems 10
#define NamaMax MAX_LEN

typedef struct {
    int id;
    int total;
    int jumlah_barang;
    Cart cart[MaxItems];
} Pembelian;

typedef Pembelian infotype;
typedef int address;

typedef struct {
    infotype T[MaxEl];
    address TOP;
} Stack;

#define Top(S) (S).TOP
#define InfoTop(S) (S).T[(S).TOP]

/* ***** Prototype *****/
/* *** Konstruktor/Kreator *** */
void CreateEmptyStack(Stack *S);
/* I.S. sembarang; */
```

```

/* F.S. Membuat sebuah stack S yang kosong berkapasitas MaxEl */
/* jadi indeksnya antara 0.. MaxEl */
/* Ciri stack kosong : TOP bernilai Nil */

/* ***** Predikat Untuk test keadaan KOLEKSI **** */
boolean IsStackEmpty(Stack S);
/* Mengirim true jika Stack kosong: lihat definisi di atas */
boolean IsStackFull(Stack S);
/* Mengirim true jika tabel penampung nilai elemen stack penuh */

/* ***** Menambahkan sebuah elemen ke Stack **** */
void Push(Stack * S, infotype X);
/* Menambahkan X sebagai elemen Stack S. */
/* I.S. S mungkin kosong, tabel penampung elemen stack TIDAK penuh */
/* F.S. X menjadi TOP yang baru, TOP bertambah 1 */

/* ***** Menghapus sebuah elemen Stack **** */
void Pop(Stack * S, infotype* X);
/* Menghapus X dari Stack S. */
/* I.S. S tidak mungkin kosong */
/* F.S. X adalah nilai elemen TOP yang lama, TOP berkurang 1 */

int stackLength(Stack S);
// Mengirim panjang stack (jumlah elemen dalam stack)

void PrintStackHistory(Stack S, int N);
// Menampilkan riwayat pembelian sebanyak N, dimulai dari pembelian terbaru (top element)
// Stack mungkin kosong, N mungkin lebih banyak dari neff stack

void AddtoPembelian(Map *M, Stack *stackhistory);
// Menambahkan pembelian dari Map ke Stack

#endif

```

Menggunakan stack untuk membuat fitur history dalam program toko sangat cocok karena stack mengikuti prinsip Last-In-First-Out (LIFO), yang sesuai dengan konsep history di mana tindakan atau halaman terakhir muncul paling atas. Stack memungkinkan navigasi yang mudah dengan operasi push dan pop, efisien dalam penggunaan memori, dan sederhana untuk diimplementasikan.

3.2 ADT Map

```

/* File map.h */

// #define false 0
// #define true 1
#define Nil 0
#define MaxEl 100
#define Undefined -999

```

```

// typedef int bool;

typedef struct {
    Barang Barang;
    int total;
    int Kuantitas;
} Cart;

typedef struct {
    Cart Elements[MaxEl];
    int Count;
} Map;

/* Definisi Map M kosong : M.Count = Nil */
/* M.Count = jumlah element Map */
/* M.Elements = tempat penyimpanan element Map */

/* ***** Prototype ***** */

/* *** Konstruktor/Kreator *** */
void CreateEmpty(Map *M);
/* I.S. Sembarang */
/* F.S. Membuat sebuah Map M kosong berkapasitas MaxEl */
/* Ciri Map kosong : count bernilai Nil */

/* ***** Predikat Untuk test keadaan KOLEKSI ***** */
boolean IsEmpty(Map M);
/* Mengirim true jika Map M kosong*/
/* Ciri Map kosong : count bernilai Nil */

boolean IsFull(Map M);
/* Mengirim true jika Map M penuh */
/* Ciri Map penuh : count bernilai MaxEl */

/* ***** Operator Dasar Map ***** */
int Value(Map M, Barang k);
/* Mengembalikan nilai value dengan key k dari M */
/* Jika tidak ada key k pada M, akan mengembalikan Undefined */

void Insert(Map *M, Barang k, int v);
/* Menambahkan Elmt sebagai elemen Map M. */
/* I.S. M mungkin kosong, M tidak penuh
   M mungkin sudah beranggotakan v dengan key k */
/* F.S. v menjadi anggota dari M dengan key k. Jika k sudah ada, operasi tidak
dilakukan */

void Delete(Map *M, Barang k);
/* Menghapus Elmt dari Map M. */
/* I.S. M tidak kosong
   element dengan key k mungkin anggota / bukan anggota dari M */
/* F.S. element dengan key k bukan anggota dari M */

```

```

boolean IsMember(Map M, Barang k);
/* Mengembalikan true jika k adalah member dari M */

#endif

```

Map cocok untuk membuat fitur cart dalam program toko karena map dapat menyimpan pasangan key-value, di mana key merepresentasikan produk dan value merepresentasikan jumlah atau informasi terkait lainnya. Map memungkinkan akses, penambahan, dan penghapusan produk dalam cart secara efisien berdasarkan key. Dengan menggunakan map dapat dengan mudah melacak produk-produk dalam cart, mengupdate jumlahnya, menghitung total harga, dan melakukan operasi lain yang relevan dengan cart.

3.3 ADT User (Update)

```

/* File user.h */

#include "boolean.h"
#include "stackhistory.h"
#include "map.h"
#include "../command/wishlist.h"

#define MAX_USERS 10
#define MAX_LEN 100

typedef struct {
    char name[MAX_LEN];
    char password[MAX_LEN];
    int money;
    Map keranjang;
    Stack riwayat_pembelian;
    List wishlist;
} User;

typedef struct {
    User buffer[MAX_USERS];
    int nEff;
} ListUser;

void CreateListUser(ListUser *list);
boolean IsUserExist(ListUser *list, char *username);
void AddUser(ListUser *list, User newUser);
int FindUser(ListUser *list, char *username);
void CartPay(ListUser *list, User *user, Map *M);
void DisplayCart(Map M);
void AddCartToUser(Map M, User *user);
void AddWishlistToUser(List wishlist, User *user);

#endif

```

ADT ini cocok untuk digunakan agar semua data yang disimpan oleh user akan terpusat. ADT ini juga memungkinkan update real-time setelah dilakukan perubahan dari wishlist, pembelian,

dan keranjang. Dengan terpusatnya data, fitur save juga akan menjadi lebih mudah karena hanya terpacu pada user list.

3.4 ADT Linked List

```
/* File linkedlist.h */

typedef char* ElType;
typedef struct node* Address;
typedef struct node {
    ElType info;
    Address next;
} Node;
typedef Address List;

#define IDX_UNDEF_LIST (-1)
#define FIRST(l) (l)
#define INFO(p) (p)->info
#define NEXT(p) (p)->next

/* PROTOTYPE */
/****************** PEMBUATAN LIST KOSONG *****/
void CreateList(List *l);
/* I.S. sembarang */
/* F.S. Terbentuk list kosong */

/****************** TEST LIST KOSONG *****/
boolean isEmptyList(List l);
/* Mengirim true jika list kosong */

/****************** GETTER SETTER *****/
ElType getElmt(List l, int idx);
/* I.S. l terdefinisi, idx indeks yang valid dalam l, yaitu 0..lengthList(l) */
/* F.S. Mengembalikan nilai elemen l pada indeks idx */

void setElmt(List *l, int idx, ElType val);
/* I.S. l terdefinisi, idx indeks yang valid dalam l, yaitu 0..lengthList(l) */
/* F.S. Mengubah elemen l pada indeks ke-idx menjadi val */

int indexOf(List l, ElType val);
/* I.S. l, val terdefinisi */
/* F.S. Mencari apakah ada elemen list l yang berisi val */
/* Jika ada, mengembalikan indeks elemen pertama l yang berisi val */
/* Mengembalikan IDX_UNDEF_LIST jika tidak ditemukan */

/****************** PRIMITIF BERDASARKAN NILAI *****/
/** PENAMBAHAN ELEMEN ***/
void insertFirst(List *l, ElType val);
/* I.S. l mungkin kosong */
/* F.S. Melakukan alokasi sebuah elemen dan */
/* menambahkan elemen pertama dengan nilai val jika alokasi berhasil. */
/* Jika alokasi gagal: I.S.= F.S. */

void insertLast(List *l, ElType val);
/* I.S. l mungkin kosong */
/* F.S. Melakukan alokasi sebuah elemen dan */
```

STEI- ITB	<nomor dokumen>	Halaman 10 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.		

```

/* menambahkan elemen list di akhir: elemen terakhir yang baru */
/* bernilai val jika alokasi berhasil. Jika alokasi gagal: I.S.= F.S. */

void insertAt(List *l, ElType val, int idx);
/* I.S. l tidak mungkin kosong, idx indeks yang valid dalam l, yaitu 0..lengthList(l) */
/* F.S. Melakukan alokasi sebuah elemen dan */
/* menyisipkan elemen dalam list pada indeks ke-idx (bukan menimpa elemen di i) */
/* yang bernilai val jika alokasi berhasil. Jika alokasi gagal: I.S.= F.S. */

/** PENGHAPUSAN ELEMEN **/
void deleteFirst(List *l, ElType *val);
/* I.S. List l tidak kosong */
/* F.S. Elemen pertama list dihapus: nilai info disimpan pada x */
/* dan alamat elemen pertama di-dealokasi */

void deleteLast(List *l, ElType *val);
/* I.S. list tidak kosong */
/* F.S. Elemen terakhir list dihapus: nilai info disimpan pada x */
/* dan alamat elemen terakhir di-dealokasi */

void deleteAt(List *l, int idx, ElType *val);
/* I.S. list tidak kosong, idx indeks yang valid dalam l, yaitu 0..lengthList(l) */
/* F.S. val diset dengan elemen l pada indeks ke-idx. */
/* Elemen l pada indeks ke-idx dihapus dari l */

***** PROSES SEMUA ELEMEN LIST *****
int lengthList(List l);
/* Mengirimkan banyaknya elemen list; mengirimkan 0 jika list kosong */

***** PROSES TERHADAP LIST *****
List concat(List l1, List l2);
/* I.S. l1 dan l2 sembarang */
/* F.S. l1 dan l2 kosong, l3 adalah hasil konkatenasi l1 & l2 */
/* Konkatenasi dua buah list : l1 dan l2 */
/* menghasilkan l3 yang baru (dengan elemen list l1 dan l2 secara berurutan). */
/* Tidak ada alokasi/dealokasi pada prosedur ini */

void swapIdx(List *l, int idx1, int idx2);
/* I.S List L terdefinisi*/
/* F.S Menukar posisi elemen sehingga idx 1 jadi terletak di idx 2 dan sebaliknya*/

boolean isInList(List l, ElType val);
/* Mengembalikan true jika val terdapat adalah salah satu elemen pada List l, false jika tidak */

void printList(List l);
/* I.S. List l terdefinisi */
/* F.S. Setiap elemen list dicetak dengan format "<nomor>. <elemen>\n" */

void clearList(List *l);
/* I.S. List l terdefinisi */
/* F.S. List l menjadi kosong dan semua memory yang digunakan didealokasi */

#endif

```

Linked list cocok untuk pembuatan fitur wishlist dalam program toko karena linked list menyediakan struktur data yang fleksibel dan dinamis. Setiap elemen dalam linked list terhubung

STEI- ITB	<nomor dokumen>	Halaman 11 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.		

melalui pointer, sehingga dapat dengan mudah menelusuri wishlist, menambahkan produk baru di posisi yang diinginkan, atau menghapus produk tanpa mengganggu urutan keseluruhan. Linked list juga efisien dalam penggunaan memori karena hanya mengalokasikan memori yang diperlukan untuk setiap elemen.

4 Program Utama

Agar menjelaskan kode utama ini menjadi lebih mudah, kami akan membaginya jadi beberapa bagian penting. Berikut adalah bagian-bagian penting dari program utama yang kami buat. dalam setiap menu terdapat “*help*” yang dapat membantu user memahami program lebih lanjut.

```
#include <stdio.h>
#include <stdlib.h>
#include "command/store.h"
#include "command/help.h"
#include "command/newload.h"
#include "command/login.h"
#include "command/logout.h"
#include "command/quit.h"
#include "command/register.h"
#include "command/save.h"
#include "command/newstart.h"
#include "command/word13.h"
#include "command/works.h"
#include "command/tebakangka.h"
#include "command/profile.h"
#include "command/optimasirute.h"

int main() {
    ListBarang listbarang;
    Queue queue;
    Works work;
    WorksList worklist;
    Stack stackhistory;
    Map map;
    List wishlist;
    User user;
    ListUser userlist;

    CreateList(&wishlist);
    CreateEmpty(&map);
    CreateEmptyStack(&stackhistory);
    CreateEmptyStack(&user.riwayat_pembelian);
    CreateEmptyWorkList(&worklist);
    LoadWorkList(&worklist);
    CreateListUser(&userlist);
    user.name[0] = '\0';
    InitializeListBarang(&listbarang);
    CreateQueue(&queue);
    printf("Selamat Datang di PURRMART, silahkan masukkan command (START, LOAD, HELP):
\n");

    char input[MAX_WORD_LENGTH];
    int started = 1;
```

```

while (started == 1){
    START();
    StartWords();
    CopyWordToString(input);
    if(string_compare(input, "START") == 0){
        if(NewStartr(&userlist, &listbarang) == 2){
            started = 0;
        }
    }
    else if(string_compare(input, "LOAD") == 0){
        if(NewLoadr(&userlist, &listbarang) == 2){
            started = 0;
        }
    }
    else if(string_compare(input, "HELP") == 0){
        help_welcome();
    }
    else{
        printf("Command tidak dikenal. Masukkan HELP jika anda butuh bantuan.\n");
    }
}

int logged = 1;
printf("Silahkan masukkan command (REGISTER, LOGIN, HELP): \n");
while (logged){
    START();
    StartWords();
    CopyWordToString(input);
    if(string_compare(input, "REGISTER") == 0){
        RegisterUser(&userlist);
    }
    else if(string_compare(input, "LOGIN") == 0){
        if(LoginUser(&userlist, &user) == 1){
            int menu = 1;
            help_main();
            CreateEmpty(&user.keranjang);
            while (menu) {
                printf("Silahkan masukkan command: ");
                START();
                StartWords();
                char inputmain[MAX_WORD_LENGTH];
                CopyWordToString(inputmain);
                if(string_compare(inputmain, "WORK") == 0){
                    UserWorks(&worklist, &user, &userlist);
                }
                else if(string_compare(inputmain, "WORK CHALLENGE") == 0){
                    printf("Daftar challenge yang tersedia:\n");
                    printf("1. Tebak Angka (biaya main = 50)\n");
                    printf("2. W0RDL399 (biaya main = 50)\n");
                    printf("3. Quit Challenge.\n");
                    int inChallenge = 0;
                    while (!inChallenge){
                        printf("\nMasukan challenge yang hendak dimainkan :");
                        char inputChallenge[MAX_LEN];
                        START();
                        StartWords();

```

```

        CopyWordToString(inputChallenge);
        if(string_compare(inputChallenge, "1") == 0){
            ChallengeTebakAngka(&user, &userlist);
            inChallenge = 1;
        }
        else if(string_compare(inputChallenge, "2") == 0){
            PlayWordl3(&user, &userlist);
            inChallenge = 1;
        }
        else if(string_compare(inputChallenge, "3") == 0){
            printf("Anda keluar dari challenge.\n");
            inChallenge = 1;
        }
        else {
            printf("Command tidak dikenal.\n");
        }
    }
}
else if(string_compare(inputmain, "STORE LIST") == 0){
    display_store_list(listbarang);
}
else if(string_compare(inputmain, "STORE REQUEST") == 0){
    store_request(listbarang, &queue);
}
else if(string_compare(inputmain, "STORE SUPPLY") == 0){
    store_supply(&listbarang, &queue);
}
else if(string_compare(inputmain, "STORE REMOVE") == 0){
    store_remove(&listbarang);
}
else if(string_compare(inputmain, "LOGOUT") == 0){
    LogoutUser(&user);
    menu = 0;
}
else if(IsSameFirstWord(inputmain, "SAVE")){
    remainderWord(inputmain, "SAVE");
    printf("%s %d\n", inputmain, strlen(inputmain));
    if(strlen(inputmain) > 0 && string_compare(inputmain, "SAVE") != 0){
        save(&userlist, &listbarang, inputmain);
    }
    else {
        printf("Nama file tidak boleh kosong. SAVE (namafile)\n");
    }
}
else if(string_compare(inputmain, "QUIT") == 0){
    quit(&queue, &userlist, &listbarang);
    return 0;
}
else if(string_compare(inputmain, "HELP") == 0){
    help_main();
}
else if(string_compare(inputmain, "PROFILE") == 0){
    Profile(&user);
}
else if(string_compare(inputmain, "OPTIMASIRUTE") == 0){
    optimasirute();
}

```

```

    }
    else if(IsSameFirstWord(inputmain, "CART")){
        char namaBarang[MAX_WORD_LENGTH];
        remainderWord(inputmain, "CART");
        if(IsSameFirstWord(inputmain, "ADD")){
            remainderWord(inputmain, "ADD");
            if(strlen(inputmain) > 0){
                int quantity = splitStringInt(inputmain, namaBarang);
                if(quantity != -1){
                    CartAdd(&map, listbarang, namaBarang, quantity);
                    AddCartToUser(map, &user);
                }
            }
        }
        else if(IsSameFirstWord(inputmain, "REMOVE")){
            remainderWord(inputmain, "REMOVE");
            if(strlen(inputmain) > 0){
                copyFirstWord(inputmain, namaBarang);
                int quantity = stringToInt(inputmain);
                if(quantity != -1){
                    CartRemove(&map, listbarang, namaBarang, quantity);
                    AddCartToUser(map, &user);
                }
            }
        }
        else if(string_compare(inputmain, "SHOW") == 0){
            DisplayCart(map);
        }
        else if(string_compare(inputmain, "PAY") == 0){
            CartPay(&userlist, &user, &map);
        }
    }
    else if(IsSameFirstWord(inputmain, "HISTORY")){
        remainderWord(inputmain, "HISTORY");
        if(strlen(inputmain) > 0){
            int N = stringToInt(inputmain);
            if(N != -1){
                PrintStackHistory(user.riwayat_pembelian, N);
            }
        }
    }
    else if(IsSameFirstWord(inputmain, "WISHLIST")){
        remainderWordalter(inputmain, "WISHLIST");
        if(IsSameFirstWord(inputmain, "ADD")){
            wishlistAdd(listbarang, &user.wishlist);
            AddWishlistToUser(user.wishlist, &user);
        }
        else if(IsSameFirstWord(inputmain, "SWAP")){
            wishlistSwap(&user.wishlist, inputmain);
            AddWishlistToUser(user.wishlist, &user);
        }
        else if(IsSameFirstWord(inputmain, "REMOVE")){
            if(isThereBlank(inputmain)){
                wishlistRemoveIdx(&user.wishlist, inputmain);
            } else {
                wishlistRemoveName(&user.wishlist);
            }
        }
    }
}

```

```

        AddWishlistToUser(user.wishlist, &user);
    }
    else if(IsSameFirstWord(inputmain, "CLEAR")){
        wishlistClear(&user.wishlist);
        AddWishlistToUser(wishlist, &user);
    }
    else if(IsSameFirstWord(inputmain, "SHOW")){
        wishlistShow(user.wishlist);
    }
}
else {
    printf("Command tidak dikenal.\n");
}
}
else{
    printf("Gagal untuk login.\n");
}
else if(string_compare(input, "HELP") == 0){
    help_login();
}
else{
    printf("Command tidak dikenal. Masukkan HELP jika anda butuh bantuan.\n");
}
}
return 0;
}

```

5 Algoritma-Algoritma Menarik

5.1 swapIdx

```

/* Fungsi pada ADT linked list (linkedlist.h) */

void swapIdx(List *l, int idx1, int idx2) {
    if (isEmptyList(*l) || lengthList(*l) <= 1) {
        return;
    }

    int listlengthList = lengthList(*l);
    if (idx1 < 0 || idx1 >= listlengthList || idx2 < 0 || idx2 >= listlengthList) {
        return;
    }

    if (idx1 == idx2) {
        return;
    }

    Address prev1 = NULL;
    Address curr1 = FIRST(*l);
    for (int i = 0; i < idx1; i++) {

```

```

    prev1 = curr1;
    curr1 = NEXT(curr1);
}

Address prev2 = NULL;
Address curr2 = FIRST(*1);
for (int i = 0; i < idx2; i++) {
    prev2 = curr2;
    curr2 = NEXT(curr2);
}

if (prev1 != NULL) {
    NEXT(prev1) = curr2;
}
else {
    *1 = curr2;
}

if (prev2 != NULL) {
    NEXT(prev2) = curr1;
}
else {
    *1 = curr1;
}

Address temp = NEXT(curr1);
NEXT(curr1) = NEXT(curr2);
NEXT(curr2) = temp;
}

```

Fungsi ini menarik karena in memperlihatkan bahwa prosedur yang terlihat dan terdengar mudah, hanya menukar posisi dari dua buah elemen dalam list. Namun saat dilakukan di bahasa pemrograman menjadi rumit :D.

5.2 optimasirute() & bubblesort

```

void bubblesort(infoo fo[], int ez){
    int tempa, tempb, tempc;
    for(int i=1;i<=ez;i++){
        for(int j=ez;j>=i+1;j--){
            if(fo[j].cost < fo[j-1].cost){
                tempa = fo[j-1].a;
                tempb = fo[j-1].b;
                tempc = fo[j-1].cost;
                fo[j-1].a = fo[j].a;
                fo[j-1].b = fo[j].b;
                fo[j-1].cost = fo[j].cost;
                fo[j].a = tempa;
                fo[j].b = tempb;
                fo[j].cost = tempc;
            }
        }
    }
}

```

```

        }
    }

void optimasirute(){
    int n,e;
    printf("Masukkan jumlah lokasi pengiriman (node): ");
    IgnoreBlanks();
    StartWord();
    n = WordToInt(currentWord);
    printf("Masukkan jumlah rute (edge): ");
    IgnoreBlanks();
    StartWord();
    e = WordToInt(currentWord);
    printf("Masukkan jarak antarlokasi (weight):\n");
    printf("(Format: <Angka Node> <Angka Node> <Cost>)\n");
    int ap,bp,c;
    for(int i=1; i<=e; i++){
        IgnoreBlanks();
        StartWord();
        ap = WordToInt(currentWord);
        IgnoreBlanks();
        StartWord();
        bp = WordToInt(currentWord);
        IgnoreBlanks();
        StartWord();
        c = WordToInt(currentWord);
        datas[i].a = ap;
        datas[i].b = bp;
        datas[i].cost = c;
        points[datas[i].a].l = -1;
        points[datas[i].a].r = -1;
        points[datas[i].b].l = -1;
        points[datas[i].b].r = -1;
        points[datas[i].a].paren = -1;
        points[datas[i].b].paren = -1;
    }

    bubblesort(datas, e);
    /*printf("\n");
    for(int i=1;i<=e;i++){
        printf("%d %d %d\n", datas[i].a, datas[i].b, datas[i].cost);
    }
    printf("\n");*/
    int pickr = n-1;
    int totalcost = 0;
    int curr = 0;
    points[datas[1].a].deg++;
    points[datas[1].b].deg++;
    if(points[datas[1].a].l != -1){
        points[datas[1].a].r = datas[1].b;
    }
}

```

```

        curr = datas[1].a;
    }
    else{
        points[datas[1].a].l = datas[1].b;
        curr = datas[1].a;
    }
    if(points[datas[1].b].l != -1){
        points[datas[1].b].r = datas[1].a;
        curr = datas[1].b;
    }
    else{
        points[datas[1].b].l = datas[1].a;
        curr = datas[1].b;
    }
    points[datas[1].a].paren = datas[1].a;
    points[datas[1].b].paren = datas[1].a;
    int originalparen = datas[1].a;
    totalcost += datas[1].cost;
    pickr--;
    for(int i=2;i<=e;i++){
        if((points[datas[i].a].deg == 2) || (points[datas[i].b].deg == 2)){
            continue;
        }
        if((points[datas[i].a].paren == points[datas[i].b].paren) &&
        (points[datas[i].a].paren != -1) && (points[datas[i].b].paren != -1)){
            continue;
        }
        points[datas[i].a].deg++;
        points[datas[i].b].deg++;
        if(points[datas[i].a].l != -1){
            points[datas[i].a].r = datas[i].b;
            curr = datas[i].a;
        }
        else{
            points[datas[i].a].l = datas[i].b;
            curr = datas[i].a;
        }
        if(points[datas[i].b].l != -1){
            points[datas[i].b].r = datas[i].a;
            curr = datas[i].b;
        }
        else{
            points[datas[i].b].l = datas[i].a;
            curr = datas[i].b;
        }
        totalcost += datas[i].cost;
        pickr--;
        if((points[datas[i].a].paren == originalparen) && (points[datas[i].b].paren ==
        -1)){
            points[datas[i].b].paren = originalparen;
            //no need
        }
    }
}

```

```

    }
    else if((points[datas[i].a].paren == -1) && (points[datas[i].b].paren == originalparen)){
        points[datas[i].a].paren = originalparen;
        //no need
    }
    else if((points[datas[i].a].paren == -1) && (points[datas[i].b].paren == -1)){
        points[datas[i].a].paren = datas[i].a;
        points[datas[i].b].paren = datas[i].a;
        //no need
    }
    else if((points[datas[i].a].paren == originalparen) && (points[datas[i].b].paren != -1)){
        //case division: 6 and 10
        if(datas[i].b != points[datas[i].b].paren){
            points[points[datas[i].b].paren].paren = originalparen;
            points[datas[i].b].paren = originalparen;
        }
        else{ //travers
            int prev1 = datas[i].b;
            int curr1 = -1;
            if(points[datas[i].b].l != -1){
                curr1 = points[datas[i].b].l;
            }
            else{
                curr1 = points[datas[i].b].r;
            }
            int deca = -2;
            while(deca == -2){
                if(points[curr1].r != -1){
                    points[curr1].paren = originalparen;
                    prev1 = curr1;
                    curr1 = points[curr1].r;
                }
                else{
                    points[curr1].paren = originalparen;
                    deca = 2;
                    break;
                }
            }
        }
    }
    else if((points[datas[i].b].paren == originalparen) && (points[datas[i].a].paren != -1)){
        //case division: 6 and 10
        if(datas[i].a != points[datas[i].a].paren){
            points[points[datas[i].a].paren].paren = originalparen;
            points[datas[i].a].paren = originalparen;
        }
        else{ //travers
            int prev1 = datas[i].a;

```

```

int curr1 = -1;
if(points[datas[i].a].l != -1){
    curr1 = points[datas[i].a].l;
}
else{
    curr1 = points[datas[i].a].r;
}
int deca = -2;
while(deca == -2){
    if(points[curr1].r != -1){
        points[curr1].paren = originalparen;
        prev1 = curr1;
        curr1 = points[curr1].r;
    }
    else{
        points[curr1].paren = originalparen;
        deca = 2;
        break;
    }
}
}
else if((points[datas[i].a].paren != -1) && (points[datas[i].b].paren != -1)){
    points[datas[i].a].paren = datas[i].a;
    points[datas[i].b].paren = datas[i].a;
    int currpar = datas[i].a;
    int prev1 = datas[i].b;
    int curr1 = -1;
    if(points[datas[i].b].l != -1){
        curr1 = points[datas[i].b].l;
    }
    else{
        curr1 = points[datas[i].b].r;
    }
    int deca = -2;
    while(deca == -2){
        if(points[curr1].r != -1){
            points[curr1].paren = currpar;
            prev1 = curr1;
            curr1 = points[curr1].r;
        }
        else{
            points[curr1].paren = currpar;
            deca = 2;
            break;
        }
    }
}
//printf("\n %d, %d %d ,%d %d %d \n", i, points[datas[i].a].deg,
points[datas[i].b].deg, points[datas[i].a].l, points[datas[i].a].r,
points[datas[i].b].l, points[datas[i].b].r);

```

```

        if(pickr == 0){
            break;
        }
    }
    int decidr = -1;
    int endr = -1;
    int prev = -1;
    if(points[curr].l != -1){
        prev = curr;
        curr = points[curr].l;
    }
    else{
        endr = curr;
        decidr = 1;
    }
    while(decidr == -1){
        if((points[curr].l !=-1) && (points[curr].r !=-1)){
            if(prev == points[curr].l){
                prev = curr;
                curr = points[curr].r;
            }
            else{
                prev = curr;
                curr = points[curr].l;
            }
        }
        else{
            decidr = 0;
            endr = curr;
        }
    }
    decidr = -1;
    curr = endr;
    prev = curr;
    printf("\nRute Optimal: %d - ", curr);
    if(points[curr].l != -1){
        curr = points[curr].l;
    }
    else{
        curr = points[curr].r;
    }
    while(decidr = -1){
        if((points[curr].l !=-1) && (points[curr].r !=-1)){
            printf("%d - ", curr);
            if(prev == points[curr].l){
                prev = curr;
                curr = points[curr].r;
            }
            else{
                prev = curr;
                curr = points[curr].l;
            }
        }
    }
}

```

```

        }
    }
    else{
        printf("%d", curr);
        break;
    }
}
printf(" Total Rute Cost: %d\n", totalcost);
totalcost = 0;
for(int i=1; i<=n; i++){
    points[i].deg = 0;
    points[i].l = -1;
    points[i].r = -1;
    points[i].paren = -1;
}
for(int i=1; i<=e; i++){
    datas[i].a = 0;
    datas[i].b = 0;
    datas[i].cost = 0;
}
return;
}

```

Bonus optimasi rute ini menggunakan pemikiran Kruskal. Awalnya ketika data node dan edge dimasukkan, langsung di-sort menggunakan algoritma Bubble Sort berdasarkan cost dari edge yang ada, dan diurutkan dari termurah hingga termahal. Kemudian optimasirute() melakukan kalkulasi dengan approach algoritma Kruskal dengan mengambil edge-edge dengan harga yang lebih murah terlebih dahulu. Beberapa kasus di handling, sehingga tidak ada pengambilan set node dengan “parent” yang sama. Di beberapa kasus juga terdapat algoritma mini extra traversal untuk mengganti parent dari satu node-chain secara keseluruhan.

5.3 *splitStringInt*

```

int splitStringInt(char *s1, char *s2) {
    int i = 0;
    int lastSpaceidx = -1;
    int num;

    while (s1[i] != '\0'){
        if (s1[i] == ' '){
            lastSpaceidx = i;
        }
        i++;
    }

    if (lastSpaceidx == -1){
        printf("Tidak ada spasi di dalam string!\n");
        return -1;
    }
}

```

```

char numToSplit[MAX_WORD_LENGTH];
int k = 0;
for (int j = lastSpaceidx + 1; j < i; j++){
    numToSplit[k++] = s1[j];
}
numToSplit[k] = '\0';

if (stringIsNum(numToSplit)){
    num = stringToInt(numToSplit);
    for (int j = 0; j < lastSpaceidx; j++){
        s2[j] = s1[j];
    }
    s2[lastSpaceidx] = '\0';
}
else{
    printf("Input digit angka invalid!\n");
    num = -1;
}

return num;
}

```

Fungsi ini digunakan untuk memisahkan input berupa string yang berakhir dengan angka dan mengembalikan angka terakhir tersebut.

6 Data Test

6.1 Profile

```
Silahkan masukkan command: WISHLIST ADD
Masukkan nama barang: Lalabu
Berhasil menambahkan Lalabu ke wishlist!
Silahkan masukkan command: WISHLIST AK47
Silahkan masukkan command: WISHLIST ADD
Masukkan nama barang: AK47
Berhasil menambahkan AK47 ke wishlist!
Silahkan masukkan command: CART ADD Lalabu 5
Berhasil menambahkan 5 Lalabu ke keranjang belanja!
Silahkan masukkan command: CART ADD Meong 1
Berhasil menambahkan 1 Meong ke keranjang belanja!
Silahkan masukkan command: CART PAY
Berikut adalah isi keranjangmu!
Kuantitas Nama Total
5 Lalabu 100
1 Meong 500
Total biaya yang harus dikeluarkan adalah 600, apakah jadi dibeli? (Ya/Tidak)
Ya
Selamat kamu telah membeli barang-barang tersebut!
Silahkan masukkan command: CART ADD Lalabu 2
Berhasil menambahkan 2 Lalabu ke keranjang belanja!
Silahkan masukkan command: CART ADD AK47 1
Berhasil menambahkan 1 AK47 ke keranjang belanja!
Silahkan masukkan command: CART PAY
Berikut adalah isi keranjangmu!
Kuantitas Nama Total
2 Lalabu 40
1 AK47 10
Total biaya yang harus dikeluarkan adalah 50, apakah jadi dibeli? (Ya/Tidak)
Ya
Selamat kamu telah membeli barang-barang tersebut!
Silahkan masukkan command: CART ADD Meong 2
Berhasil menambahkan 2 Meong ke keranjang belanja!
Silahkan masukkan command: CART ADD AK47 3
Berhasil menambahkan 3 AK47 ke keranjang belanja!
Silahkan masukkan command: PROFILE

>> PROFILE
Nama : Shiori
Saldo : 350

Keranjang Belanja:
Berikut adalah isi keranjangmu!
Kuantitas Nama Total
2 Meong 1000
3 AK47 30

Menunjukkan 5 Riwayat pembelian terakhir
Riwayat Pembelian:
Pembelian 2 - Total 50
Kuantitas Nama Total
2 Lalabu 40
1 AK47 10
```

6.2 Cart

```
Silahkan masukkan command: CART ADD Lalabu 3
Berhasil menambahkan 3 Lalabu ke keranjang belanja!
Silahkan masukkan command: CART ADD AK47 2
Berhasil menambahkan 2 AK47 ke keranjang belanja!
Silahkan masukkan command: CART ADD Meong 1
Berhasil menambahkan 1 Meong ke keranjang belanja!
Silahkan masukkan command: CART SHOW
Berikut adalah isi keranjangmu!
Kuantitas Nama Total
3 Lalabu 60
2 AK47 20
1 Meong 500
Silahkan masukkan command: CART REMOVE AK47 1
Berhasil mengurangi 1 AK47 dari keranjang belanja!
Silahkan masukkan command: CART REMOVE Meong 1
Berhasil mengurangi 1 Meong dari keranjang belanja!
Silahkan masukkan command: CART SHOW
Berikut adalah isi keranjangmu!
Kuantitas Nama Total
3 Lalabu 60
1 AK47 10
Silahkan masukkan command: CART PAY
Berikut adalah isi keranjangmu!
Kuantitas Nama Total
3 Lalabu 60
1 AK47 10
Total biaya yang harus dikeluarkan adalah 70, apakah jadi dibeli? (Ya/Tidak)
Ya
Selamat kamu telah membeli barang-barang tersebut!
Silahkan masukkan command: CART SHOW
Tidak ada Barang apapun di Keranjang.
Silahkan masukkan command: |
```

```
Silahkan masukkan command: CART ADD Meong 10
Berhasil menambahkan 10 Meong ke keranjang belanja!
Silahkan masukkan command: CART SHOW
Berikut adalah isi keranjangmu!
Kuantitas Nama Total
10 Meong 5000
Silahkan masukkan command: CART PAY
Berikut adalah isi keranjangmu!
Kuantitas Nama Total
10 Meong 5000
Total biaya yang harus dikeluarkan adalah 5000, apakah jadi dibeli? (Ya/Tidak)
Ya
Uang kamu hanya 930, tidak cukup untuk membeli barang di keranjang
```

6.3 Wishlist

```
Silahkan masukkan command: WISHLIST ADD
Masukkan nama barang: Lalabu
Berhasil menambahkan Lalabu ke wishlist!
Silahkan masukkan command: WISHLIST ADD
Masukkan nama barang: AK47
Berhasil menambahkan AK47 ke wishlist!
Silahkan masukkan command: WISHLIST ADD
Masukkan nama barang: Meong
Berhasil menambahkan Meong ke wishlist!
Silahkan masukkan command: WISHLIST SHOW
1. Lalabu
2. AK47
3. Meong
Silahkan masukkan command: WISHLIST SWAP 2 3
Berhasil menukar posisi Meong dengan AK47 pada wishlist!
Silahkan masukkan command: WISHLIST SHOW
1. Lalabu
2. Meong
3. AK47
Silahkan masukkan command: WISHLIST SWAP 1 2
Berhasil menukar posisi Meong dengan Lalabu pada wishlist!
Silahkan masukkan command: WISHLIST SHOW
1. Meong
2. Lalabu
3. AK47
Silahkan masukkan command: WISHLIST REMOVE 1
Berhasil menghapus barang posisi ke-1 dari wishlist!
Silahkan masukkan command: WISHLIST SHOW
1. Lalabu
2. AK47
Silahkan masukkan command: WISHLIST RENOVE
Silahkan masukkan command: WISHLIST REMOVE
Masukkan nama barang yang akan dihapus: Lalabu
Lalabu berhasil dihapus dari WISHLIST!
Silahkan masukkan command: WISHLIST SHOW
1. AK47
```

```

Silahkan masukkan command: WISHLIST ADD
Masukkan nama barang: Meong
Berhasil menambahkan Meong ke wishlist!
Silahkan masukkan command: WISHLIST ADD
Masukkan nama barang: Lalabu
Berhasil menambahkan Lalabu ke wishlist!
Silahkan masukkan command: WISHLIST SHOW
1. AK47
2. Meong
3. Lalabu
Silahkan masukkan command: WISHLIST CLEAR
Wishlist telah dikosongkan
Silahkan masukkan command: WISHLIST SHOW
List kamu kosong!
Silahkan masukkan command: |

```

6.4 History

```

Silahkan masukkan command: CART ADD Meong 1
Berhasil menambahkan 1 Meong ke keranjang belanja!
Silahkan masukkan command: CART ADD Lalabu 2
Berhasil menambahkan 2 Lalabu ke keranjang belanja!
Silahkan masukkan command: CART PAY
Berikut adalah isi keranjangmu!
Kuantitas Nama Total
1 Meong 500
2 Lalabu 40
Total biaya yang harus dikeluarkan adalah 540, apakah jadi dibeli? (Ya/Tidak)
Ya
Selamat kamu telah membeli barang-barang tersebut!
Silahkan masukkan command: CART ADD Lalabu 5
Berhasil menambahkan 5 Lalabu ke keranjang belanja!
Silahkan masukkan command: CART ADD AK47 3
Berhasil menambahkan 3 AK47 ke keranjang belanja!
Silahkan masukkan command: CART PAY
Berikut adalah isi keranjangmu!
Kuantitas Nama Total
5 Lalabu 100
3 AK47 30
Total biaya yang harus dikeluarkan adalah 130, apakah jadi dibeli? (Ya/Tidak)
Ya
Selamat kamu telah membeli barang-barang tersebut!
Silahkan masukkan command: HISTORY 3
Riwayat Pembelian:
Pembelian 2 - Total 130
Kuantitas Nama Total
5 Lalabu 100
3 AK47 30

Pembelian 1 - Total 540
Kuantitas Nama Total
1 Meong 500
2 Lalabu 40

Silahkan masukkan command: HISTORY 1
Riwayat Pembelian:
Pembelian 2 - Total 130
Kuantitas Nama Total
5 Lalabu 100
3 AK47 30

```

6.5 Save & Load

6.6 Optimisasi Rute Ekspedisi

```
Silahkan masukkan command: OPTIMASIRUTE
Masukkan jumlah lokasi pengiriman (node): 6
Masukkan jumlah rute (edge): 15
Masukkan jarak antarlokasi (weight):
(Format: <Angka Node> <Angka Node> <Cost>)
0 1 10
0 2 25
0 3 35
0 4 15
0 5 40
1 2 30
1 3 45
1 4 20
1 5 25
2 3 50
2 4 40
2 5 35
3 4 15
3 5 25
4 5 10

Rute Optimal: 3 - 5 - 4 - 0 - 1 - 2, Total Rute Cost: 90
Silahkan masukkan command: █
```

```
Silahkan masukkan command: OPTIMASIRUTE
Masukkan jumlah lokasi pengiriman (node): 4
Masukkan jumlah rute (edge): 5
Masukkan jarak antarlokasi (weight):
(Format: <Angka Node> <Angka Node> <Cost>)
0 1 10
0 2 15
0 3 20
1 2 35
1 3 25

Rute Optimal: 2 - 0 - 1 - 3, Total Rute Cost: 50
Silahkan masukkan command: █
```

7 Test Script

Berikut skrip pengujian yang dilakukan.

STEI- ITB	<nomor dokumen>	Halaman 29 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.		

No.	Fitur yang Dtes	Tujuan Testing	Langkah-Langkah Testing	Input Data Test	Hasil yang Diharapkan	Hasil yang Keluar
1	PROFILE	Mengecek apakah data di user sudah masuk dan ter-update, serta menampilkan beberapa informasi mengenai user.	1. Ketik command "START" di terminal. 2. Ketik command "REGISTER" di terminal. 3. Masukkan username dan password sesuai keinginan. 4. Ketik command "LOGIN" untuk masuk ke dalam purrmart. 5. Ketik command "PROFILE" untuk cek profil user.	Data Test 1	Menampilkan nama, saldo user, keranjang belanja, riwayat pembelian, dan wishlist.	Sesuai ekspektasi
2.	CART	Mengecek apakah user dapat menambahkan barang (CART ADD), membeli barang (CART PAY), menampilkan barang dalam keranjang barang (CART SHOW), menghapus barang pada keranjang barang (CART REMOVE).	1. Ketik "CART ADD <nama barang> <n>" dengan n adalah jumlah barang. Ketik "CART SHOW" untuk mengecek apakah barang sudah ada dalam cart. 2. Ketik "CART REMOVE <name> <n>" dengan n adalah jumlah barang yang ingin dihapus. Ketik "CART SHOW" untuk mengecek apakah barang sudah ada dalam cart. 3. Gunakan command "CART PAY" untuk membayar seluruh isi dari CART, akan mengosongkan cart bila uang user cukup dan tidak akan bisa pay jika uang user tidak cukup untuk mengosongkan cart.	Data Test 2	Barang dapat ditambahkan ke keranjang dan dibeli, serta user dapat melihat dan menghapus barang pada keranjang barang.	Sesuai ekspektasi
3.	WISHLIST	Mengecek apakah user dapat menambah barang di wishlist (WISHLIST ADD), Menghapus	1. Ketik "WISHLIST ADD" kemudian masukkan nama barang (masukkan minimal 2 barang), check apakah berhasil dengan mengetik "WISHLIST SHOW" 2. Gunakan command "WISHLIST REMOVE" tanpa	Data Test 3	Dapat menambah, menghapus, meng-clear, dan menampilkan barang di wishlist	Sesuai ekspektasi

		<p>barang yang ada pada wishlist baik berdasarkan index maupun urutan (WISHLIST REMOVE dan WISHLIST REMOVE <i> <j>), menghapus seluruh barang (WISHLIST CLEAR), menukar posisi barang sesuai urutan (WISHLIST SWAP), dan menampilkan wishlist (WISHLIST SHOW).</p>	<p>spasi, kemudian tulis nama barang yang ingin dihapus dan “WISHLIST REMOVE <i> <j>” dengan i dan j sebagai urutan barang yang ingin dihapus, check apakah berhasil dengan mengetik “WISHLIST SHOW”</p> <p>3. Tambahkan lagi barang kemudian ketik “WISHLIST SWAP <i> <j>” dengan i sama j adalah posisi barang yang ingin ditukar, check apakah berhasil dengan mengetik “WISHLIST SHOW”</p> <p>4. Gunakan command “WISHLIST CLEAR” untuk menghapus wishlist secara keseluruhan, check apakah berhasil dengan mengetik “WISHLIST SHOW”.</p>			
4.	HISTORY & RIWAYAT MAKSIMAL	Memastikan command “HISTORY” Dapat menampilkan pembelian ke-berapa yang dilakukan oleh user beserta barang, harga masing-masing barang, dan harga total.	1. Setelah melakukan beberapa fungsi utama dalam fungsi cart, ketik “HISTORY <n>” dengan n adalah jumlah pembelian terakhir yang ingin ditampilkan dan akan memprioritaskan pembelian terabru bila n kurang dari jumlah pembelian dan akan menampilkan semua bila n lebih besar dari jumlah pembelian.	Data Test 4	Dapat menampilkan info pembelian user dengan rincian pembelian ke-berapa, nama barang, jumlah barang, harga total masing-masing, dan harga total keseluruhan.	Sesuai ekspektasi
5.	SAVE & LOAD			Data Test 5		Sesuai ekspektasi
6.	OPTIMASI RUTE EKSPEDISI		1. Ketika diminta, ketik jumlah node (n) dari rute ekspedisi yang ingin dicari	Data Test 6	Mengirimkan Rute ekspedisi yang paling murah harganya, dengan harga 90 untuk	Sesuai ekspektasi

		<p>2. Ketika diminta, ketik jumlah edge (e) dari rute ekspedisi yang ingin dicari</p> <p>3. Kemudian masukkan data dengan format <Nomor Node> <Nomor Node> <Harga> sebanyak e kali.</p> <p>TC#1:</p> <p>6 15 0 1 10 0 2 25 0 3 35 0 4 15 0 5 40 1 2 30 1 3 45 1 4 20 1 5 25 2 3 50 2 4 40 2 5 35 3 4 15 3 5 25 4 5 10</p> <p>TC#2:</p> <p>0 1 10 0 2 15 0 3 20 1 2 35 1 3 25</p>		TC#1, dan harga 50 untuk TC#2	
--	--	--	--	-------------------------------	--

8 Pembagian Kerja dalam Kelompok

Nama Lengkap - NIM	Deskripsi Tugas
Karunia Mega Lestari - 18221126	
Mahesa Satria Prayata - 18223082	<ul style="list-style-type: none"> - Membuat laporan - Membuat ADT linked list, fitur wishlist - Modifikasi file konfigurasi dan fitur help
Rafli Dwi Nugraha - 18223038	<ul style="list-style-type: none"> - Membuat ADT map - Membuat fitur Cart - Menyesuaikan ADT dengan history - Memperbaiki fungsi dalam user

	<ul style="list-style-type: none"> - Memperbaiki struktur file github - Membuat batch file dan shell script - Merevisi main
M Khalfani Shaquille Indrajaya - 18223104	<ul style="list-style-type: none"> - Membuat fitur history - Membuat ADT stack - Mengerjakan bonus riwayat maksimal - Membuat, debug, dan modifikasi main.c - Membantu debugging fungsi-fungsi lain
Alghan Pridanusuta - 18223058	<ul style="list-style-type: none"> - Membuat profile - Membuat ADT User baru - Membantu membuat main - Membantu revisi ADT - Membantu membuat laporan - Fix bugs
Michael Ballard Isaiah Silaen - 18223080	<ul style="list-style-type: none"> - Merevisi start dan load, dan menyesuaikan dengan bonus Riwayat Maksimal - Mengupdate bagian dari Readme - Modifikasi sedikit dari bagian Main - Solo Bonus Optimasi Rute - Modifikasi file konfigurasi - Spesialis Debugging

9 Lampiran

9.1 Deskripsi Tugas Besar

Agen Purry sedang menikmati tidur siangnya ketika dia tiba-tiba mendengar alarm dari belakang sofa. Suatu pintu rahasia terbuka di bawah dirinya dan ia jatuh ke ruang bawah tanah dan langsung disambut dengan misi terbarunya.



“Ah Agen Purry, maaf harus mengganggu waktu tidur kamu tapi kami mendapatkan laporan bahwa Dr. Asep Spakbor sedang membuat suatu mesin yang dinamakan ‘Oppenheimer-inator’ yang akan menghancurkan wilayah tiga negara bagian. Aku membutuhkan bantuanmu untuk menghentikan Dr. Asep Spakbor, ini merupakan ancaman terbesar yang pernah ia buat.”

And indeed it was. Setelah pertarungan sengit selama 3 bulan 13 hari 2 jam 47 menit dan 2 detik, suplai senjata dan suplai peralatan yang dimiliki OWCA mulai menipis. Harapan kemenangan OWCA mulai memudar...

Tanpa disangka, Agen Purry mengeluarkan senjata rahasia miliknya: menjadi orang Bojongsoang yang memiliki kenalan pegawai Borma. Toko Borma adalah komponen penting yang dapat membawakan kemenangan untuk OWCA pada waktu-waktu kritis ini. Sebab, meskipun Borma terlihat seperti supermarket pada umumnya, mereka sebenarnya merupakan pemasok barang-barang perang. Namun terdapat satu masalah kecil, Borma masih beroperasi secara tatap muka dan OWCA tidak memiliki transport untuk pergi ke Bojongsoang.



Untuk menyelesaikan permasalahan ini, OWCA mengontak tim programmer paling andalnya untuk merancang suatu sistem jual beli ke Borma dengan nama PURRMART! Benar, tim tersebut adalah kalian! Misi ini akan menantang dan menguji kalian. Namun, dengan kerja tim dan tekad yang kuat, kalian pasti dapat menghadapi tantangan ini.

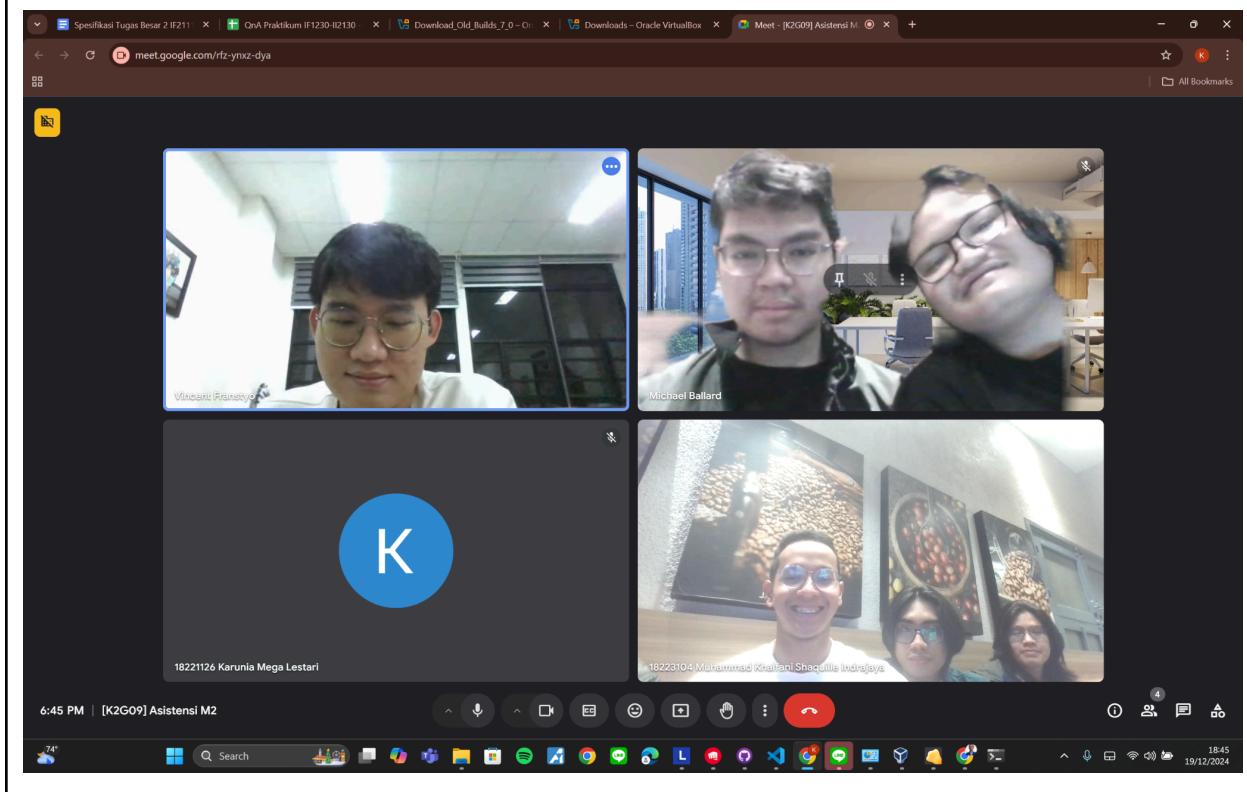
Good luck, programmers! OWCA counts on you! (•`ω•) ♦

9.2 Notulen Rapat

Hari	Jumat	Tanggal	19 Desember 2024
Kelas	02	Kelompok	03
Waktu	18.30 - 19.00	Tempat	Google Meet
Anggota Kelompok	NIM	Nama	
	18221126	Karunia Mega Lestari	
	18223082	Mahesa Satria Prayata	
	18223038	Rafli Dwi Nugraha	
	18223104	M Khalfani Shaquille Indrajaya	
	18223058	Alghan Pridanusuta	
	18223080	Michael Ballard Isaiah Silaen	
Nama Asisten	Vincent Franstyo		

Hasil Asistensi	
No.	Catatan Asisten
1.	<p>Q: Di bonus riwayat maksimal, nampilin pembelian kalau $n >$ jumlah pembelian yang ditampilkan dari pembelian yang paling baru ya?</p> <p>A: kasih error message aja “Tidak bisa melebihi jumlah history yg ada”</p>
2.	Di help jangan lupa tambah command-command baru
3.	Dirapihkan file-file di reponya
4.	Kalau ada perubahan di Milestone 1 direvisi aja

Dokumentasi



9.3 Log Activity Anggota Kelompok

NAMA	LOG ACTIVITY
Alghan Pridanusuta	<p>17 - 20 Desember 2024</p> <ul style="list-style-type: none">- Mengerjakan ADT User update dan profile- Kerja kelompok offline- Revisi beberapa ADT <p>20 - 22 Desember 2024</p> <ul style="list-style-type: none">- Kerja kelompok- Membantu membuat main- Membantu membuat laporan
Mahesa Satria Prayata	<p>17 - 20 Desember 2024</p> <ul style="list-style-type: none">- Mengerjakan ADT linked list dan fitur wishlist

	<p>19 Desember 2024</p> <ul style="list-style-type: none"> - Kerja kelompok offline - Update file konfigurasi dan fitur help - Membuat laporan
Michael Ballard Isaiah Silaen	<p>17-22 Desember 2024</p> <ul style="list-style-type: none"> - Mengerjakan revisi Start dan Load agar sesuai dengan bonus Riwayat Lengkap - Mengerjakan bonus optimasi rute - Mengupdate file_konfig.txt - Membantu debugging di berbagai area berbeda <p>20 Desember 2024</p> <ul style="list-style-type: none"> - Kerja kelompok offline
Rafli Dwi Nugraha	<p>17 - 20 Desember 2024</p> <ul style="list-style-type: none"> - Mengerjakan ADT map dan fitur cart <p>19 Desember 2024</p> <ul style="list-style-type: none"> - Kerja kelompok offline <p>20 Desember 2024</p> <ul style="list-style-type: none"> - Kerja kelompok offline - Debugging fungsi <p>21 Desember 2024</p> <ul style="list-style-type: none"> - Memperbaiki fungsi dalam cart <p>22 Desember 2024</p> <ul style="list-style-type: none"> - Memperbaiki struktur file github - Membuat shell script dan batch file
M Khalfani Shaquille Indrajaya	<p>17-22</p> <ul style="list-style-type: none"> - Mengerjakan fitur cart dan history - Membuat, memodifikasi, dan debug main.c - Membantu debugging <p>20 Desember dan 19 Desember</p> <ul style="list-style-type: none"> - kerja kelompok offline
Karunia Mega Lestari	

DOKUMENTASI

STEI- ITB	<nomor dokumen>	Halaman 37 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.		