**PROGRAM LIST PENGIRIMAN PAKET**

LAPORAN PROYEK AKHIR

MATA KULIAH COMP6362004 – DATA STRUCTURES

KELAS LC20



Oleh:

2602211902 – Ari Danda Widhia Putra

2602126920 – Dimas Sirajuddin Arafa

2602087402– Ezra Aufaa Malefa

Semester Genap 2022/2023

MALANG

**LEMBAR PERSETUJUAN PROYEK AKHIR**

**PROGRAM LIST PENGIRIMAN PAKET**

MATA KULIAH COMP6362004 – DATA STRUCTURES

KELAS LC20

Semester Genap 2022/2023

Laporan akhir proyek ini adalah benar karya kami :

|  |  |  |
| --- | --- | --- |
| **(Ari Danda Widhia Putra)**  **2602211902** | **(Dimas Sirajuddin Arafa)**  **2602126920** | **(Ezra Aufaa Malefa)**  **2602087402** |

**Malang, Sabtu – 17 Juni 2023**

|  |  |  |
| --- | --- | --- |
|  | **(Mirza Ramadhani, S.Kom., M.kom)D6261** |  |

**1. Latar Belakang**

Dalam era teknologi saat ini, pemanfaatan berbagai inovasi dapat secara signifikan meningkatkan efisiensi dan pengalaman pengguna di berbagai sektor, termasuk dalam industri pengiriman paket. Kami memiliki keinginan untuk mengembangkan proyek "List Pengiriman Paket menggunakan Stack" dengan menggunakan bahasa pemrograman C, dengan tujuan untuk meningkatkan pengelolaan dan pengiriman paket secara efisien.

Pada kasus ini, kami mengasumsikan bahwa terdapat kebutuhan untuk mengelola pengiriman paket dalam suatu perusahaan atau layanan kurir. Penggunaan aplikasi yang dirancang oleh kami akan menjadi solusi sederhana untuk mengatur dan mencatat daftar pengiriman paket yang dilakukan setiap harinya. Dengan menggunakan pendekatan stack dalam struktur data, kami dapat memanfaatkan keuntungan dari prinsip Last-In-First-Out (LIFO) untuk mengelola daftar pengiriman paket dengan lebih efisien.

Dalam pengembangan program stack ini, kami memilih bahasa pemrograman C karena bahasa ini memiliki keunggulan dalam performa dan pengelolaan memori yang efisien. Dengan menerapkan struktur data stack pada proyek ini, kami dapat menyimpan informasi pengiriman paket dalam urutan yang tepat, memungkinkan kami untuk dengan mudah menambah dan menghapus elemen-elemen dalam daftar pengiriman paket.

Melalui proyek ini, kami berharap dapat meningkatkan efisiensi dalam pengelolaan dan pengiriman paket, mengoptimalkan operasional perusahaan atau layanan kurir, serta meningkatkan pengalaman pengguna dalam mengikuti status pengiriman paket mereka. Dengan adanya aplikasi ini, diharapkan dapat tercapai peningkatan kinerja secara keseluruhan dan meningkatkan kepuasan pelanggan.

**2. Tinjauan Pustaka**

**Stack**

Stack adalah struktur data linear yang mengikuti prinsip "last in, first out" (LIFO). Dalam stack, elemen-elemen disimpan dan diakses sesuai dengan urutan penambahan, di mana elemen terakhir yang ditambahkan adalah yang pertama kali dihapus. Istilah "stack" sendiri mengacu pada tumpukan objek fisik yang dapat kita temui sehari-hari, seperti tumpukan buku di atas meja.

Beberapa fungsi umum yang sering digunakan dalam stack adalah:

* Push: Operasi push digunakan untuk menambahkan elemen baru ke dalam stack. Elemen ini akan ditempatkan di bagian paling atas (top) dari stack. Ketika push dilakukan, ukuran stack akan bertambah. Elemen yang ditambahkan dengan push akan menjadi elemen teratas baru dalam stack, menggantikan elemen sebelumnya.
* Pop: Operasi pop digunakan untuk menghapus elemen teratas (top) dari stack. Saat pop dilakukan, elemen teratas akan dihapus dari stack, dan stack akan mengambil elemen di bawahnya sebagai elemen teratas baru. Dengan demikian, ukuran stack akan berkurang. Elemen yang dihapus dengan pop biasanya adalah elemen terakhir yang ditambahkan ke dalam stack.
* Peek atau Top: Fungsi ini digunakan untuk melihat nilai dari elemen teratas (top) dalam stack tanpa menghapusnya. Hal ini berguna untuk mengakses nilai teratas dalam stack tanpa memodifikasinya.
* isEmpty: Fungsi ini digunakan untuk memeriksa apakah stack kosong atau tidak. Jika stack kosong, fungsi ini mengembalikan nilai true; jika tidak, mengembalikan nilai false.
* isFull: Fungsi ini digunakan untuk memeriksa apakah stack penuh atau tidak. Beberapa implementasi stack memiliki batasan ukuran, dan fungsi ini digunakan untuk memeriksa apakah stack telah mencapai kapasitas maksimum.
* Size: Fungsi ini digunakan untuk mengembalikan jumlah elemen yang ada dalam stack saat ini.

Fungsi-fungsi ini memungkinkan kita untuk memanipulasi dan mengelola elemen-elemen dalam stack dengan cara yang sesuai dengan prinsip LIFO. Stack digunakan dalam berbagai aplikasi seperti evaluasi ekspresi matematika, implementasi rekursi, penanganan pemanggilan fungsi, pemrosesan undo/redo, dan banyak lagi.

**3. Gambaran Umum Program**

**A. Pseudocode and Flowchart**

**Psudeocode**

Deklarasi tipe data:

- Akun (struct):

- username (string)

- password (string)

- AkunStack (struct):

- data (array of Akun)

- top (integer)

- Paket (struct):

- pengirim (string)

- alamat\_pengirim (string)

- penerima (string)

- alamat\_penerima (string)

- nomor\_penerima (string)

- berat (float)

- status (string)

- Stack (struct):

- data (array of Paket)

- top (integer)

Deklarasi fungsi:

- initializeAkunStack(akunStack) - inisialisasi AkunStack

- isAkunStackFull(akunStack) - cek apakah AkunStack penuh

- isAkunStackEmpty(akunStack) - cek apakah AkunStack kosong

- saveAkunStackToFile(akunStack) - simpan data AkunStack ke file

- loadPaketStackFromFile(akunStack) – muat data AkunStack dari file

- pushAkun(akunStack, newAkun) - tambahkan Akun baru ke AkunStack

- popAkun(akunStack) - ambil Akun teratas dari AkunStack

- searchAkun(akunStack, akunUsername) - cari Akun berdasarkan username

- displayAkun(akunStack) – tampilkan daftar akun dalam Stack

- initialize(paketStack) - inisialisasi Stack

- isFull(paketStack) - cek apakah Stack penuh

- isEmpty(paketStack) - cek apakah Stack kosong

- savePaketStackToFile(paketStack) - simpan data Stack ke file

- loadPaketStackFromFile(paketStack) - muat data Stack dari file

- push(paketStack, newPaket) - tambahkan Paket baru ke Stack

- pop(paketStack) - ambil Paket teratas dari Stack

- displayPaket(paketStack) - tampilkan daftar Paket dalam Stack

- searchPaket(paketStack, namaPengirim) - cari Paket berdasarkan nama pengirim

Deklarasi variabel:

- akunStack (AkunStack)

- paketStack (Stack)

- username (string)

- password (string)

- loginSuccess (boolean)

- choice (integer)

- paketChoice (integer)

- akunChoice (integer)

Int main

Inisialisasi akunStack

Inisialisasi paketStack

loadAkunStack

loadPaketStack

Selama loginSuccess bernilai salah:

Tampilkan pesan "Enter username: "

Baca input username

Tampilkan pesan "Enter password: "

Baca input password

Untuk setiap akun dalam akunStack:

Jika username dan password yang dimasukkan sama dengan akun saat ini dalam akunStack:

Tampilkan pesan "Login successful"

Set loginSuccess menjadi benar

Hentikan pengulangan

Jika loginSuccess masih salah, tampilkan pesan "Invalid username or password"

Selama choice tidak sama dengan 0:

Tampilkan menu pilihan

Baca input choice

Jika choice sama dengan 1:

Tampilkan menu pilihan paket

Baca input paketChoice

Jika paketChoice sama dengan 1:

Tampilkan pesan "Enter sender's name: "

Baca input pengirim

Tampilkan pesan "Enter sender's address: "

Baca input alamat\_pengirim

Tampilkan pesan "Enter recipient's name: "

Baca input penerima

Tampilkan pesan "Enter recipient's address: "

Baca input alamat\_penerima

Tampilkan pesan "Enter recipient's phone number: "

Baca input nomor\_penerima

Tampilkan pesan "Enter weight (in kg): "

Baca input berat

Tampilkan pesan "Enter status: "

Baca input status

Buat objek paketBaru dengan data yang telah dimasukkan

Push paketBaru ke dalam paketStack

Tampilkan pesan "Package added successfully"

Jika paketChoice sama dengan 2:

Jika paketStack kosong:

Tampilkan pesan "No packages available"

Jika tidak:

Tampilkan daftar paket dalam paketStack menggunakan fungsi displayPaket

Jika paketChoice sama dengan 3:

Pop paketStack

Jika paketChoice sama dengan 4:

Jika paketStack kosong:

Tampilkan pesan "No packages available"

Jika tidak:

Tampilkan pesan "Enter sender's name to search: "

Baca input namaPengirim

Cari paket berdasarkan nama pengirim menggunakan fungsi searchPaket

Jika choice sama dengan 2:

Tampilkan menu pilihan akun

Baca input akunChoice

Jika akunChoice sama dengan 1:

Tampilkan pesan "Enter username: "

Baca input username

Tampilkan pesan "Enter password: "

Baca input password

Buat objek akunBaru dengan data yang telah dimasukkan

Push akunBaru ke dalam akunStack

Simpan akunStack ke dalam file menggunakan fungsi saveAkunStackToFile

Tampilkan pesan "Account created successfully"

Jika akunChoice sama dengan 2:

Jika akunStack kosong:

Tampilkan pesan "No accounts available"

Jika tidak:

Tampilkan semua akun dalam akunStack

Jika akunChoice sama dengan 3:

Pop akunStack

Jika akunChoice sama dengan 4:

Tampilkan pesan "Enter username to search: "

Baca input username

Cari akun berdasarkan username menggunakan fungsi searchAkun

Jika choice sama dengan 0:

Tampilkan pesan "Exiting program..."

Selesai

**Flowchart**

Sebuah gambar berisi diagram, teks, Rencana, skematis

Deskripsi dibuat secara otomatis

**B. Program Overview**

Menu Kelola Paket:

* Tambah Paket: Menu ini memungkinkan pengguna untuk menambahkan informasi tentang sebuah paket, termasuk nama pengirim, alamat pengirim, nama penerima, alamat penerima, nomor penerima, dan berat paket. Setelah informasi paket ditambahkan, paket akan disimpan dalam sebuah stack.
* Hapus Paket: Menu ini memungkinkan pengguna untuk mengambil paket teratas dari stack. Informasi tentang paket yang diambil akan ditampilkan setelah pengambilan.
* Display Paket: Menu ini menampilkan daftar semua paket yang ada dalam stack beserta informasi rinci seperti nomor paket, nama pengirim, alamat pengirim, nama penerima, dan alamat penerima.
* Search Paket: Menu ini memungkinkan pengguna untuk mencari paket berdasarkan nama pengirim. Hasil pencarian akan menampilkan daftar paket yang sesuai dengan nama pengirim yang dimasukkan.

Menu Kelola Akun:

* Tambah Akun: Menu ini memungkinkan pengguna untuk menambahkan akun baru dengan memasukkan username dan password. Akun yang ditambahkan akan disimpan dalam sebuah stack.
* Display Akun : Menu ini menampilkan daftar semua akun yang ada dalam stack dan hanya menampilkan usernamenya saja.
* Delete Akun : Menu ini memungkinkan pengguna untuk mengambil stack paling teratas dari stack. Informasi akun yang diambil akan ditampilkan setelah pengambilan.
* Search Akun: Menu ini memungkinkan pengguna untuk mencari akun berdasarkan username. Hasil pencarian akan menampilkan daftar akun yang sesuai dengan username yang dimasukkan.

Kedua menu tersebut dapat diakses oleh pengguna setelah melakukan login dengan memasukkan username dan password yang benar. Program juga menyediakan kemampuan untuk menyimpan dan memuat data stack ke dalam file. Data akun disimpan dalam file "akun.txt", sementara data paket disimpan dalam file "paket.txt".

**4. Hasil**

1. **Program Screenshot**

a) Tampilan Awal

Terdapat admin login pada tampilan awal, setelah berhasil login terdapat 3 pilihan di dalam menu. Berikut adalah screenshot tampilan awal ketika aplikasi pertama kali dibuka:

Sebuah gambar berisi teks, Font, cuplikan layar

Deskripsi dibuat secara otomatis

b) Memilih Menu Pertama (Kelola Paket)

Terdapat menu kelola paket yang berisikan 4 pilihan utama dan 1 pilihan untuk kembali ke menu utama. Berikut adalah screenshot tampilan menu kelola paket.

Sebuah gambar berisi teks, Font, cuplikan layar, tipografi

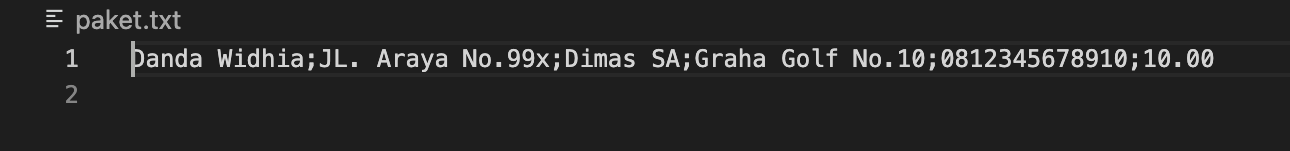
Deskripsi dibuat secara otomatis

* Memilih Pilihan Pertama pada menu kelola paket

Admin akan diminta untuk memasukkan nama pengirim, alamat pengirim, nama penerima, alamat penerima, no telp penerima dan berat paket.

Sebuah gambar berisi teks, Font, cuplikan layar

Deskripsi dibuat secara otomatis

Setelah berhasil mengisi data yang diminta, maka data akan disimpan kedalam file paket.txt

* Memilih pilihan kedua pada menu kelola paket

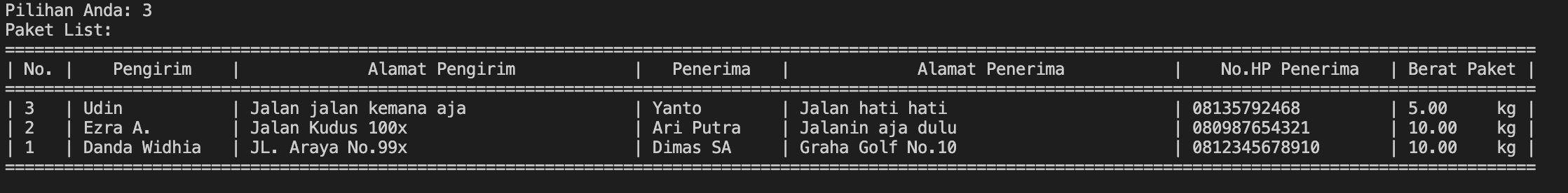
Admin dapat menghapus data yang paling atas dalam stack. Setelah berhasil dihapus, data didalam file txt juga akan ikut terhapus.

Sebuah gambar berisi teks, Font, cuplikan layar

Deskripsi dibuat secara otomatis

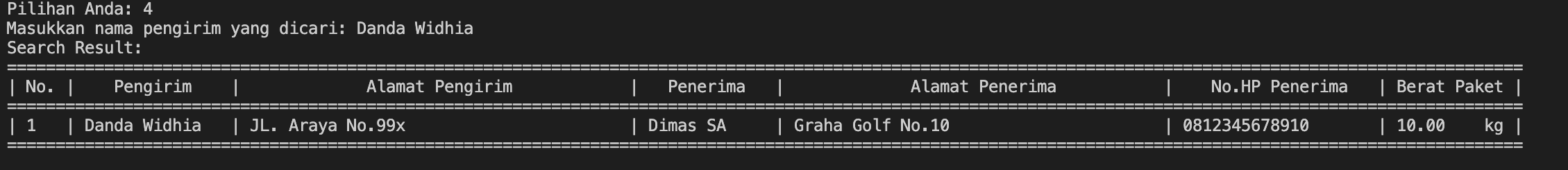
* Memilih pilihan ketiga pada meu kelola paket

Admin dapat melihat display keseluruhan data yang sudah diinput, baik itu No, Nama pengirim, Alamat pengirim, Nama penerima, Alamat penerima, No HP penerima, Berat paket. Sebagai contoh kami sudah menginput 3 data paket yang berbeda – beda.



* Memilih pilihan keempat pada menu kelola paket

Admin dapat mencari paket pada stack dengan menginput nama pengirim yang akan dicari, maka hanya data dari nama pengirim yang dicarilah muncul.



* Memilih pilihan “0” pada menu kelola paket

Jika admin memilih “0”, maka program akan kembali ke menu utama yang terdiri dari kelola paket dan kelola akun.

c) Memilih Menu Kedua (Kelola Akun)

Terdapat menu kelola akun yang memiliki 4 pilihan utama dan 1 pilihan untuk kembali ke menu utama. Berikut adalah tampilan screenshotnya.

Sebuah gambar berisi teks, Font, cuplikan layar, hitam

Deskripsi dibuat secara otomatis

* Memilih pilihan pertama pada menu kelola akun

Admin diarahkan untuk menginput data username dan password baru yang nantinya akan disimpan ke dalam akun.txt. Data ini dapat digunakan ketika login pada awal program dimulai

Sebuah gambar berisi teks, Font, cuplikan layar, tipografi

Deskripsi dibuat secara otomatis

* Memilih pilihan kedua pada menu kelola akun

Admin dapat melihat display kesulurahn dari username yang terdaftar dalam akun.txt. Pada menu ini hanya akan ditampilkan usernaemnya saja.

Sebuah gambar berisi teks, Font, cuplikan layar, diagram

Deskripsi dibuat secara otomatis

* Memilih pilihan ketiga pada menu kelola akun

Admin dapat menghapus data akun paling atas yang terdapat pada stack. Setelah berhasil menghapus, maka data akun yang di dalam file txt juga ikut terhapus.

Sebuah gambar berisi teks, Font, cuplikan layar, Grafis

Deskripsi dibuat secara otomatis

* Memilih pilihan keempat pada menu kelola akun

Admin dapat mencari username akun dengan kata kunci usernamenya guna mengetahui apakah akun terdaftar dalam file txt.

Sebuah gambar berisi teks, cuplikan layar, Font

Deskripsi dibuat secara otomatis

* Memilih pilihan “0” pada menu kelola akun

Jika admin memilih “0”, maka program akan kembali ke menu utama yang terdiri dari kelola paket dan kelola akun.

d) Memilih Menu Ketiga (Keluar)

Setelah program ini digunakan, admin dapat mengakhiri program ini dengan memilih menu 0 atau menu keluar.

Sebuah gambar berisi teks, Font, cuplikan layar

Deskripsi dibuat secara otomatis

1. **Source Code**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <stdbool.h>

typedef struct

{

char username[50];

char password[50];

} Akun;

typedef struct

{

Akun data[100];

int top;

} AkunStack;

void initializeAkunStack(AkunStack \*akunStack)

{

akunStack->top = -1;

}

int isAkunStackFull(AkunStack \*akunStack)

{

return akunStack->top == 99;

}

int isAkunStackEmpty(AkunStack \*akunStack)

{

return akunStack->top == -1;

}

void loadAkunStackFromFile (AkunStack \*akunStack);

void saveAkunStackToFile(AkunStack \*akunStack);

void pushAkun(AkunStack \*akunStack, Akun newAkun)

{

if (isAkunStackFull(akunStack))

{

printf("Akun stack is full\n");

}

else

{

akunStack->top++;

akunStack->data[akunStack->top] = newAkun;

}

}

Akun popAkun(AkunStack \*akunStack)

{

Akun emptyAkun = { "", "" };

if (isAkunStackEmpty(akunStack))

{

printf("Akun stack is empty\n");

return emptyAkun;

}

else

{

Akun poppedAkun = akunStack->data[akunStack->top];

akunStack->top--;

return poppedAkun;

saveAkunStackToFile(&akunStack);

}

}

void displayAkun(AkunStack \*akunStack)

{

if (isAkunStackEmpty(akunStack))

{

printf("Akun stack is empty\n");

}

else

{

printf("Akun List\n");

printf("======================\n");

printf("| No. | Username |\n");

printf("======================\n");

for (int i = akunStack->top; i >= 0; i--)

{

printf("| %-3d | %-12s |\n", i+1, akunStack->data[i].username);

}

printf("======================\n");

}

}

int searchAkun(AkunStack \*akunStack, char \*akunUsername)

{

int found = 0;

printf("Search Result\n");

printf("======================\n");

printf("| No. | Username |\n");

printf("======================\n");

for (int i = akunStack->top; i >= 0; i--)

{

if (strcmp(akunStack->data[i].username, akunUsername) == 0)

{

printf("| %-3d | %-12s |\n", i+1, akunStack->data[i].username);

found = 1;

}

}

printf("======================\n");

if (!found)

{

printf("No akun found for username: %s\n", akunUsername);

}

return found;

}

typedef struct

{

char pengirim[50];

char alamat\_pengirim[100];

char penerima[50];

char alamat\_penerima[100];

char nomor\_penerima[50];

float berat;

} Paket;

typedef struct

{

Paket data[100];

int top;

} Stack;

void initialize(Stack \*paketStack)

{

paketStack->top = -1;

}

int isFull(Stack \*paketStack)

{

return paketStack->top == 99;

}

int isEmpty(Stack \*paketStack)

{

return paketStack->top == -1;

}

void savePaketStackToFile(Stack \*paketStack);

void loadPaketStackFromFile(Stack \*paketStack);

void push(Stack \*paketStack, Paket newPaket)

{

if (isFull(paketStack))

{

printf("Stack is full\n");

} else

{

paketStack->top++;

paketStack->data[paketStack->top] = newPaket;

}

}

Paket pop(Stack \*paketStack)

{

Paket poppedPaket;

if (isEmpty(paketStack))

{

printf("Stack is empty\n");

strcpy(poppedPaket.pengirim, "");

} else

{

poppedPaket = paketStack->data[paketStack->top];

paketStack->top--;

savePaketStackToFile(paketStack);

}

return poppedPaket;

}

void displayPaket(Stack \*paketStack)

{

if (isEmpty(paketStack))

{

printf("No paket\n");

} else

{

printf("Paket List:\n");

printf("============================================================================================================================================================\n");

printf("| No. | Pengirim | Alamat Pengirim | Penerima | Alamat Penerima | No.HP Penerima | Berat Paket |\n");

printf("============================================================================================================================================================\n");

for (int i = paketStack->top; i >= 0; i--)

{

printf("| %-3d | %-14s | %-38s | %-12s | %-37s | %-19s | %-8.2f kg |\n", i + 1, paketStack->data[i].pengirim, paketStack->data[i].alamat\_pengirim, paketStack->data[i].penerima, paketStack->data[i].alamat\_penerima, paketStack->data[i].nomor\_penerima, paketStack->data[i].berat);

}

printf("============================================================================================================================================================\n");

}

}

void searchPaket(Stack \*paketStack, char \*namaPengirim)

{

int found = 0;

for (int i = paketStack->top; i >= 0; i--)

{

if (strcmp(paketStack->data[i].pengirim, namaPengirim) == 0)

{

printf("Search Result:\n");

printf("============================================================================================================================================================\n");

printf("| No. | Pengirim | Alamat Pengirim | Penerima | Alamat Penerima | No.HP Penerima | Berat Paket |\n");

printf("============================================================================================================================================================\n");

printf("| %-3d | %-14s | %-38s | %-12s | %-37s | %-19s | %-8.2f kg |\n", i + 1, paketStack->data[i].pengirim, paketStack->data[i].alamat\_pengirim, paketStack->data[i].penerima, paketStack->data[i].alamat\_penerima, paketStack->data[i].nomor\_penerima, paketStack->data[i].berat);

found = 1;

}

}

printf("============================================================================================================================================================\n");

if (!found)

{

printf("No paket found for sender: %s\n", namaPengirim);

}

}

int main()

{

AkunStack akunStack;

initializeAkunStack(&akunStack);

loadAkunStackFromFile(&akunStack);

char username[50];

char password[50];

int loginSuccess = 0;

while (!loginSuccess)

{

printf("Enter username: ");

scanf("%s", username);

printf("Enter password: ");

scanf("%s", password);

for (int i = 0; i <= akunStack.top; i++)

{

if (strcmp(akunStack.data[i].username, username) == 0 && strcmp(akunStack.data[i].password, password) == 0)

{

printf("\nLogin successful!\n");

loginSuccess = 1;

break;

}

}

if (!loginSuccess)

{

printf("Invalid username or password\n");

}

}

Stack paketStack;

initialize(&paketStack);

loadPaketStackFromFile(&paketStack);

int choice;

do

{

printf("\n");

printf("========== MENU ==========\n");

printf("1. Kelola Paket\n");

printf("2. Kelola Akun\n");

printf("0. Keluar\n");

printf("===========================\n");

printf("Pilih menu: ");

scanf("%d", &choice);

switch (choice)

{

case 1:

{

int paketChoice;

do

{

printf("\nMenu Kelola Paket:\n");

printf("1. Tambah Paket\n");

printf("2. Hapus Paket\n");

printf("3. Display Paket\n");

printf("4. Search Paket\n");

printf("0. Kembali\n");

printf("Pilihan Anda: ");

scanf("%d", &paketChoice);

getchar();

switch (paketChoice)

{

case 1:

{

Paket newPaket;

printf("Masukkan nama pengirim: ");

scanf("%[^\n]", newPaket.pengirim);

getchar();

printf("Masukkan alamat pengirim: ");

scanf("%[^\n]", newPaket.alamat\_pengirim);

getchar();

printf("Masukkan nama penerima: ");

scanf("%[^\n]", newPaket.penerima);

getchar();

printf("Masukkan alamat penerima: ");

scanf("%[^\n]", newPaket.alamat\_penerima);

getchar();

printf("Masukkan nomor penerima: ");

scanf("%[^\n]", newPaket.nomor\_penerima);

getchar();

printf("Masukkan berat paket (kg): ");

scanf("%f", &newPaket.berat);

push(&paketStack, newPaket);

savePaketStackToFile(&paketStack);

printf("Paket berhasil ditambahkan\n");

break;

}

case 2:

{

Paket poppedPaket = pop(&paketStack);

if (strcmp(poppedPaket.pengirim, "") != 0)

{

printf("\nPaket berhasil diambil\n");

printf("Pengirim : %s\n", poppedPaket.pengirim);

printf("Alamat Pengirim : %s\n", poppedPaket.alamat\_pengirim);

printf("Penerima : %s\n", poppedPaket.penerima);

printf("Alamat Penerima : %s\n", poppedPaket.alamat\_penerima);

printf("Nomor Penerima : %s\n", poppedPaket.nomor\_penerima);

printf("Berat Paket : %.2f kg\n\n", poppedPaket.berat);

}

break;

}

case 3:

displayPaket(&paketStack);

break;

case 4:

{

char namaPengirim[50];

printf("Masukkan nama pengirim yang dicari: ");

scanf("%[^\n]", namaPengirim);

searchPaket(&paketStack, namaPengirim);

break;

}

case 0:

printf("Kembali ke menu utama\n");

break;

default:

printf("Pilihan tidak valid\n");

break;

}

} while (paketChoice != 0);

break;

}

case 2:

{

int akunChoice;

do

{

printf("\n");

printf("Menu Kelola Akun:\n");

printf("1. Tambah Akun\n");

printf("2. Display Akun\n");

printf("3. Delete Akun\n");

printf("4. Search Akun\n");

printf("0. Kembali\n");

printf("Pilihan Anda: ");

scanf("%d", &akunChoice);

switch (akunChoice)

{

case 1:

{

Akun newAkun;

printf("Masukkan username: ");

scanf("%s", newAkun.username);

getchar();

printf("Masukkan password: ");

scanf("%s", newAkun.password);

getchar();

pushAkun(&akunStack, newAkun);

saveAkunStackToFile(&akunStack);

printf ("Akun berhasil ditambah\n");

break;

}

case 2:

{

displayAkun(&akunStack);

break;

}

case 3 :

{

Akun poppedAkun = popAkun(&akunStack);

printf("Username akun '%s' telah dihapus\n", poppedAkun.username);

break;

}

case 4:

{

char akunUsername[50];

printf("Masukkan username yang dicari: ");

scanf("%s", akunUsername);

searchAkun(&akunStack, &akunUsername);

break;

}

case 0:

printf("Terima kasih, program selesai.\n");

break;

default:

printf("Pilihan tidak valid\n");

}

} while (akunChoice != 0);

break;

}

case 0:

printf("\nTerima kasih !!!\n\n");

printf("Program telah berakhir.\n");

break;

default:

printf("Pilihan menu tidak valid\n");

break;

}

} while (choice != 0);

return 0;

}

/\* Function - function Untuk file proccesing \*/

void saveAkunStackToFile(AkunStack \*akunStack) {

FILE \*file = fopen("akun.txt", "w");

if (file == NULL) {

printf("Error opening file\n");

return;

}

for (int i = 0; i <= akunStack->top; i++) {

fprintf(file, "%s %s\n", akunStack->data[i].username,

akunStack->data[i].password);

}

fclose(file);

}

void savePaketStackToFile(Stack \*paketStack)

{

FILE \*file = fopen("paket.txt", "w");

if (file == NULL)

{

printf("Error opening file\n");

return;

}

for (int i = 0; i <= paketStack->top; i++)

{

fprintf(file, "%s;%s;%s;%s;%s;%.2f\n", paketStack->data[i].pengirim,

paketStack->data[i].alamat\_pengirim,

paketStack->data[i].penerima,

paketStack->data[i].alamat\_penerima,

paketStack->data[i].nomor\_penerima,

paketStack->data[i].berat);

}

fclose(file);

}

void loadPaketStackFromFile(Stack \*paketStack)

{

FILE \*file = fopen("paket.txt", "r");

if (file == NULL)

{

printf("Error opening file\n");

return;

}

paketStack->top = -1;

while (fscanf(file, "%[^;];%[^;];%[^;];%[^;];%[^;];%f\n", paketStack->data[paketStack->top + 1].pengirim,

paketStack->data[paketStack->top + 1].alamat\_pengirim,

paketStack->data[paketStack->top + 1].penerima,

paketStack->data[paketStack->top + 1].alamat\_penerima,

paketStack->data[paketStack->top + 1].nomor\_penerima,

&paketStack->data[paketStack->top + 1].berat) != EOF)

{

paketStack->top++;

}

fclose(file);

}

void loadAkunStackFromFile (AkunStack \*akunStack)

{

FILE \*file;

file = fopen("akun.txt", "r");

if (file == NULL)

{

printf("Error opening file\n");

}

while (fscanf(file, "%s %s\n", akunStack->data[akunStack->top + 1].username,

akunStack->data[akunStack->top + 1].password) != EOF)

{

akunStack->top++;

}

fclose(file);

}

**5. Referensi**

<https://www.educative.io/answers/how-to-implement-a-stack-in-c-using-an-array>

<https://www.markijar.com/2015/05/stack-tumpukan-dalam-bahasa-c.html#:~:text=Stack%20merupakan%20jenis%20linked%20list,terakhir%20akan%20diproses%20%2F%20dikeluarkan%20pertama>.

<https://www.geeksforgeeks.org/introduction-to-stack-data-structure-and-algorithm-tutorials/>

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| LEMBAR PENILAIAN      **PROGRAM LIST PENGIRIMAN PAKET**    MATA KULIAH COMP6362004 – DATA STRUCTURES  KELAS LC20  Semester Genap 2022/2023     |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | **DAFTAR MAHASISWA** |  | **NILAI** | |  | **BOBOT** | | | |  | **KREDIT** | |  | **TOTAL KREDIT** | | **1** | **2** | **3** | **4** | **1** | **2** | **3** | 4 | 1 | 2 | 3 | 4 | | <NIM 01> - <NAMA MAHASISWA 01> |  |  |  |  | 20% | 30% | 30% | 20% |  |  |  |  |  | | <NIM 02> - <NAMA MAHASISWA 02> |  |  |  |  | 20% | 30% | 30% | 20% |  |  |  |  |  | | <NIM 03> - <NAMA MAHASISWA 03> |  |  |  |  | 20% | 30% | 30% | 20% |  |  |  |  |  | |  |  | **TOTAL** | |  |  | | | |  |  | |  |  |   **KETERANGAN :**   * **Skala Penilaian : 0 sd 100** * **Komponen**    1. : Laporan   2. : Produk   3. : Pengetahuan dan Solusi   4. : Presentasi     **Malang, Sabtu – 17 Juni 2023**          **(Mirza Ramadhani, S.Kom., M.kom)D6261** |

**Pembagian Tugas**

- Danda : - Membuat menu display paket

- Membuat menu tambah paket

- Membuat menu hapus paket

- Mmebuat file processing paket.txt dan akun.txt

- Membuat laporan

- Dimas : - Membuat menu search paket

- Membuat menu search akun

- Fix bug and maintenance

- Merapikan coding

- Membuat laporan

- Ezra : - Membuat menu tambah akun

- Membuat menu hapus akun

- Mmebuat menu display akun

- Melakukan pengetesan aplikasi guna memastikan code berjalan

- Mmebuat laporan