**GROCERY STORE**

LAPORAN STUDI KASUS

MATA KULIAH COMP6362004 – DATA STRUCTURES

KELAS BB-20

A screenshot of a video game

Description automatically generated with medium confidence

Oleh:

Monica Wahyu Sayyidina - 2602187481

Rahel Andrea Arabella - 2602085920

Nabila Shifa Putri Farhanah - 2602183584

Semester Genap 2022/2023

MALANG

**LEMBAR PERSETUJUAN PROYEK AKHIR**

**GROCERY STORE WAREHOUSE**

MATA KULIAH COMP6362004 – DATA STRUCTURES

KELAS BB20

Semester Genap 2022/2023

Laporan akhir proyek ini adalah benar karya kami :

**MONICA WAHYU S. RAHEL ANDREA A. NABILA SHIFA P. F.**

**2602187481 2602085920 2602183584**

**Malang, 18 Juni 2023**

**NYOMAN WIRA PRASETYA, S. Kom., M.T., M.Sc.**

**D6494**

1. **BACKGROUND**
   1. **Team’s Planning**

Team kami berencana untuk membuat suatu gudang dari toko grosir yaitu toko sembako. Pada program toko grosir ini, kami mengimplementasikannya menggunakan metode Binary Search Tree (BST). Struktur data BST digunakan untuk menyimpan dan mengelola informasi produk yang tersedia di toko grosir. Dengan mengimplementasikan program gudang toko grosir dengan BST, pengguna dapat melakukan berbagai fungsi seperti, memasukkan informasi produk baru, melihat informasi produk, memperbarui informasi produk, menghapus informasi produk, dan mencari berdasarkan nama produk atau harga produk yang ada.

* 1. **Reason of Choosing the Topic and Data Structures Method**

Kami memilih topik gudang toko grosir karena topik ini cocok dengan metode yang kami inginkan, yaitu BST. Dengan menggunakan BST, kami dapat memperoleh beberapa manfaat dalam mengorganisir data, yaitu mempercepat proses pencarian (daripada linear), dan data dapat ditampilkan secara urut menggunakan baik traversal preorder, inorder dan postorder (di program ini hanya ditampilkan traversal inorder saja).

* 1. **Overview of Program Definition**

Dalam program ini, pengguna yang bisa mengakses hanyalah admin yang mempunyai username dan passwordnya masing-masing. Di awal program, pengguna akan ditampilkan suatu judul mengenai program apa, kemudian pengguna diminta untuk memasukkan username dan passwordnya sesuai yang ada di data. Setelah berhasil masuk, user akan disuguhi 6 menu, sebelumnya kami jelaskan bahwa setiap node BST di program ini berisi informasi tentang suatu produk, yaitu nama produk, jumlah produk, harga barang, dan kunci yang dihasilkan dari hash nama produk :

* + - 1. Input data : menambahkan informasi tentang suatu produk di 1 node BST yang berisi nama produk, jumlah produk, harga barang dan kunci yang dihasilkan dari hash nama produk.
      2. View data : menampilkan seluruh data yang ada dalam inorder traversal (urut dari terkecil ke terbesar berdasarkan key yang dihasilkan).
      3. Update data : pengguna diminta menginput nama dari produk yang ingin diupdate. Fitur ini akan memberikan pilihan kepada pengguna apa yang ingin diupdate (terdapat 4 pilihan yang bisa diubah, yaitu nama produk saja, jumlah barang saja, harga barang saja, atau semua detail).
      4. Delete data : pengguna diminta menginput nama dari produk yang ingin didelete. Fitur ini akan memberikan pilihan kepada pengguna ingin menghapus stok saja atau menghapus semua detail.
      5. Search data : pencarian didasarkan pada kunci (hash) dari nama produk. Pengguna akan diminta untuk memilih ingin mencari berdasarkan nama produk atau harga barang.

1. **LITERATURE REVIEW**

Binary Search Tree (BST) adalah struktur data yang digunakan untuk menyimpan data dalam bentuk pohon biner yang urut (anak kiri selalu lebih kecil dari parent, sedangkan anak kanan selalu lebih besar dari parent). BST memungkinkan pencarian, penambahan, dan pengurangan data dengan cepat dan efisien. Pada kasus ini, BST digunakan untuk mengelola data gudang dari produk yang dijual oleh toko grosir.

BST memiliki beberapa properti yang membedakannya dari pohon biner biasa, yaitu:

* Semua node pada subtree kiri harus lebih kecil dari node root.
* Semua node pada subtree kanan harus lebih besar dari node root.
* Setiap subtree juga harus merupakan BST.

Penambahan data pada BST juga dilakukan dengan membandingkan nilai yang akan ditambahkan dengan nilai pada node root. Jika nilai yang akan ditambahkan lebih kecil dari nilai pada node root, maka penambahan dilakukan pada subtree kiri. Jika nilai yang akan ditambahkan lebih besar dari nilai pada node root, maka penambahan dilakukan pada subtree kanan. Penambahan dilakukan secara rekursif hingga nilai yang akan ditambahkan ditempatkan pada posisi yang tepat.

Pencarian data pada BST dilakukan dengan membandingkan nilai yang dicari dengan nilai pada node root. Jika nilai yang dicari lebih kecil dari nilai pada node root, maka pencarian dilanjutkan pada subtree kiri. Jika nilai yang dicari lebih besar dari nilai pada node root, maka pencarian dilanjutkan pada subtree kanan. Pencarian dilakukan secara rekursif hingga nilai yang dicari ditemukan atau tidak ditemukan.

Selain pengertian serta kegunaan yang telah disebutkan sebelumnya, BST juga memiliki beberapa keunggulan dan kelemahan yang perlu diperhatikan :

**Keunggulan BST :**

* + - 1. Pencarian efisien

Dengan menggunakan konsep pemisahan data dalam subtree yang lebih kecil dan lebih besar dari knot root, pencarian dalam BST dapat dilakukan secara efisien dengan kompleksitas waktu O(log n) pada rata- rata dan O(n) dalam kasus terburuk atau worse case.

* + - 1. Penyimpanan terurut

BST secara alami menyimpan data dalam urutan terurut. Hal ini memudahkan operasi yang membutuhkan data terurut, seperti mencari elemen terkecil atau terbesar dalam BST.

* + - 1. Pemeliharaan struktur terbilang mudah

BST memiliki aturan yang sederhana untuk pemeliharaan strukturnya. Setelah operasi penambahan atau pengurangan dilakukan, struktur BST dapat dengan mudah dipulihkan dengan melakukan rotasi atau pertukaran simpul yang tepat.

**Kelemahan BST :**

Ketidakseimbangan

Jika operasi penambahan atau pengurangan dilakukan secara tidak seimbang, BST dapat mengalami ketidakseimbangan. Hal ini dapat mengakibatkan waktu eksekusi operasi yang tidak efisien dengan kompleksitas waktu mencapai O(n), di mana n adalah jumlah simpul dalam BST.

Sensitivitas terhadap urutan input

Urutan penambahan atau pengurangan data dalam BST dapat mempengaruhi keseimbangan dan kinerja keseluruhan struktur. Jika data diurutkan secara menurun atau terurut sebelumnya, BST akan menjadi mirip dengan senarai terkait yang tidak efisien.

Diperlukan alokasi memori yang dinamis BST membutuhkan alokasi memori dinamis saat menambah atau menghapus simpul.

Jika pengelolaan memori tidak efisien, dapat terjadi fragmentasi memori atau penggunaan memori yang tidak teroptimalkan. Dalam konteks pengelolaan data gudang toko grosir sembako ini, BST dapat memberikan keunggulan dalam pencarian data dengan cepat dan efisien, serta kemampuan untuk mempertahankan data dalam urutan terurut. Namun, perlu diperhatikan pula untuk menjaga keseimbangan BST agar tetap efisien dalam penggunaan memori dan waktu eksekusi operasi.

1. **PROGRAM DEFINITION**
   1. **Pseudocode and Flowchart of the Program**

**Pseudocode :**

Deklarasi struct node:

struct node {

char namaProduk[21]

int jumlahBarang

int hargaBarang

unsigned int key

struct node \*left

struct node \*right

}

Deklarasi variabel global:

node \*root = NULL

int done = 0

Fungsi deleteTree(curr: node):

Jika curr bukan NULL, lakukan:

Panggil deleteTree(curr->left)

Panggil deleteTree(curr->right)

Bebaskan memori curr

Fungsi printHeader():

Cetak header tabel produk di toko grosir

Fungsi hash(str: char[]): unsigned int:

Inisialisasi hash\_val dengan 0

Untuk setiap karakter char dalam str, lakukan:

Tambahkan nilai ASCII karakter tersebut ke hash\_val

Kembalikan hasil hash\_val modulo MAX

Fungsi insert(curr: node, newnode: node, key: unsigned int):

Jika root adalah NULL, set root sebagai newnode

Jika key bukan sama dengan curr->key, lakukan:

Jika key kurang dari curr->key dan curr->left adalah NULL, lakukan:

Set temp sebagai newnode

Set curr->left sebagai temp

Jika key lebih dari curr->key dan curr->right adalah NULL, lakukan:

Set temp sebagai newnode

Set curr->right sebagai temp

Jika key kurang dari curr->key, panggil insert(curr->left, newnode, key) secara rekursif

Jika tidak, panggil insert(curr->right, newnode, key) secara rekursif

Fungsi importData():

Buka file "data.txt" dalam mode baca

Jika fp adalah NULL, kembalikan

Buat variabel temp bertipe node

Selama masih ada baris yang bisa dibaca dari fp, lakukan:

Baca namaProduk, jumlahBarang, dan hargaBarang dari fp

Buat newnode dengan alokasi memori berukuran node

Salin namaProduk, jumlahBarang, dan hargaBarang ke newnode

Hitung key menggunakan fungsi hash(namaProduk)

Set left dan right pada newnode sebagai NULL

Panggil insert(root, newnode, key)

Tutup file fp

Fungsi cekNamaProduk(namaProduk: char[]): int:

Hitung key menggunakan fungsi hash(namaProduk)

Set curr sebagai root

Selama curr bukan NULL, lakukan:

Jika key sama dengan curr->key dan strcmp(namaProduk, curr->namaProduk) adalah 0, lakukan:

Kembalikan 0

Jika key sama dengan curr->key dan strcmp(namaProduk, curr->namaProduk) tidak sama dengan 0 dan panjang namaProduk tidak sama dengan panjang curr->namaProduk, lakukan:

Kembalikan 2

Jika key sama dengan curr->key dan strcmp(namaProduk, curr->namaProduk) tidak sama dengan 0, lakukan:

Kembalikan 3

Jika key kurang dari curr->key, lakukan:

Set curr sebagai curr->left

Jika key lebih dari curr->key, lakukan:

Set curr sebagai curr->right

Kembalikan 1

Fungsi newNode(key: int, namaProduk: char[], jumlahBarang: int, hargaBarang: int): node

Buat variabel newNode bertipe node

Salin nilai key ke newNode->key

Salin nilai namaProduk ke newNode->namaProduk

Salin nilai jumlahBarang ke newNode->jumlahBarang

Salin nilai hargaBarang ke newNode->hargaBarang

Set newNode->left dan newNode->right sebagai NULL

Kembalikan newNode

Fungsi addProduct():

Buat variabel namaProduk[21] bertipe char

Buat variabel jumlahBarang dan hargaBarang bertipe int

Cetak "Masukkan nama produk: "

Baca input dan simpan dalam namaProduk

Jika cekNamaProduk(namaProduk) bukan 1, lakukan:

Cetak "Produk sudah ada dalam daftar!"

Kembalikan

Cetak "Masukkan jumlah barang: "

Baca input dan simpan dalam jumlahBarang

Cetak "Masukkan harga barang: "

Baca input dan simpan dalam hargaBarang

Hitung key menggunakan fungsi hash(namaProduk)

Buat newnode dengan pemanggilan fungsi newNode(key, namaProduk, jumlahBarang, hargaBarang)

Panggil insert(root, newnode, key)

Cetak "Produk berhasil ditambahkan!"

Fungsi searchProduct():

Buat variabel namaProduk[21] bertipe char

Cetak "Masukkan nama produk yang ingin dicari: "

Baca input dan simpan dalam namaProduk

Jika cekNamaProduk(namaProduk) adalah 0, lakukan:

Cetak "Produk ditemukan dalam daftar!"

Jika cekNamaProduk(namaProduk) adalah 1, lakukan:

Cetak "Produk tidak ditemukan dalam daftar!"

Jika cekNamaProduk(namaProduk) adalah 2, lakukan:

Cetak "Nama produk tidak cocok dengan entri lain dalam daftar!"

Jika cekNamaProduk(namaProduk) adalah 3, lakukan:

Cetak "Produk tidak ditemukan dalam daftar, namun nama produk serupa ada dalam daftar!"

Fungsi printProduct(curr: node):

Jika curr bukan NULL, lakukan:

Panggil printProduct(curr->left)

Cetak curr->namaProduk, curr->jumlahBarang, dan curr->hargaBarang

Panggil printProduct(curr->right)

Fungsi displayProducts():

Jika root adalah NULL, lakukan:

Cetak "Tidak ada produk dalam daftar!"

Jika root tidak NULL, lakukan:

Panggil printHeader()

Panggil printProduct(root)

Fungsi exportData():

Buka file "data.txt" dalam mode tulis

Jika fp adalah NULL, kembalikan

Tulis data produk ke file fp menggunakan inorder traversal

Tutup file fp

Fungsi main():

Panggil importData()

Set done sebagai 0

Selama done sama dengan 0, lakukan:

Cetak "Pilih operasi:"

Cetak "1. Tambah produk"

Cetak "2. Cari produk"

Cetak "3. Tampilkan produk"

Cetak "4. Keluar"

Baca input operasi dan simpan dalam done

Jika done adalah 1, panggil addProduct()

Jika done adalah 2, panggil searchProduct()

Jika done adalah 3, panggil displayProducts()

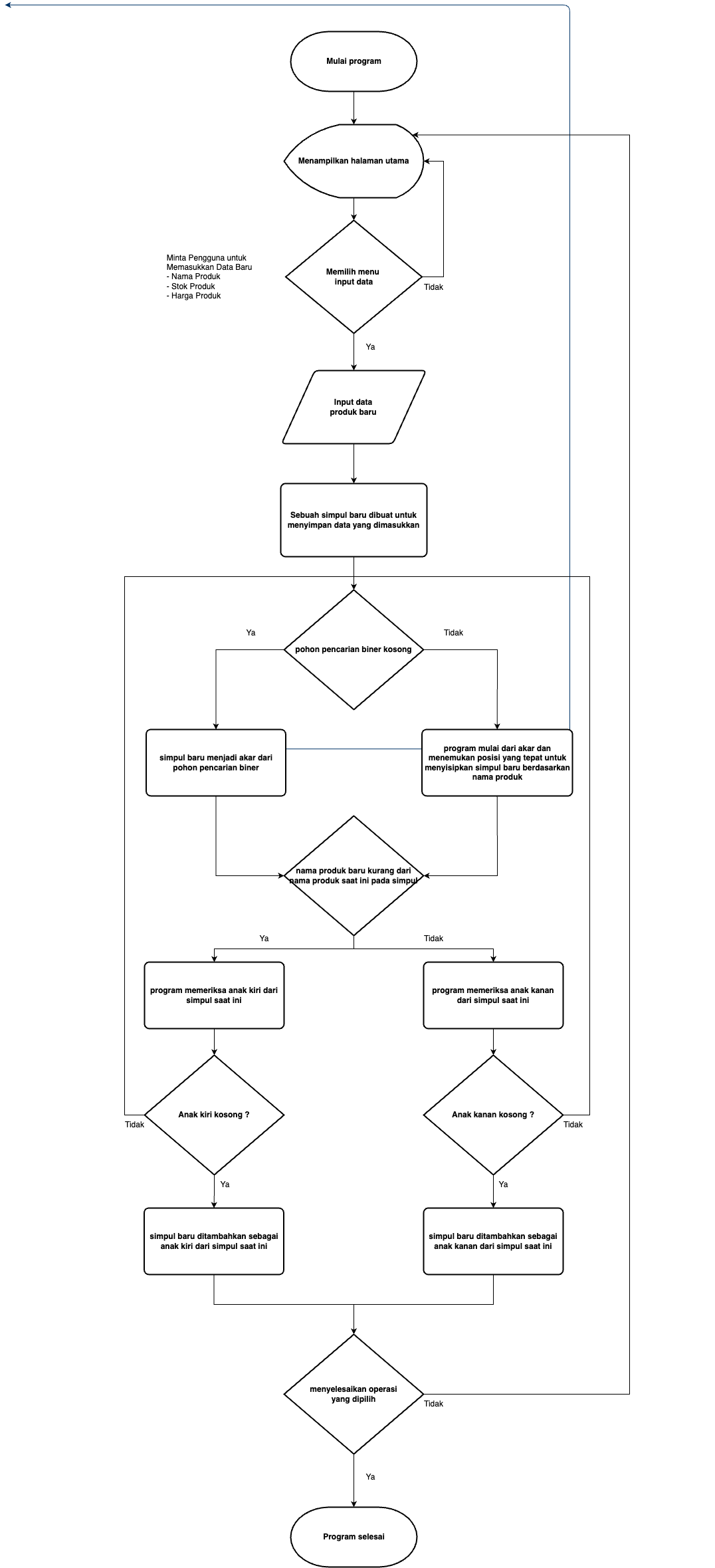
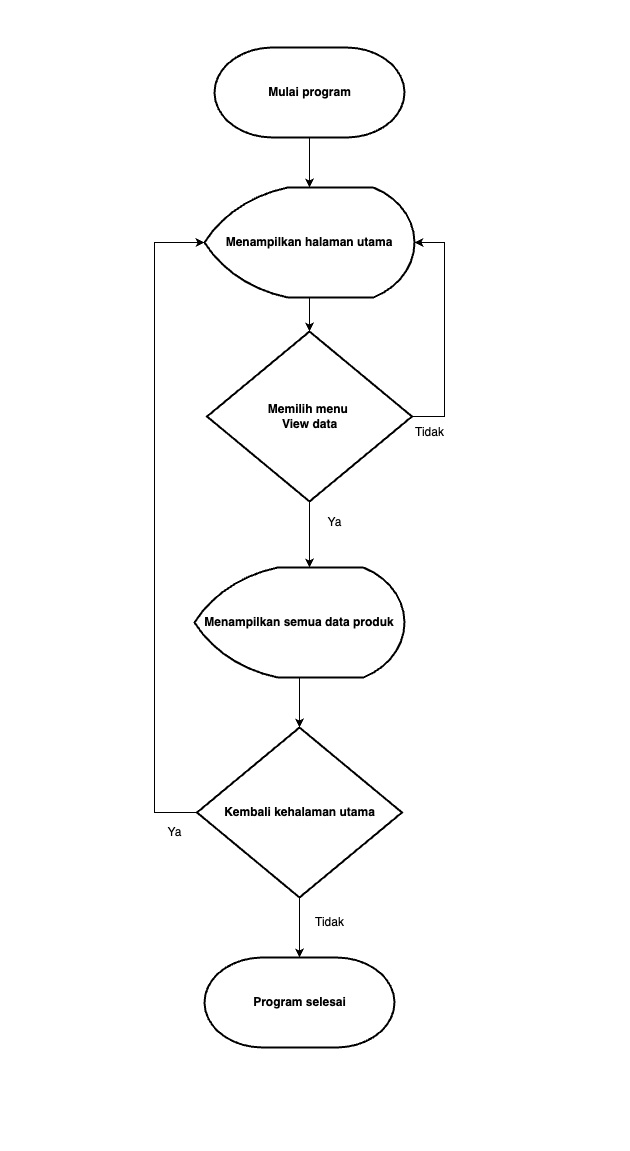
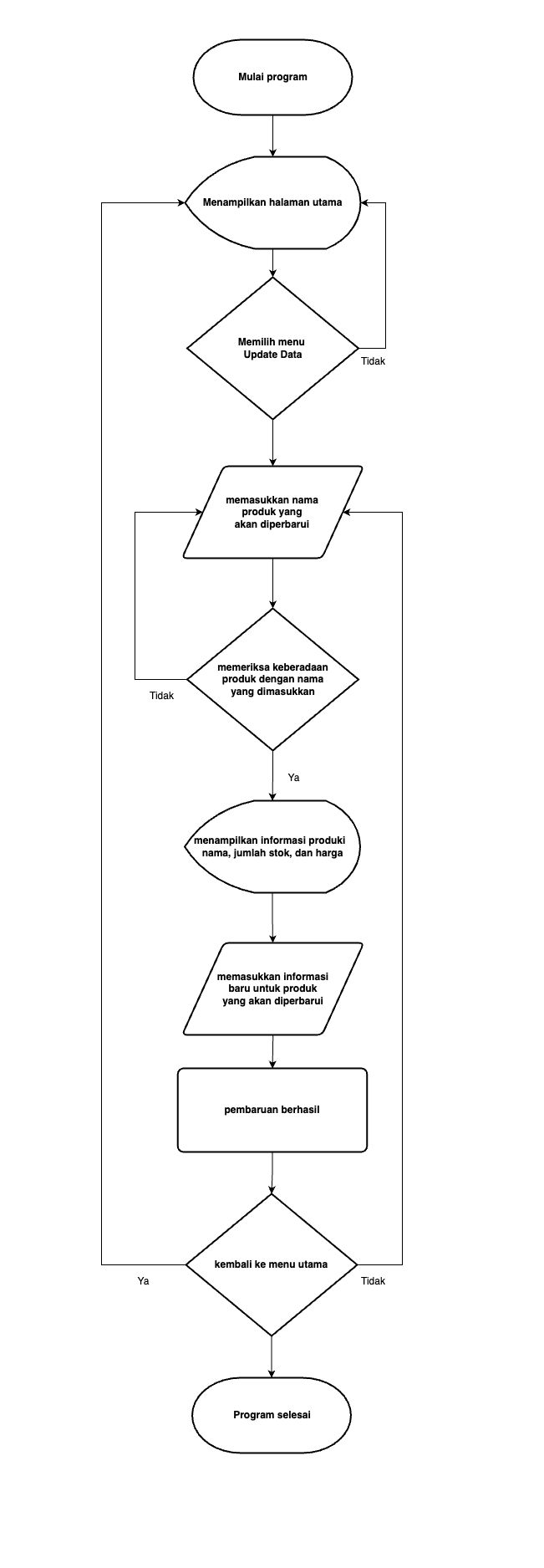
Jika done adalah 4, set done sebagai 1

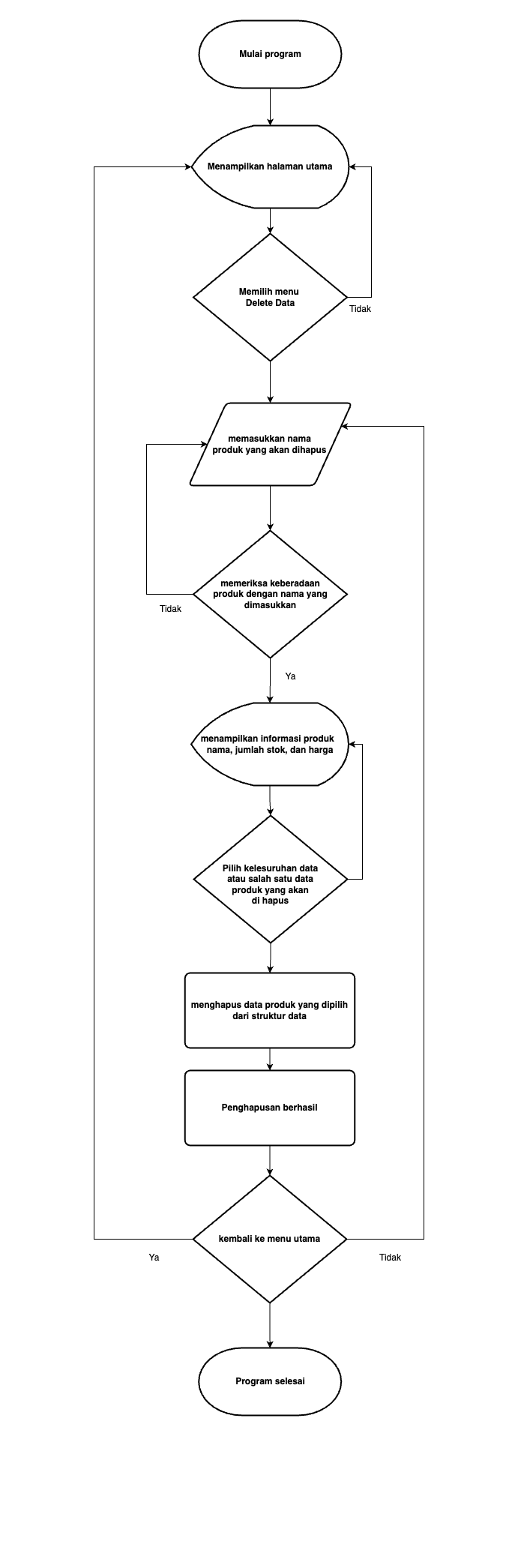
Panggil exportData()

Panggil deleteTree(root)

Kembalikan 0

**Flowchart :**

* + - 1. Input Data
      2. View Data
      3. Update Data
      4. Delete Data



* + - 1. Search Data

A picture containing drawing, sketch, diagram, line art

Description automatically generated

* 1. **Program Overview**

1. **Program Description**

Program toko grosir yang kami buat menggunakan metode binary search tree, program ini dibuat untuk mengelola data dari bahan pokok yang dijual oleh toko grosir tersebut. Toko ini memungkinkan penjual menambahkan, melihat, dan mengurangi data dari bahan pokok. Program pendataan toko grosir yang kami buat menggunakan metode Binary Search Tree. Metode atau struktur ini memungkinkan struktur data ini terbentuk dimana setiap node memiliki paling banyak dua anak. Dengan metode ini, program akan memastikan bahwa setiap elemen atau data bahan pokok yang disimpan pada suatu urutan tertentu. Tujuannya membuat penjual lebih mudah saat mengambil suatu data. Dengan ini kami memanfaatkan Binary Search Tree agar penyimpanan, penambahan, dan penghapusan data dari program ini dapat lebih efisien dan cepat terorganisir.

1. **A picture containing text, diagram, screenshot, line

   Description automatically generatedLayout Design**
2. **Program Features and Details on the Program**
   * 1. **Input data**

Memungkinkan toko menambahkan stok dari bahan pokok baru ke program. Data yang ditambahkan dapat berupa nama produk, stok produk, dan harga dalam produk tersebut. Data bahan pokok yang ditambahkan menggunakan Binary Search Tree yang memungkinkan program ini tertata dalam urutan yang benar berdasarkan data yang akan ditambahkan.

* + 1. **View data**

Menu lainnya yang dapat diakses dalam program ini adalah, program yang dibuat dapat menampilkan semua data dari toko grosir. Tampilan yang diberikan oleh program dapat berupa data nama, jumlah stok, harga dari produk yang dijualkan secara urut (inorder traversal).

* + 1. **Update data**

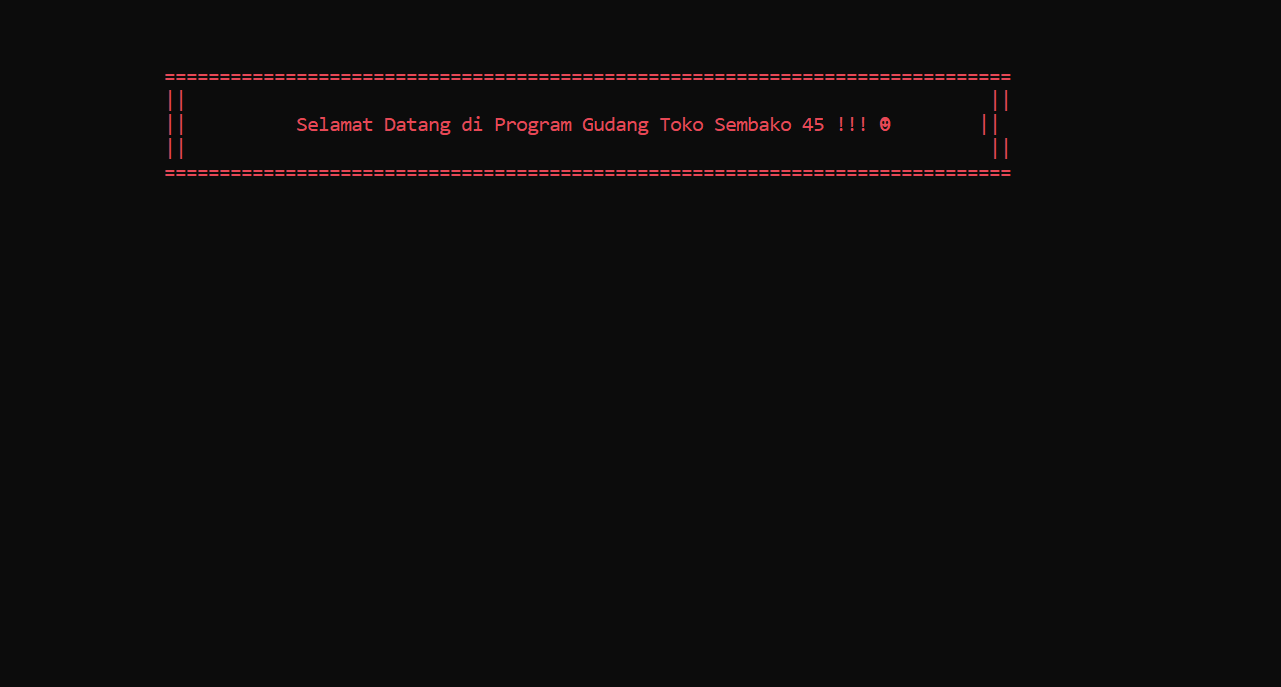
Pada menu update data, program yang telah dibuat dapat melakukan pembaharuan informasi contohnya seperti nama barang, stok barang, dan juga harga barang. Update data secara teratur dapat dimanfaatkan untuk sistem memberi informasi pada pelanggan sendiri tentang berapa jumlah barang yang telah tersedia.

* + 1. **Delete data**

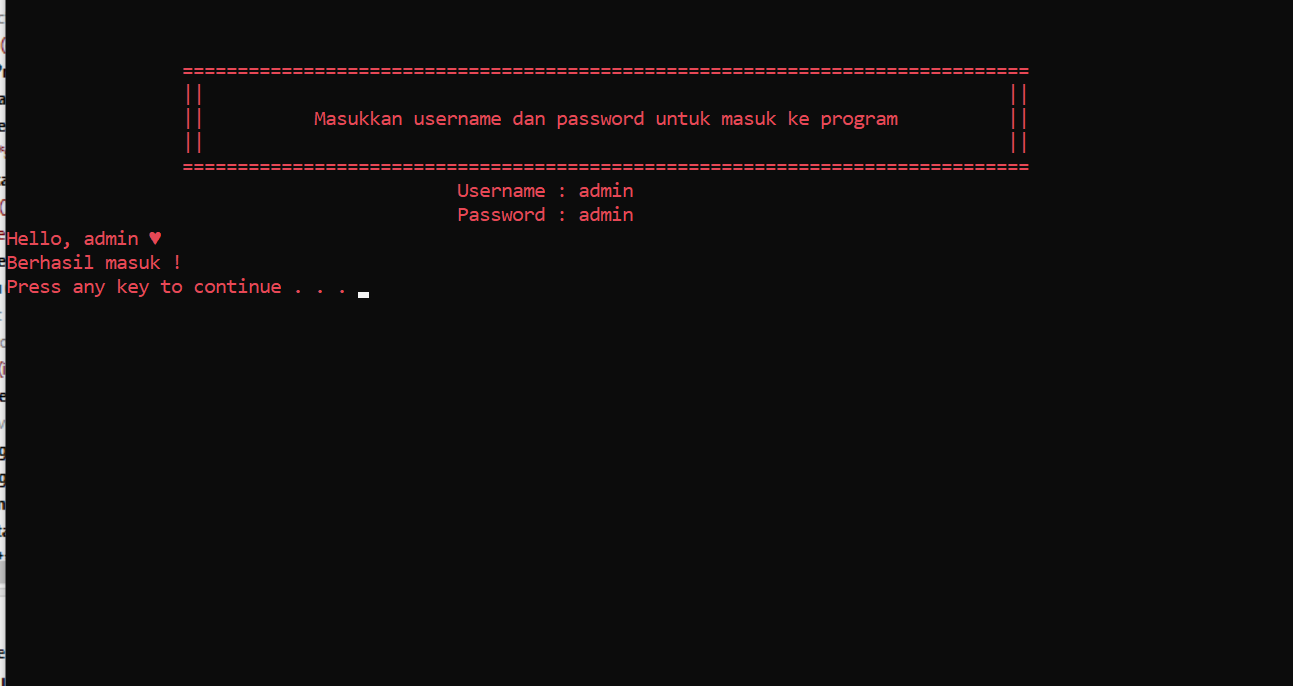
Program ini menyediakan menu penghapusan, sehingga penjual dengan mudah dapat menghapus suatu bahan pokok yang sudah tidak dijualkan kembali (semua detail dihapus) atau tidak tersedia (stok habis).

* + 1. **Search data**

Pada menu search data ini, digunakan untuk mencari data dengan memasukkan nama produk yang kemudian diubah menjadi sebuah key menggunakan hash. Lalu key tersebut akan dibandingkan oleh beberapa data yang ada, apabila terdapat kecocokan, maka program akan menampilkan hasil pencariannya. Adapun pencarian dibagi menjadi 2, yaitu pencarian berdasarkan nama produk (menggunakan metode di atas) dan pencarian berdasarkan harga barang tertentu (memasukkan harga barang yang ingin dicari : misal 10000, nanti yang muncul di pencarian adalah barang yang mempunyai harga Rp 10.000,-).

1. **RESULT**
   1. **Program Screenshot**
2. **Program Awal**

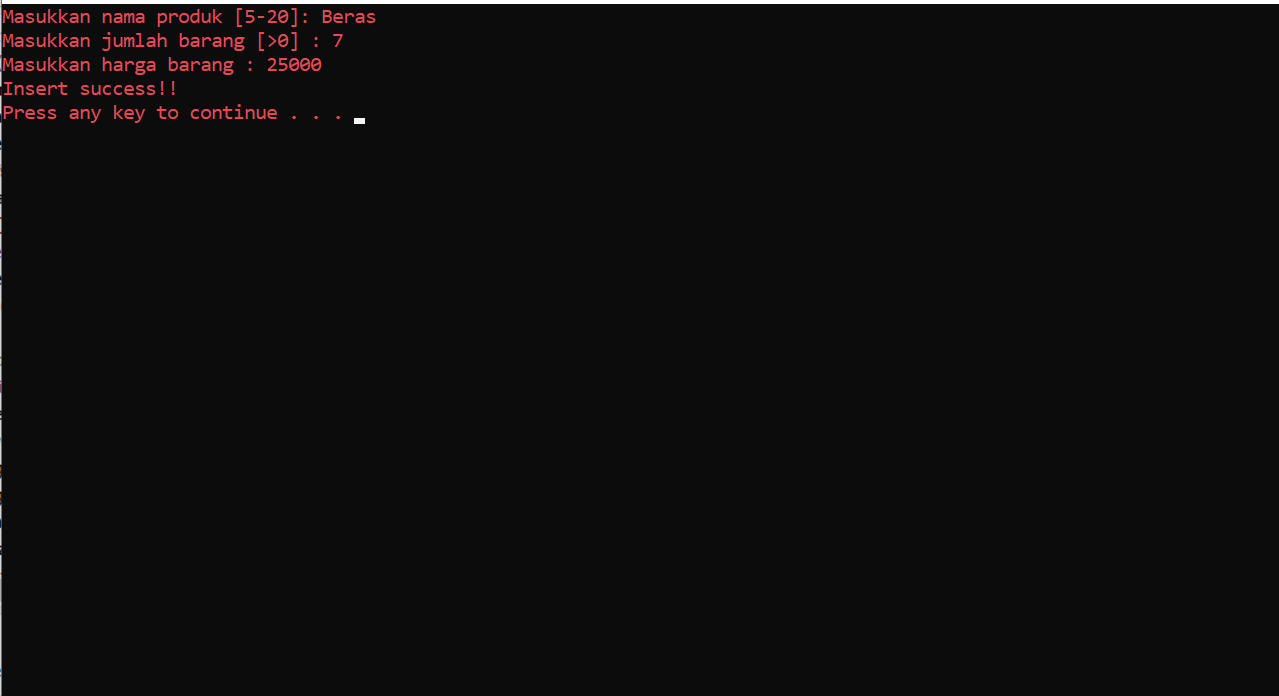
Program awal akan menampilkan tulisan selamat datang di program tersebut.



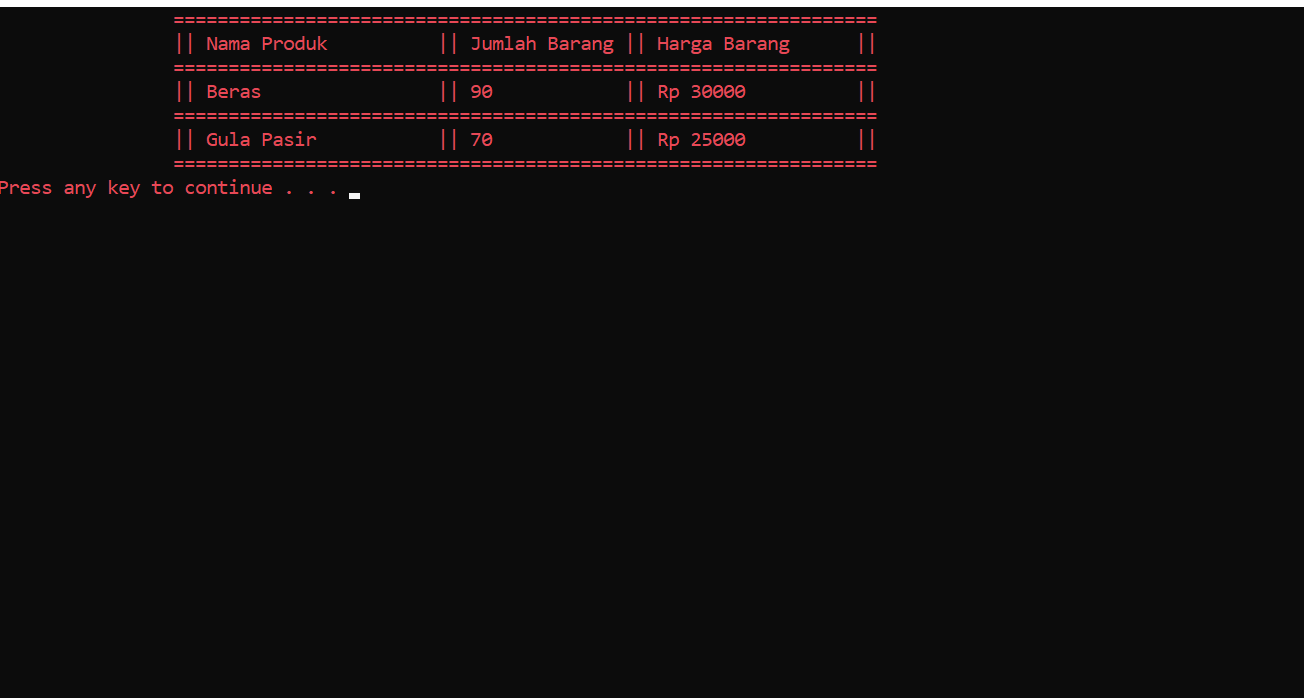
Setelah masuk kedalam program, admin dapat melakukan login pada program yaitu dengan menginput username dan password, disini program akan melakukan pengecekan username dan password yang diinput. Jika yang diinputkan benar, admin akan langsung masuk ke program.

1. **Menu Awal**

Admin akan diberi tampilan menu seperti ini, admin tinggal menginput angka sesuai dengan menu yang ada, misal ingin menginput data, maka admin harus menginput angka 1. Apabila angka yang diinput tidak ada di dalam menu, maka menu akan terulang sampai admin memasukkan angka yang benar.

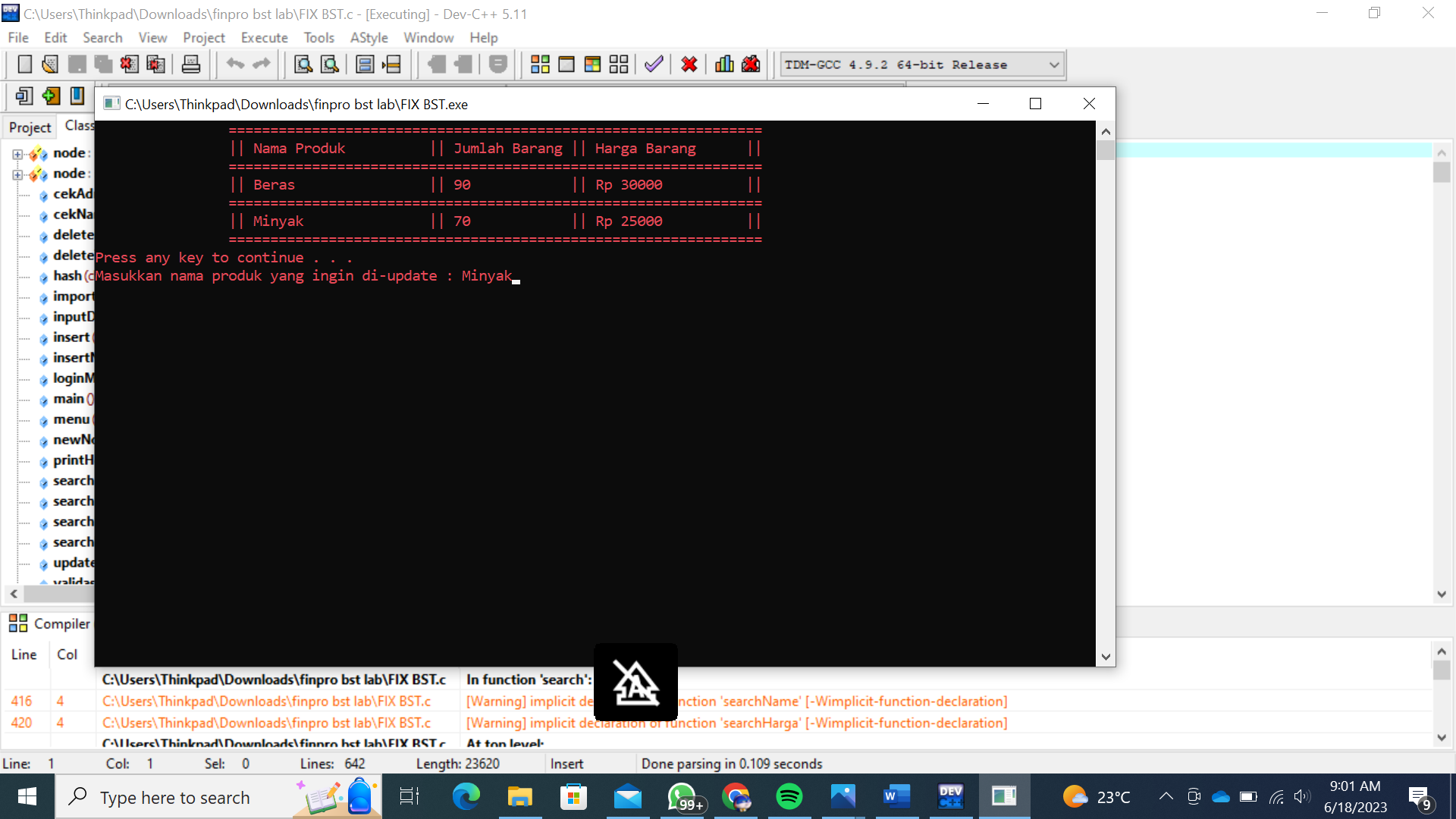
1. **Input Data**

Dalam menu input data, kita akan diminta 3 data mengenai produk yang ingin kita input : nama produk, jumlah barang, dan harga barang. Sudah terdapat sanitasi apabila nama produk yang diinput tidak sesuai dengan kriteria (huruf 5-20 dan apabila nama sudah ada di dalam data).

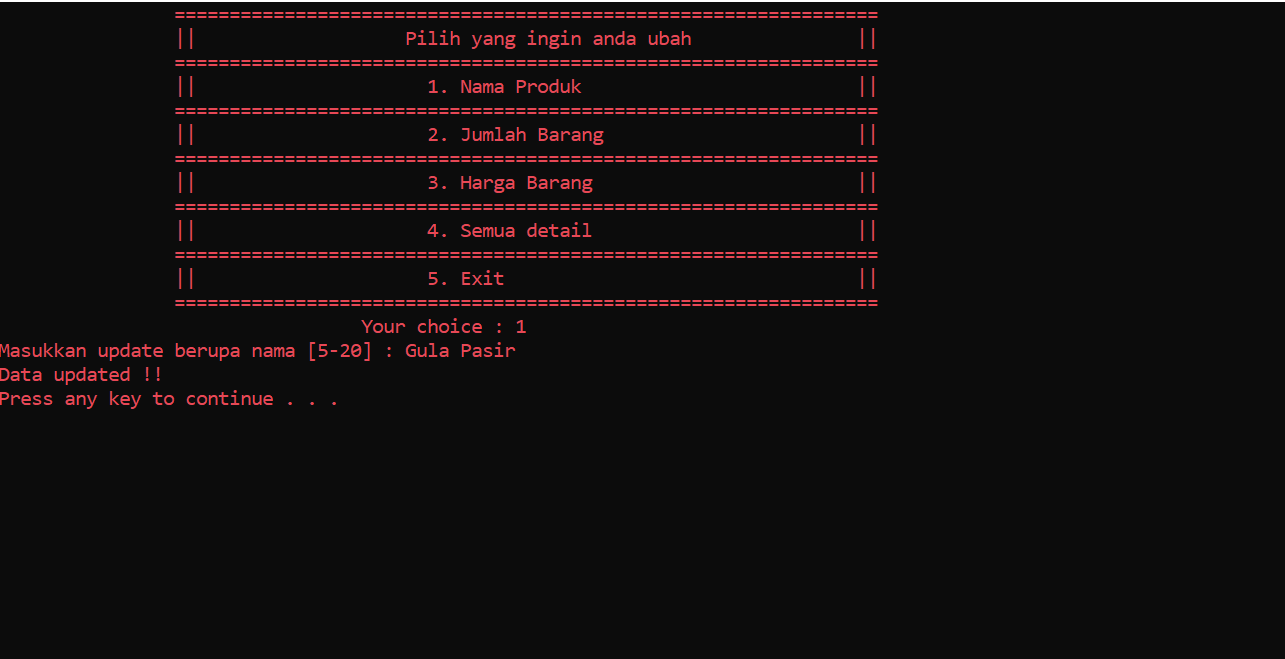
1. **View Data**

Program akan menampilkan data secara inorder traversal (berdasarkan key yang diperoleh dari nama produk) dan mengeluarkan semua detail produk yang ada (nama produk, jumlah barang, dan harga barang).

1. **Update Data**

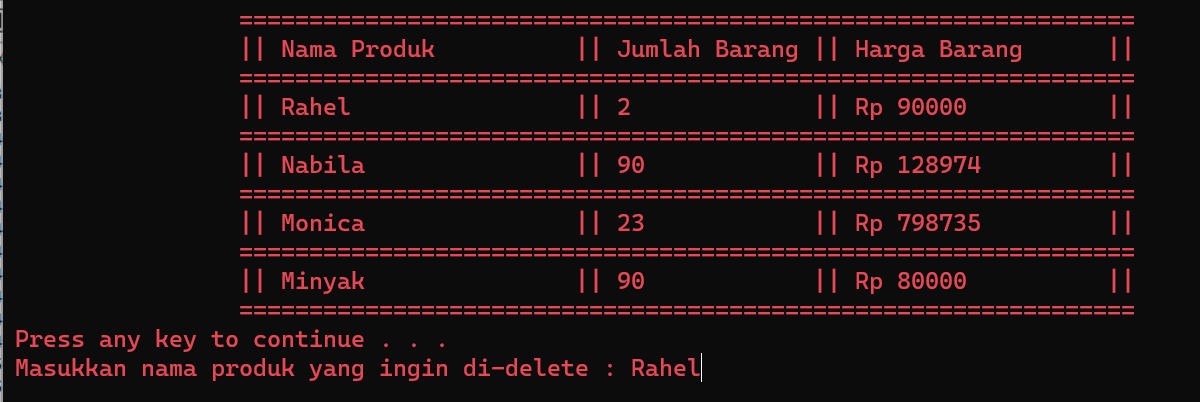


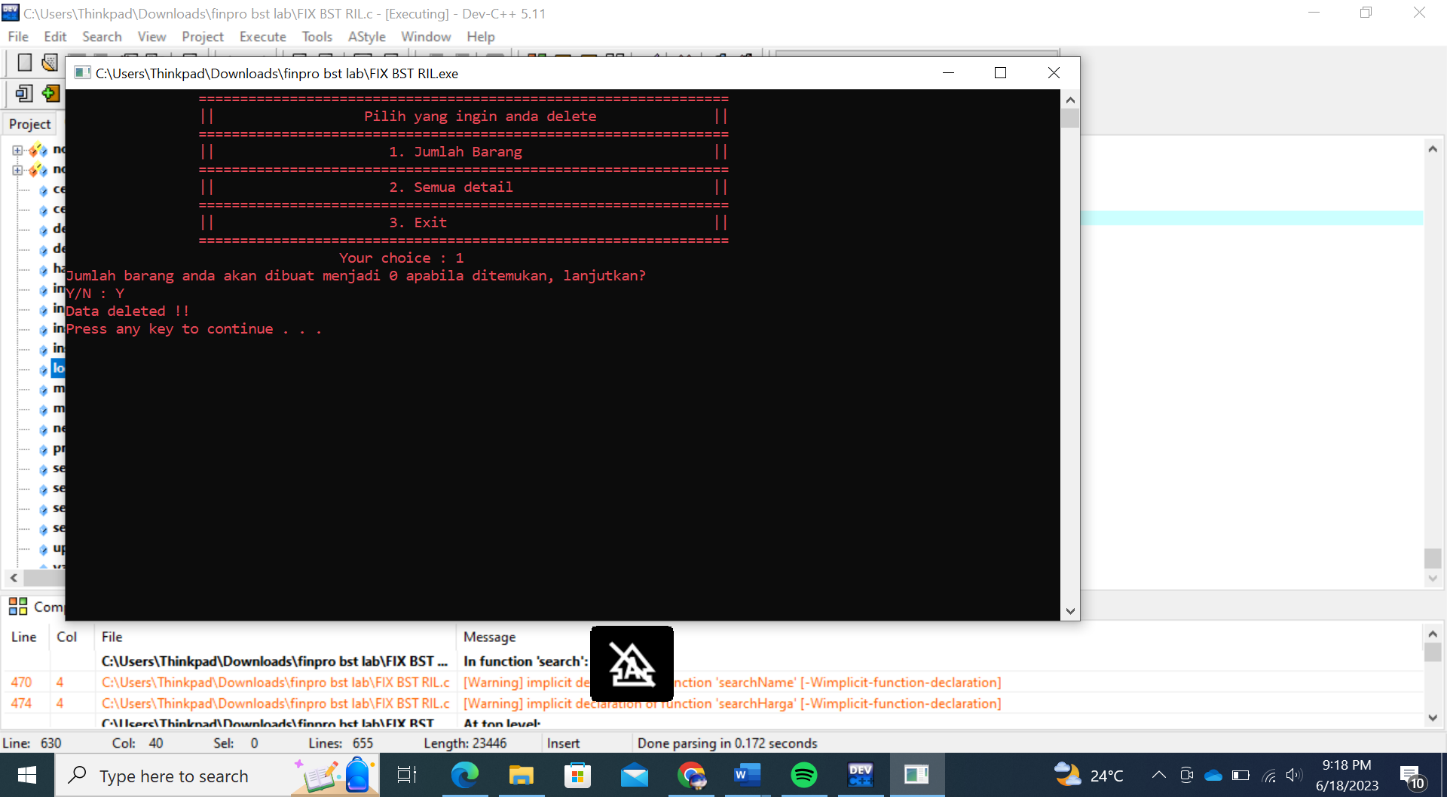
Dalam update data, pertama-tama admin akan ditunjukkan data produk yang sudah ada, lalu admin diminta untuk memasukkan nama produk yang ingin di update.



Setelah itu, program akan memberikan pilihan pada admin ingin mengupdate detail apa dari produk tersebut. Apabila admin memilih nomor 1, maka admin hanya perlu memberikan inputan nama produk yang baru. Adapun inputan produk yang baru dilarang sama dengan yang sudah ada di tree atau di data. Apabila admin memilih nomor 2, maka admin hanya perlu memberikan inputan jumlah barang yang baru, dan apabila admin memilih nomor 3, maka admin hanya perlu memberikan inputan harga barang yang baru. Yang terakhir, apabila admin ingin mengubah semua detail, maka admin akan diminta memasukkan semua inputan yang dibutuhkan. Apabila data ditemukan dan berhasil diubah, maka program akan menampilkan pesan bahwa data sudah berhasil diperbarui atau diupdate. Sedangkan apabila tidak ada data yang cocok untuk diubah di tree atau di data, maka program akan menampilkan pesan bahwa data tidak ditemukan.

1. **Delete Data**

****

Di program delete data awal, admin akan ditampilkan data yang ada dan admin diminta memasukkan nama produk yang ingin di delete.

Setelah itu, admin akan diminta memilih ingin mendelete apa. Setelah memilih, admin akan diberi pesan konfirmasi dan admin akan mengonfirmasinya. Setelah itu apabila admin memilih untuk delete dan data ditemukan, maka akan muncul pesan bahwa data telah terhapus. Sedangkan sebaliknya, apabila data tidak ditemukan maka program akan menampilkan pesan bahwa data tidak ditemukan.

1. **Search Data**

**A picture containing text, screenshot, font, line

Description automatically generated**

Apabila admin memilih menu search data, maka admin akan disuguhi tampilan seperti ini. Admin diminta untuk menginput ingin mencari data apa di dalam fitur tersebut.

A picture containing screenshot, font, graphics, typography

Description automatically generated

A screenshot of a video game

Description automatically generated with low confidence

Apabila data ditemukan, maka akan ditampilkan, apabila tidak ditemukan, maka program akan menampilkan tulisan bahwa data tidak ditemukan.

A picture containing font, graphics, screenshot, typography

Description automatically generated

A picture containing text, screenshot, font, line

Description automatically generated

Atau bisa juga mencari data menggunakan harga yang ingin kita cari.

1. **Exit**

**A screenshot of a computer program

Description automatically generated with low confidence**

Apabila admin memilih menu 6 yaitu exit, maka admin akan langsung keluar dari program. Program akan tertutup, data sudah pasti tersimpan.

* 1. **Program Code**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <conio.h> // Library untuk fungsi-fungsi khusus seperti getch() yang digunakan dalam program ini

#include <ctype.h> // Library untuk fungsi-fungsi karakter seperti isdigit() yang digunakan dalam program ini

typedef struct node{

char namaProduk[21];

int jumlahBarang;

int hargaBarang;

unsigned int key;

struct node \*left;

struct node \*right;

}node;

node \*root = NULL;

#define MAX 1000

int done=0;

void deleteTree(node \*curr){ // menghapus simpul kanan dan kiri pada b tree menggunakan free(curr)

if(curr!=NULL){

deleteTree(curr->left);

deleteTree(curr->right);

free(curr);

}

}

void printHeader(){ // cetak header tabel produk di toko grosir

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

printf(" || Nama Produk || Jumlah Barang || Harga Barang ||\n");

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

}

unsigned int hash(char \*str) { // mencari key, diambil dari angka ascii per char yang ada pada string

unsigned int hash\_val = 0;

for (int i = 0; str[i] != '\0'; i++) {

hash\_val = hash\_val + (unsigned int) str[i];

}return hash\_val % MAX;

}

void insert(node \*curr, node \*newnode, unsigned int key){ // mengelola penyisipan node baru dalam tree sesuai dengan nilai kunci yang ditentukan.

if(root == NULL){

root = newnode;

}else if(key != curr->key){ // jika key kurang dari nilai kunci maka node tidak memiliki anak kiri

if(key<curr->key && curr->left == NULL){

node \*temp = newnode;

curr->left = temp; // jika key kurang dari nilai kunci kanan, maka ia tidak memiliki anak kanan

}else if(key>curr->key && curr->right == NULL){

node \*temp = newnode;

curr->right = temp;

}else if(key<curr->key){

insert(curr->left, newnode, key); //rekursif

}else{

insert(curr->right, newnode, key);

}

}

}

void importData(){

FILE \*fp = fopen("data.txt", "r"); // membuka file txt

if(fp==NULL){

return;

}

node temp;

while(fscanf(fp, "%[^;];%d;%d\n", temp.namaProduk, &temp.jumlahBarang, &temp.hargaBarang)!=EOF){ // membaca file txt

node \*newnode = (node\*)malloc(sizeof(node));

strcpy(newnode->namaProduk, temp.namaProduk);

newnode->jumlahBarang = temp.jumlahBarang;

newnode->hargaBarang = temp.hargaBarang;

newnode->key = hash(newnode->namaProduk);

newnode->left = NULL; newnode->right = NULL;

insert(root, newnode, newnode->key); // panggil insert() agar newnode dapat disisipkan ke BST

}

fclose(fp);

return;

}

int cekNamaProduk(char namaProduk[21]){

unsigned int key = hash(namaProduk);

node \*curr = root;

while(curr!=NULL){

if(key == curr->key && strcmp(namaProduk, curr->namaProduk)==0){

//printf("Nama sudah ada!!\n");

return 0;

}else if(key == curr->key && strcmp(namaProduk, curr->namaProduk)!=0 && strlen(namaProduk)!=strlen(curr->namaProduk)){

return 2; // sudah ada data yang bernama itu

}else if(key == curr->key && strcmp(namaProduk, curr->namaProduk)!=0){

return 3; // untuk data yang hampir sama menghindari kesalahan pada typo (misal beras dengan bersa)

}else if(key < curr->key){

curr = curr->left;

}else if(key > curr->key){

curr = curr->right;

}

}return 1;

}

node \*newNode(int key, char namaProduk[21], int jumlahBarang, int hargaBarang){

node \*newnode = (node\*)malloc(sizeof(node));

newnode->key = key;

strcpy(newnode->namaProduk, namaProduk);

newnode->jumlahBarang = jumlahBarang;

newnode->hargaBarang = hargaBarang;

newnode->left = NULL;

newnode->right = NULL;

FILE \*fp = fopen("data.txt", "a");

fprintf(fp, "%s;%d;%d\n", newnode->namaProduk, newnode->jumlahBarang, newnode->hargaBarang);

fclose(fp);

return(newnode);

}

node \*insertNode(node \*curr, char namaProduk[21], int jumlahBarang, int hargaBarang){

if (curr == NULL) // cek apakah node saat ini NULL, jika ya maka node baru dapat dimasukkan

return(newNode(hash(namaProduk), namaProduk, jumlahBarang, hargaBarang));

if (hash(namaProduk) < curr->key) // ditempatkan sebelah kiri

curr->left = insertNode(curr->left, namaProduk, jumlahBarang, hargaBarang);

else if (hash(namaProduk) > curr->key) // ditempatkan sebelah kanan

curr->right = insertNode(curr->right, namaProduk, jumlahBarang, hargaBarang);

else

return curr;

}

void inputData(){

char namaProduk[21];

int jumlahBarang;

int hargaBarang;

system("cls");

do{

printf("Masukkan nama produk [5-20]: "); scanf("%[^\n]", namaProduk); getchar();

if(strlen(namaProduk)<5 || strlen(namaProduk)>20) printf("Nama yang anda masukkan harus sepanjang 5-20 huruf!\n");

else if(cekNamaProduk(namaProduk)==0) printf("Nama yang anda masukkan sudah ada di data\n");

}while(cekNamaProduk(namaProduk)==0 || (strlen(namaProduk))<5 || (strlen(namaProduk))>20);

do{

printf("Masukkan jumlah barang [>0] : "); scanf("%d", &jumlahBarang); getchar();

if(jumlahBarang<=0) printf("Jumlah barang yang anda masukkan harus lebih dari 0!\n");

}while(jumlahBarang<=0);

do{

printf("Masukkan harga barang : "); scanf("%d", &hargaBarang); getchar();

if(hargaBarang<=0) printf("Harga barang yang anda masukkan harus lebih dari 0!\n");

}while(hargaBarang<=0);

if(root==NULL){

root = insertNode(root, namaProduk, jumlahBarang, hargaBarang);

}else if(root!=NULL){

insertNode(root, namaProduk, jumlahBarang, hargaBarang);

}

printf("Insert success!!\n");

system("pause");

}

void view(node \*curr){

if(curr!=NULL){

view(curr->left);

printf(" || %-20s || %-13d || Rp %-14d ||\n", curr->namaProduk, curr->jumlahBarang, curr->hargaBarang);

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

view(curr->right);

}

}

void validasiData(){

if(root==NULL){

system("cls");

printf("There is no data !!\n");

}else{

system("cls");

printHeader();

view(root);

}

}

void updateData(){

system("cls");

FILE \*fp = fopen("data.txt", "r");

if(root==NULL){

printf("There is no data");

system("pause");

return;

}

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

int found = 0;

validasiData();

system("pause");

char namaProduk[21];

do{

printf("Masukkan nama produk yang ingin di-update : "); scanf("%[^\n]", namaProduk); getchar();

}while(cekNamaProduk(namaProduk)==2 || strlen(namaProduk)<5 || strlen(namaProduk)>20);

unsigned int key = hash(namaProduk);

int choice;

node \*tempData = (node\*)malloc(sizeof(node));

do{

system("cls");

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

printf(" || Pilih yang ingin anda ubah ||\n");

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

printf(" || 1. Nama Produk ||\n");

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

printf(" || 2. Jumlah Barang ||\n");

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

printf(" || 3. Harga Barang ||\n");

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

printf(" || 4. Semua detail ||\n");

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

printf(" || 5. Exit ||\n");

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

printf(" Your choice : "); scanf("%d", &choice); getchar();

}while(choice<1 || choice>5);

switch(choice){

case 1:

while(fscanf(fp, "%[^;];%d;%d\n", tempData->namaProduk, &tempData->jumlahBarang, &tempData->hargaBarang)!=EOF){

if(strcmp(namaProduk, tempData->namaProduk)==0 && found==0){

found = 1;

char tempNama[21];

do{

printf("Masukkan update berupa nama [5-20] : "); scanf("%[^\n]", tempNama); getchar();

}while(cekNamaProduk(tempNama)==2 || cekNamaProduk(tempNama)==0 || strlen(tempNama)<5 || strlen(tempNama)>20);

// kita delete setelah user memasukkan

deleteTree(root);

root = NULL;

fprintf(temp, "%s;%d;%d\n", tempNama, tempData->jumlahBarang, tempData->hargaBarang);

}else{

fprintf(temp, "%s;%d;%d\n", tempData->namaProduk, tempData->jumlahBarang, tempData->hargaBarang);

}

}if(found==0){

printf("Data not found!!\n");

system("pause");

}else if(found==1){

printf("Data updated !!\n");

system("pause");

}

fclose(temp); fclose(fp);

remove("data.txt");

rename("temp.txt", "data.txt");

break;

case 2:

while(fscanf(fp, "%[^;];%d;%d\n", tempData->namaProduk, &tempData->jumlahBarang, &tempData->hargaBarang)!=EOF){

if(strcmp(namaProduk, tempData->namaProduk)==0 && found==0){

found = 1;

int tempJumlah;

deleteTree(root);

root = NULL;

do{

printf("Masukkan update jumlah barang : "); scanf("%d", &tempJumlah); getchar();

}while(tempJumlah<=0);

fprintf(temp, "%s;%d;%d\n", tempData->namaProduk, tempJumlah, tempData->hargaBarang);

}else{

fprintf(temp, "%s;%d;%d\n", tempData->namaProduk, tempData->jumlahBarang, tempData->hargaBarang);

}

}if(found==0){

printf("Data not found!!\n");

system("pause");

}else if(found==1){

printf("Data updated !!\n");

system("pause");

}

fclose(temp); fclose(fp);

remove("data.txt");

rename("temp.txt", "data.txt");

break;

case 3:

while(fscanf(fp, "%[^;];%d;%d\n", tempData->namaProduk, &tempData->jumlahBarang, &tempData->hargaBarang)!=EOF){

if(strcmp(namaProduk, tempData->namaProduk)==0 && found==0){

found = 1;

int tempHarga;

// kita delete setelah user memasukkan

deleteTree(root);

root = NULL;

do{

printf("Masukkan update harga barang : "); scanf("%d", &tempHarga); getchar();

}while(tempHarga<=0);

fprintf(temp, "%s;%d;%d\n", tempData->namaProduk, tempData->jumlahBarang, tempHarga);

}else{

fprintf(temp, "%s;%d;%d\n", tempData->namaProduk, tempData->jumlahBarang, tempData->hargaBarang);

}

}if(found==0){

printf("Data not found!!\n");

system("pause");

}else if(found==1){

printf("Data updated !!\n");

system("pause");

}

fclose(temp); fclose(fp);

remove("data.txt");

rename("temp.txt", "data.txt");

break;

case 4:

while(fscanf(fp, "%[^;];%d;%d\n", tempData->namaProduk, &tempData->jumlahBarang, &tempData->hargaBarang)!=EOF){

if(strcmp(namaProduk, tempData->namaProduk)==0 && found==0){

found = 1;

char tempNama[21];

int tempJumlah;

int tempHarga;

do{

printf("Masukkan update berupa nama [5-20] : "); scanf("%[^\n]", tempNama); getchar();

}while(cekNamaProduk(tempNama)==2 || cekNamaProduk(tempNama)==0 || strlen(tempNama)<5 || strlen(tempNama)>20);

// kita delete setelah user memasukkan

deleteTree(root);

root = NULL;

do{

printf("Masukkan update jumlah barang : "); scanf("%d", &tempJumlah); getchar();

}while(tempJumlah<=0);

do{

printf("Masukkan update harga barang : "); scanf("%d", &tempHarga); getchar();

}while(tempHarga<=0);

fprintf(temp, "%s;%d;%d\n", tempNama, tempJumlah, tempHarga);

}else{

fprintf(temp, "%s;%d;%d\n", tempData->namaProduk, tempData->jumlahBarang, tempData->hargaBarang);

}

}if(found==0){

printf("Data not found!!\n");

system("pause");

}else if(found==1){

printf("Data updated !!\n");

system("pause");

}

fclose(temp); fclose(fp);

remove("data.txt");

rename("temp.txt", "data.txt");

break;

case 5:

exit(0);

break;

}

}

void deleteData(){ // hapus data dari BST dan memperbarui data yang ada

system("cls");

FILE \*fp = fopen("data.txt", "r"); // buka data yang ada

if(root==NULL){

printf("There is no data");

system("pause");

return;

}

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

int found = 0;

validasiData();

system("pause");

char namaProduk[21];

do{

printf("Masukkan nama produk yang ingin di-delete : "); scanf("%[^\n]", namaProduk); getchar();

}while(cekNamaProduk(namaProduk)==2 || strlen(namaProduk)<5 || strlen(namaProduk)>20);

unsigned int key = hash(namaProduk);

int choice;

node \*tempData = (node\*)malloc(sizeof(node));

do{

system("cls");

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

printf(" || Pilih yang ingin anda delete ||\n");

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

printf(" || 1. Jumlah Barang ||\n");

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

printf(" || 2. Semua detail ||\n");

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

printf(" || 3. Exit ||\n");

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

printf(" Your choice : "); scanf("%d", &choice); getchar();

}while(choice<1 || choice>3);

char ch;

switch(choice){

case 1:

printf("Jumlah barang anda akan dibuat menjadi 0 apabila ditemukan, lanjutkan?\n");

printf("Y/N : "); scanf("%c", &ch); getchar();

if(ch=='Y'){

while(fscanf(fp, "%[^;];%d;%d\n", tempData->namaProduk, &tempData->jumlahBarang, &tempData->hargaBarang)!=EOF){

if(strcmp(namaProduk, tempData->namaProduk)==0 && found==0){

found = 1;

deleteTree(root);

root = NULL;

fprintf(temp, "%s;0;%d\n", tempData->namaProduk, tempData->hargaBarang);

}else{

fprintf(temp, "%s;%d;%d\n", tempData->namaProduk, tempData->jumlahBarang, tempData->hargaBarang);

}

}

fclose(temp); fclose(fp);

remove("data.txt");

rename("temp.txt", "data.txt");

}

break;

case 2:

printf("Semua detail barang anda akan dihapus apabila ditemukan datanya, lanjutkan?\n");

printf("Y/N : "); scanf("%c", &ch); getchar();

if(ch=='Y'){

while(fscanf(fp, "%[^;];%d;%d\n", tempData->namaProduk, &tempData->jumlahBarang, &tempData->hargaBarang)!=EOF){

if(strcmp(namaProduk, tempData->namaProduk)==0 && found==0){

found = 1; // melacak apakah data yang ingin dihapus ditemukan

deleteTree(root); // agar root menjadi NULL

root = NULL;

}else{

fprintf(temp, "%s;%d;%d\n", tempData->namaProduk, tempData->jumlahBarang, tempData->hargaBarang);

}

}fclose(temp); fclose(fp);

remove("data.txt"); // penghapusan data pada txt

rename("temp.txt", "data.txt");

}

break;

case 3:

break;

}if(found==0){

printf("Data not found!!\n");

system("pause");

}else if(found==1){

printf("Data deleted !!\n");

system("pause");

}

}

void search(){

if(root==NULL){

system("cls");

printf("There is no data!!\n");

system("pause");

return;

}

int choice;

done = 0;

do{

system("cls");

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

printf(" || Pilih yang ingin anda gunakan sebagai detail pencarian ||\n");

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

printf(" || 1. Nama Produk ||\n");

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

printf(" || 2. Harga Barang ||\n");

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

printf(" || 3. Keluar ||\n");

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

printf(" Your choice : "); scanf("%d", &choice); getchar();

}while(choice<1 || choice>3);

switch(choice){

case 1:

searchName();

system("pause");

break;

case 2:

searchHarga();

system("pause");

break;

case 3:

break;

}

}

void searchHarga(){

system("cls");

int harga;

do{

printf("Masukkan harga barang yang ingin dicari : "); scanf("%d", &harga); getchar();

}while(harga<=0);

searchHarga2(root, harga);

if(done==0){

printf("Data not found!!\n");

}

}

void searchHarga2(node \*curr, int harga){

if(curr!=NULL){

searchHarga2(curr->left, harga);

if(harga == curr->hargaBarang){

if(done==0){

printHeader();

done=1;

}

printf(" || %-20s || %-13d || Rp %-14d ||\n", curr->namaProduk, curr->jumlahBarang, curr->hargaBarang);

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

}searchHarga2(curr->right, harga);

}return;

}

void searchName(){

system("cls");

char namaProduk[21];

do{

printf("Masukkan nama produk yang ingin dicari : "); scanf("%[^\n]", namaProduk); getchar();

}while((strlen(namaProduk))<5 || (strlen(namaProduk))>20);

unsigned int key = hash(namaProduk);

node \*curr = root;

int found=0;

while(curr!=NULL){

if(key == curr->key && strcmp(namaProduk, curr->namaProduk)==0){

system("cls");

printHeader();

printf(" || %-20s || %-13d || Rp %-14d ||\n", curr->namaProduk, curr->jumlahBarang, curr->hargaBarang);

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

found = 1;

return;

}else if(key < curr->key){

curr = curr->left;

}else if(key > curr->key){

curr = curr->right;

}

}if(found==0){

printf("Data not found !!\n");

}

}

void menu(){

int choice;

do{

system("cls");

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

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

printf(" || ||\n");

printf(" || PROGRAM GUDANG TOKO SEMBAKO 45 ||\n");

printf(" || ||\n");

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

printf(" || ||\n");

printf(" || 1. INPUT DATA ||\n");

printf(" || 2. VIEW DATA ||\n");

printf(" || 3. UPDATE DATA ||\n");

printf(" || 4. DELETE DATA ||\n");

printf(" || 5. SEARCH DATA ||\n");

printf(" || 6. EXIT ||\n");

printf(" || ||\n");

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

printf(" Input your choice [1-6] : "); scanf("%d", &choice); getchar();

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

}while(choice<1 || choice>6);

importData();

switch(choice){

case 1:

inputData();

break;

case 2:

validasiData();

system("pause");

break;

case 3:

updateData();

break;

case 4:

deleteData();

break;

case 5:

search();

break;

case 6:

exit(0);

break;

}

}

int cekAdmin(char username[11], char password[11]){

FILE \*admin = fopen("user.txt", "r");

char tempName[11];

char tempPw[11];

int found = 0;

int foundpw = 0;

int result;

while(fscanf(admin, "%[^;];%[^\n]\n", tempName, tempPw)!=EOF){

if(strcmp(username, tempName)==0){

if(strcmp(password, tempPw)==0){

result = 1; // berhasil masuk

foundpw = 1;

}found = 1;

}

}if(found==0){

result = 0; // wrong username

}else if(found==1 && foundpw==0){

result = 2; // wrong password

}

fclose(admin);

return result;

}

void loginMenu(){

char username[11]; char password[11];

do{

system("cls");

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

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

printf(" || ||\n");

printf(" || Masukkan username dan password untuk masuk ke program ||\n");

printf(" || ||\n");

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

printf(" Username : "); scanf("%[^\n]", username); getchar();

printf(" Password : "); scanf("%[^\n]", password); getchar();

if(cekAdmin(username, password)==2){

printf("Wrong password!\n");

system("pause");

}else if(cekAdmin(username, password)==0){

printf("Wrong username!\n");

system("pause");

}

}while(cekAdmin(username, password)!=1);

printf("Hello, %s \3\n", username);

printf("Berhasil masuk !\n");

system("pause");

}

int main() {

system("COLOR 0C");

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

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

printf(" || ||\n");

printf(" || Selamat Datang di Program Gudang Toko Sembako 45 !!! \2 ||\n");

printf(" || ||\n");

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

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

getch();

system("cls");

loginMenu();

while (1) {

menu();

}

return 0;

}

1. **REFERENCES**

Ide Code :

<https://www.programiz.com/dsa/binary-search-tree>

<https://www.digitalocean.com/community/tutorials/binary-search-tree-bst-search-insert-remove>

1. **OTHER**

Job Description :

1. Monica Wahyu Sayyidina (Nilai : XY)

* Code main dan tampilan
* Code fitur bersama (input, delete, update)
* Laporan bagian bab 1, 2, 3, 4 (kerja bareng)

1. Rahel Andrea Arabella

* Code search data
* Code fitur bersama (input, delete, update)
* Laporan bagian bab 1, 2, 3, 4 (kerja bareng)

1. Nabila Shifa Putri Farhanah (Nilai : XY)

* Code validasi
* Code fitur bersama (input, delete, update)
* Laporan bagian bab 1, 2, 3, 4 (kerja bareng)

**REVISI PROGRAM**

* + - 1. Tampilan menu loading dihapus
      2. Menu delete ditambah dengan pemilihan dan konfirmasi sebelum benar dihapus
      3. Sanitasi untuk penginputan nama produk, jumlah barang maupun harga barang