**<Implementasi Program Laporan Toko Sembako dengan Metode Hashing>**

LAPORAN PROYEK AKHIR

MATA KULIAH COMP6360004 – DATA STRUCTURES

KELAS < BD20 - LAB>

Sebuah gambar berisi cuplikan layar, Grafis, diagram, teks

Deskripsi dibuat secara otomatis

Oleh :

<2602126896> - <KEVIN HUSODO>

<2602080636> - <KEVIN FARRELL FEBRIANUS TANAMATRAS>

Semester [Ganjil, Genap] <2022 - 2023>

MALANG

**LEMBAR PERSETUJUAN PROYEK AKHIR**

**<Implementasi Program Laporan Toko Sembako dengan Metode Hashing>**

MATA KULIAH COMP6362004 – DATA STRUCTURES

KELAS < BD20 - LAB >

Semester [Ganjil, Genap] <2022,2023>

Laporan akhir proyek ini adalah benar karya kami :

**(<KEVIN HUSODO >) (<KEVIN FARRELL FEBRIANUS TANAMATRAS >)**

**<26021268960> <2602080636>**

**Malang, …… - ……………. ……..**

**(<ELIZABETH PASKAHLIA GUNAWAN, S.Kom., M.Cs.>)**

**<D5707>**

**BAB 1**

**Latar Belakang**

Kami telah merencanakan dan mengembangkan sebuah program untuk melacak dan mencetak laporan pemasok barang di toko sembako menggunakan struktur data kunci hash. Latar belakang kami dalam mengembangkan program ini didasarkan pada kebutuhan dan tantangan yang dihadapi oleh pemilik dan pengelola toko sembako dalam mengelola data pemasok barang.

Toko sembako adalah bisnis yang memasok berbagai macam barang kebutuhan sehari-hari kepada konsumen. Sebagai pemilik atau pengelola toko sembako, penting bagi Anda untuk dapat mengelola dan melacak informasi tentang pemasok barang dengan efisien. Beberapa tantangan yang sering dihadapi dalam hal ini adalah:

1. Volume Data yang Besar: Toko sembako seringkali memiliki banyak pemasok yang memasok berbagai jenis barang. Mengelola dan melacak data pemasok secara manual dapat memakan waktu dan rentan terhadap kesalahan. Oleh karena itu, diperlukan sebuah sistem yang dapat menyimpan dan mengakses data pemasok dengan cepat dan efisien.
2. Pengelompokan dan Pengurutan Data: Data pemasok barang perlu dikelompokkan dan diurutkan berdasarkan kriteria tertentu, seperti nama pemasok atau jenis barang yang disuplai. Dengan banyaknya pemasok dan barang yang berbeda, mengelompokkan dan mengurutkan data secara manual menjadi sangat sulit dan tidak efisien. Sebuah sistem yang dapat melakukan pengelompokan dan pengurutan data secara otomatis akan sangat membantu dalam mempermudah pencarian informasi.
3. Laporan Pemasok Barang: Pemilik atau pengelola toko sembako seringkali membutuhkan laporan mengenai pemasok barang untuk keperluan analisis dan pengambilan keputusan. Laporan tersebut dapat mencakup informasi seperti jumlah barang yang disuplai oleh setiap pemasok, total pembayaran kepada pemasok, atau informasi lainnya yang relevan. Membuat laporan secara manual dapat memakan waktu dan tidak efisien.

Dengan mempertimbangkan tantangan dan kebutuhan yang dihadapi oleh pemilik dan pengelola toko sembako terkait data pemasok barang, kami memutuskan untuk mengembangkan program ini. Program ini dirancang untuk memberikan solusi yang mudah dan efisien dalam mengelola dan melacak data pemasok barang di toko sembako. Dengan menerapkan struktur data kunci hash, program ini memungkinkan pengguna untuk dengan cepat dan efisien mencari, mengelompokkan, mengurutkan, serta mencetak laporan pemasok barang berdasarkan kriteria tertentu.

Program yang kami kembangkan berfokus pada pengelolaan data di toko sembako. Kami menyediakan fitur-fitur berikut:

1. Data Laporan: Fitur ini memungkinkan pengguna untuk memasukkan data laporan baru ke dalam sistem. Pengguna dapat memasukkan tanggal transaksi, nama pemasok, jenis barang, nama barang, jumlah barang, harga barang, dan total pembayaran. Setelah memasukkan data, sistem akan menyimpannya menggunakan struktur data kunci hash.
2. Cetak Laporan: Fitur ini memungkinkan pengguna untuk mencetak laporan berdasarkan kriteria tertentu. Pengguna dapat memilih untuk mencetak laporan berdasarkan tanggal transaksi atau jumlah barang. Sistem akan mengambil data yang sesuai dari struktur data kunci hash dan mencetak laporan sesuai dengan format yang ditentukan.
3. Update Laporan: Fitur ini memungkinkan pengguna untuk memperbarui catatan laporan yang sudah ada. Pengguna dapat memilih laporan yang ingin diperbarui berdasarkan pengidentifikasi unik. Setelah itu, pengguna dapat mengubah data yang diperlukan, seperti tanggal transaksi, jumlah barang, harga barang, dan total pembayaran. Sistem akan memperbarui catatan laporan sesuai dengan perubahan yang dilakukan.
4. Cari Laporan: Fitur ini memungkinkan pengguna untuk mencari laporan penjualan berdasarkan pengidentifikasi unik. Pengguna dapat memasukkan pengidentifikasi unik tertentu, dan sistem akan mencari laporan yang sesuai dalam struktur data kunci hash.

Diharapkan dengan adanya program ini, pemilik dan pengelola toko sembako dapat mengoptimalkan pengelolaan data pemasok barang, meningkatkan efisiensi operasional, serta mendapatkan informasi yang relevan untuk pengambilan keputusan yang lebih baik dalam bisnis mereka.

**BAB 2**

**Tinjauan Pustaka**

1. Hash Table

Hash Table merupakan struktur data yang digunakan untuk menyimpan dan mengakses data dengan cepat melalui operasi pencarian, penambahan, dan penghapusan. yang menggunakan fungsi hash. contohnya sebuah kompiler yang menerjemahkan bahasa pemrograman yang berisi tabel simbol, di mana kunci dari sebuah elemen adalah string karakter arbitrer yang sesuai dengan pengidentifikasi dalam bahasa. (Thomas h. Cormen et al, 2009). Sebenarnya Hash table merupakan cara yang efektif untuk membuat kamus kosakata, walaupun pencarian dalam hash table mirip seperti linked list yang tergolong lama namun hashing memiliki performa yang sangat mumpuni.

1. Operasi dari Hash Table

Hash table sendiri merupakan cara yang efektif untuk mengimplementasikan mapping, dan operasinya mirip seperti linked list. (Aho, A. et al, 2009). sehingga operasi yang dapat dilakukan hash table adalah :

* Penyisipan (Insertion): Data dimasukkan ke dalam Hash Table menggunakan fungsi hash yang menghasilkan indeks.
* Pencarian (Search): Fungsi hash digunakan untuk mencari indeks yang sesuai dengan kunci dan memeriksa apakah data ada dalam Hash Table.
* Penghapusan (Deletion): Data dihapus dari Hash Table dengan menggunakan fungsi hash untuk mencari indeks dan menghapus data yang sesuai.

1. Division

Division adalah metode pemetaan yang memanfaatkan operasi pembagian dan sisa bagi (modulo). Metode ini menghitung alamat relatif (ADDR) dari rekord dengan menggunakan persamaan berikut:

ADDR = KEY MOD DIV (1)

Dalam persamaan tersebut, ADDR merupakan alamat relatif dari rekord, KEY adalah kunci (key) dari rekord tersebut, dan DIV adalah bilangan pembagi (prime number). Penggunaan bilangan pembagi yang merupakan bilangan prima memiliki keuntungan karena meminimalkan kemungkinan terjadinya konflik alamat (collision). Konflik alamat terjadi ketika dua kunci berbeda menghasilkan alamat yang sama setelah diterapkan operasi modulo.

Dalam division,proses hashing melibatkan langkah-langkah berikut:

1. Memperoleh kunci (KEY) dari rekord yang akan dimapping.
2. Melakukan operasi modulo (MOD) antara kunci (KEY) dan bilangan pembagi (DIV).
3. Menggunakan hasil modulo sebagai alamat relatif (ADDR) dari rekord.

Manfaat dari division sebagai hash function antara lain:

* Sederhana: Metode ini mudah dipahami dan diimplementasikan.
* Efisien: Proses pemetaan menggunakan operasi modulo relatif cepat.
* Pengurangan konflik: Dengan menggunakan bilangan pembagi yang merupakan bilangan prima, kemungkinan terjadinya konflik alamat dapat diminimalkan.

Namun, ada beberapa hal yang perlu diperhatikan saat menggunakan division:

* Memilih bilangan pembagi yang tepat: Bilangan pembagi harus dipilih dengan cermat untuk meminimalkan konflik alamat. Idealnya, bilangan pembagi harus berukuran relatif besar dan merupakan bilangan prima.
* Pengelompokan data yang tidak merata: Jika kunci (KEY) memiliki pola yang cenderung terkumpul pada beberapa nilai, dapat menyebabkan konflik alamat yang tinggi. Hal ini dapat mempengaruhi kinerja hash table.

Dalam penggunaan praktis, division dapat digunakan dalam implementasi struktur data hash table, di mana kunci (KEY) digunakan untuk mencari alamat fisik dari rekord yang sesuai. Dengan menggunakan metode ini, pencarian dan penyisipan data dapat dilakukan secara efisien dengan kompleksitas waktu yang relatif rendah.

1. Linear Probing

Linear Probing adalah salah satu metode yang digunakan untuk mengatasi kolisi atau benturan dalam hashing. Ketika terjadi kolisi, yaitu dua atau lebih rekord memiliki alamat yang sama setelah proses hashing, Linear Probing melakukan hash kedua terhadap nilai kunci dengan menambah satu secara linear sampai ditemukan alamat relatif yang kosong. Menurut jurnal yang dikutip (Ridho & Prianto, 2020), Linear Probing menggunakan rumus berikut untuk menghitung alamat relatif:

h'i(X) = (h'i-1(X) + 1) mod [ukuran tabel] + 1 (2)

Dalam rumus tersebut, h'i(X) adalah alamat relatif yang dihasilkan pada perhitungan awal, i adalah variabel yang bertambah satu secara linier, dan [ukuran tabel] merupakan ukuran tabel hash. Dalam proses Linear Probing, jika terjadi kolisi, maka akan dilakukan iterasi pada alamat berikutnya dengan menambahkan nilai i secara linier hingga ditemukan alamat relatif yang masih kosong. Dengan demikian, pencarian dapat langsung merujuk pada lokasi record yang sesuai.

Metode Linear Probing memiliki beberapa kelebihan dan kelemahan. Kelebihannya meliputi:

* Sederhana: Metode ini relatif mudah dipahami dan diimplementasikan.
* Efisien: Proses pencarian dapat dilakukan dengan cepat karena hanya melibatkan iterasi linear pada alamat berikutnya.
* Lokalitas yang baik: Rekord dengan kunci yang serupa cenderung terdapat pada alamat relatif yang berdekatan, sehingga dapat mengurangi waktu pencarian.

Namun, ada beberapa kelemahan dalam penggunaan Linear Probing, antara lain:

* Kemungkinan terjadinya clustering: Jika terdapat banyak kolisi, rekord-rekord dengan kunci yang serupa akan terkumpul dalam wilayah alamat yang berdekatan, yang dapat mengurangi efisiensi operasi penyisipan dan penghapusan data.
* Faktor beban: Jika faktor beban tabel hash mendekati atau melebihi kapasitas maksimal, performa Linear Probing dapat menurun drastis.

1. Sembako

Istilah "sembako" pertama kali populer setelah krisis mata uang pada tahun 1998 di Indonesia. Krisis tersebut menyebabkan ketidakstabilan ekonomi, dan sembako menjadi penting sebagai bahan pangan dasar yang memenuhi kebutuhan masyarakat. Sembako meliputi beras, gula, minyak, tepung, dan produk penting lainnya. Dalam situasi ini, toko sembako perlu memiliki sistem yang efisien untuk melacak dan mencetak laporan penjualan guna memastikan pasokan yang memadai bagi masyarakat.

Regulasi terkait sembako diberlakukan dalam Keputusan Menteri Perindustrian dan Perdagangan No.115/MPP/Kep/2/1998 tanggal 27 Februari 1998. Regulasi ini menjadikan sembako sebagai kebutuhan pokok yang harus tersedia, memberikan pedoman dan kerangka hukum dalam distribusi dan penyaluran sembako. Program yang dapat melacak dan mencetak laporan penjualan pada toko sembako akan membantu pemilik toko mematuhi regulasi ini dan memastikan distribusi sembako kepada masyarakat yang membutuhkan.

Paket sembako saat ini umumnya dibagikan sebelum Hari Raya Idul Fitri, terutama kepada masyarakat menengah ke bawah. Program yang dapat melacak dan mencetak laporan penjualan pada toko sembako akan membantu dalam memantau persediaan dan distribusi paket sembako tersebut. Dengan data penjualan yang terbarukan secara akurat, pemilik toko sembako dapat memastikan distribusi sembako yang tepat waktu dan efisien kepada masyarakat yang membutuhkan.

Sembako dianggap sebagai kebutuhan pokok sehari-hari manusia, sehingga manajemen inventori yang efisien menjadi sangat penting dalam toko sembako. Program yang menggunakan struktur data kunci hash dapat membantu pemilik toko sembako dalam melacak stok barang, memperbarui persediaan, dan menghasilkan laporan penjualan yang informatif. Dengan informasi yang terorganisir dengan baik, pemilik toko sembako dapat mengelola persediaan dengan lebih baik, memastikan ketersediaan sembako yang memadai, dan menghindari kekurangan stok yang berdampak negatif pada masyarakat.

**BAB 3**

**Definisi Program**

**3.1 Pseudocode dan Flowchart Program**

1. Pseudocode

CONSTANT MAX\_HASH\_SIZE = some\_value

struct sembako {

int hashKey

char pembeli[255]

long long harga

long long total\_harga

char barang[255]

char jenis[255]

long int tanggal

int jumlah\_barang

bool available

}

struct temp\_data\_info {

char pembeli[255]

long long harga

long long total\_harga

char barang[255]

char jenis[255]

long int tanggal

int jumlah\_barang

bool available

}

temp\_data\_info temps[MAX\_HASH\_SIZE]

sembako data[MAX\_HASH\_SIZE]

int tempCount = 0

int total\_data = 0

FILE\* sales

FUNCTION divisionHash(str)

FUNCTION linearProbing(hashKey, index

FUNCTION divisionHashKey(str1, str2, tanggal

PROCEDURE readData()

PROCEDURE loading()

PROCEDURE login()

PROCEDURE jam()

PROCEDURE depan()

PROCEDURE dataPenjualan()

PROCEDURE cetakLaporanTanggal()

PROCEDURE cetakLaporanJumlah()

PROCEDURE cariLaporan()

PROCEDURE updateFile()

PROCEDURE updateLaporan()

PROCEDURE showUpdateChoices()

PROCEDURE updateMenuChoice(i)

PROCEDURE menu()

MAIN

Berikut adalah penjelasan singkat tentang setiap bagian pseudocode tersebut:

1. Struct sembako dan temp\_data\_info adalah struktur data yang digunakan untuk menyimpan informasi penjualan sembako. Masing-masing memiliki beberapa variabel yang merepresentasikan atribut-atribut penjualan seperti hashKey, pembeli, harga, total\_harga, barang, jenis, tanggal, jumlah\_barang, dan available.
2. temps dan data adalah array yang digunakan untuk menyimpan data penjualan. temps adalah array dari tipe temp\_data\_info, sedangkan data adalah array dari tipe sembako. tempCount adalah variabel yang digunakan untuk melacak jumlah data sementara yang ditambahkan, sedangkan total\_data melacak total data yang ada.
3. sales adalah file pointer yang digunakan untuk membuka dan menulis ke file "sales.txt".
4. Fungsi divisionHash digunakan untuk menghitung nilai hash dari string input menggunakan metode division hashing. Ini akan menghasilkan indeks dalam rentang 0 hingga MAX\_HASH\_SIZE - 1.
5. Fungsi linearProbing digunakan untuk menangani tabrakan atau collision dalam metode hashing yang digunakan. Ini mengembalikan indeks baru dengan menggunakan linear probing untuk mencari slot yang tersedia.
6. Fungsi divisionHashKey digunakan untuk menghasilkan hash key unik berdasarkan atribut-atribut tertentu dari data penjualan. Ini menggabungkan tanggal, nama pembeli, dan jenis barang menjadi satu string unik. Kemudian, itu akan menghitung nilai hash dari string tersebut menggunakan fungsi divisionHash dan menggunakan linear probing jika terjadi tabrakan.
7. readData adalah prosedur yang membaca data dari file "sales.txt" dan memasukkannya ke dalam array data.
8. loading adalah prosedur yang menampilkan animasi loading sederhana dan menunggu beberapa waktu sebelum membersihkan layar.
9. login adalah prosedur yang meminta pengguna untuk memasukkan nama pengguna dan kata sandi. Ini membandingkan input pengguna dengan nilai yang diharapkan dan memberikan tiga kesempatan untuk login sebelum program keluar.
10. jam adalah prosedur yang menampilkan waktu saat ini.
11. depan adalah prosedur yang menampilkan menu utama aplikasi.
12. dataPenjualan adalah prosedur yang meminta pengguna untuk memasukkan data penjualan sembako baru. Ini akan meminta input untuk atribut-atribut penjualan seperti tanggal, nama pembeli, jenis barang, dll. Kemudian, itu akan menghitung hash key menggunakan divisionHashKey dan menyimpan data penjualan ke dalam array data dan juga menulisnya ke file "sales.txt".
13. cetakLaporanTanggal adalah prosedur yang mencetak laporan penjualan sembako yang diurutkan berdasarkan tanggal transaksi.
14. cetakLaporanJumlah adalah prosedur yang mencetak laporan penjualan sembako yang diurutkan berdasarkan jumlah barang terjual.
15. cariLaporan adalah prosedur yang memungkinkan pengguna mencari laporan penjualan dengan memasukkan hash key yang diinginkan. Ini akan mencari data yang sesuai
16. updateFile adalah prosedur yang digunakan untuk menyimpan perubahan data penjualan ke file "sales.txt". Ini membuka file dengan mode tulis ("w") dan menulis ulang semua data penjualan ke dalam file.
17. updateLaporan adalah prosedur yang memungkinkan pengguna untuk memperbarui data penjualan. Pengguna diminta untuk memasukkan hash key dari data yang ingin diperbarui. Jika data ditemukan, pengguna akan diberikan opsi untuk memilih atribut mana yang ingin diperbarui. Setelah memilih, atribut yang dipilih akan diperbarui dalam array data dan file "sales.txt" akan diperbarui menggunakan updateFile.
18. showUpdateChoices adalah prosedur yang menampilkan opsi-opsi yang tersedia untuk pembaruan data penjualan.
19. updateMenuChoice adalah prosedur yang menangani pemilihan pengguna untuk memperbarui atribut tertentu dari data penjualan. Berdasarkan pilihan pengguna, atribut yang sesuai akan diperbarui dalam array data.
20. menu adalah prosedur utama yang mengatur alur program. Ini menampilkan menu utama dan meminta pengguna untuk memilih opsi. Berdasarkan pilihan pengguna, fungsi dan prosedur yang sesuai akan dipanggil.
21. Pada main, program dimulai dengan memanggil fungsi loading untuk tampilan awal. Kemudian, fungsi login dipanggil untuk otentikasi pengguna. Setelah login berhasil, data penjualan dibaca dari file menggunakan readData. Selanjutnya, fungsi menu dipanggil untuk menampilkan menu utama dan menjalankan logika program sesuai dengan pilihan pengguna.
22. Flowchart

Sebuah gambar berisi teks, hitam dan putih, tulisan tangan, diagram

Deskripsi dibuat secara otomatis

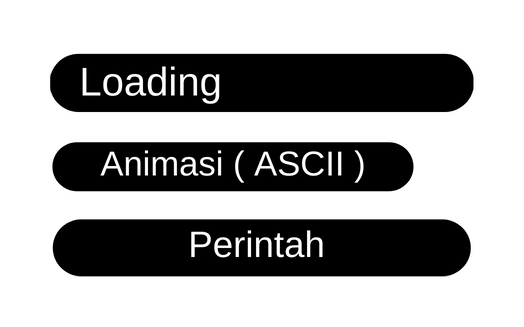
**3.2 Tinjauan Program**

1. Deskripsi Program

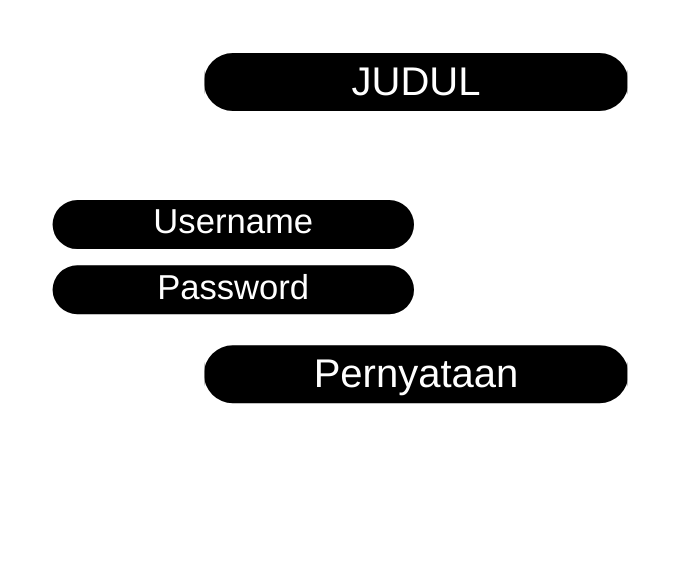
Program ini merupakan sebuah aplikasi untuk pendataan penjualan sembako di toko sembako “SembakoXpert". Aplikasi ini memiliki fitur untuk melakukan pendataan penjualan, pencetakan laporan penjualan berdasarkan tanggal transaksi atau jumlah terjual, pencarian laporan penjualan, dan pembaruan laporan penjualan. Program SembakoXpert juga menggunakan struktur data hash untuk menyimpan data penjualan.

1. Desain Tata Letak

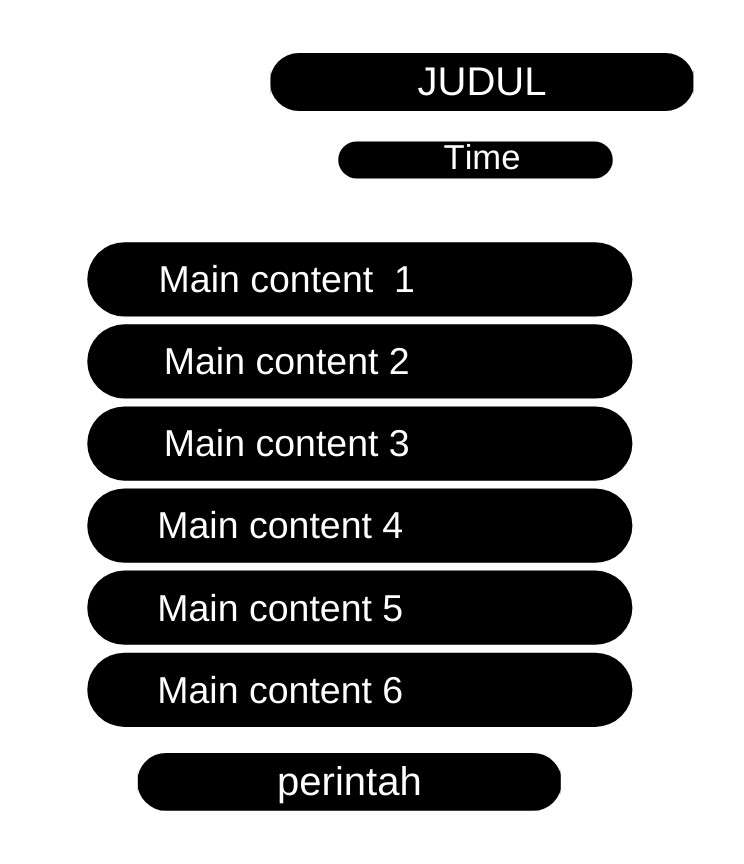
Tampilan awal yang beisi animasi dan perintah, dimana perintah yang akan dilakukan membawa ke tampilan berikutnya.



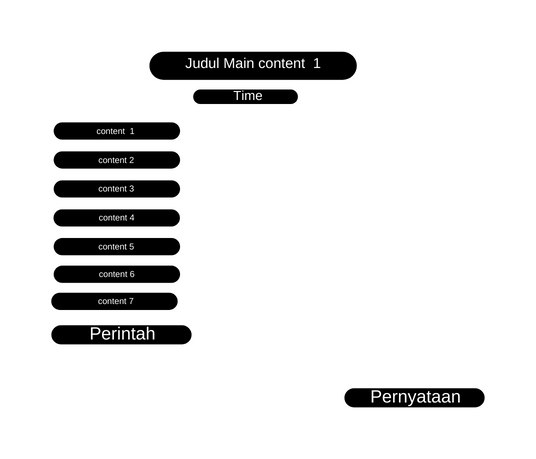
Pada design ini, ada judul, ada username dan password yang akan diinput, dimana akan mengeluarkan sebuah pernyataan sesuai dengan hasil inputan yang dimasukan.

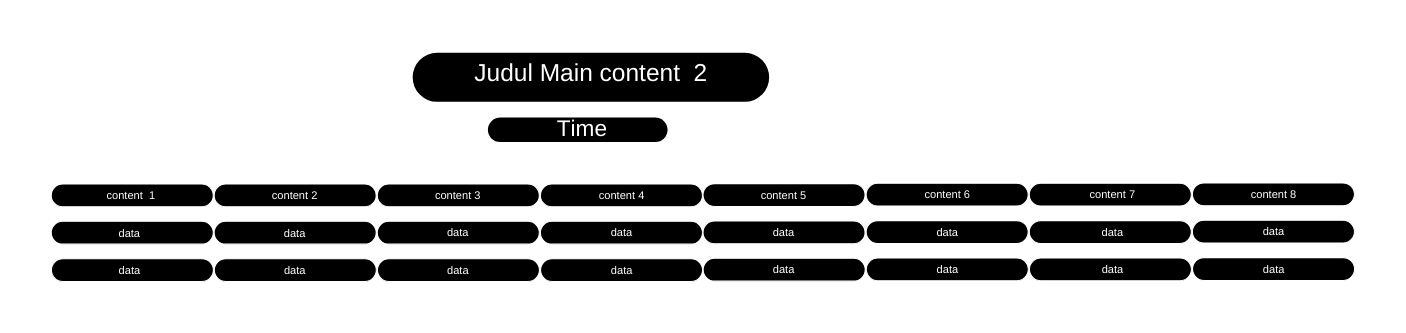


Pada bagian ini, ada main content yang berisi menu dan ada perintah yang diminta untuk memilih main content yang mana akan dipilih.

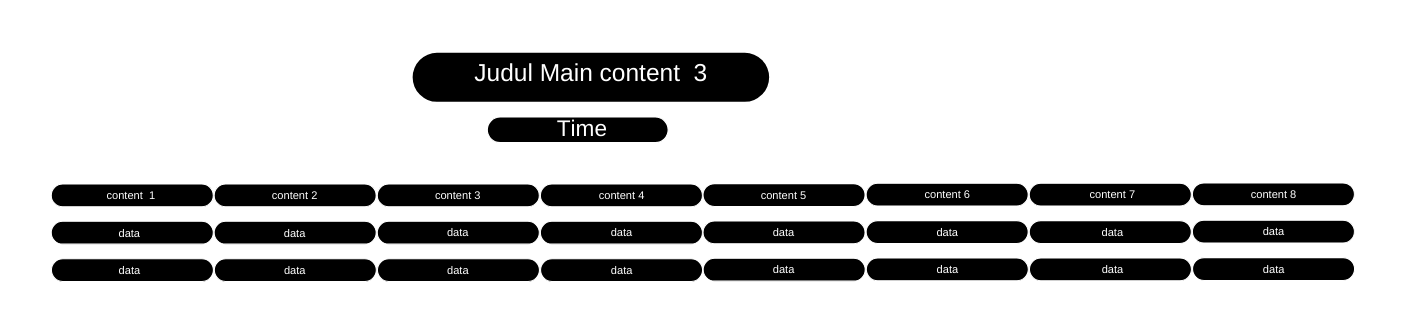


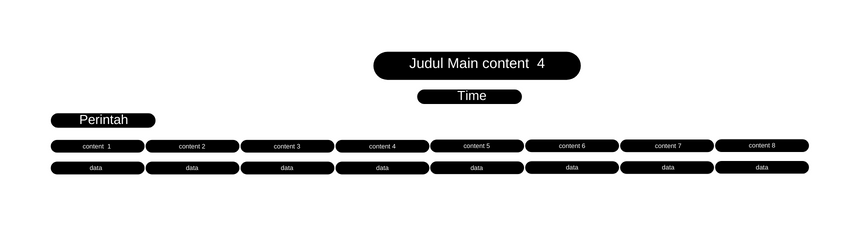
Pada bagian ini masuk ke main content 1, yang berisi judul, waktu, input data dan pernyataan.

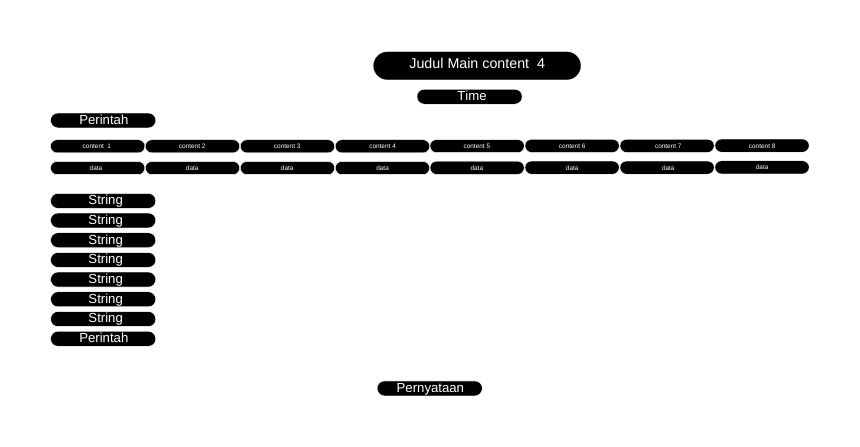


Pada bagian ini masuk ke main content 2, yang berisi judul, waktu, content dari main content dan ada data dari masing - masing content.   


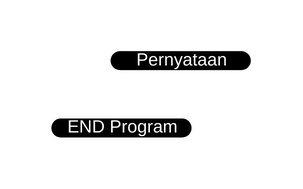
Pada bagian ini masuk ke main content 3, yang berisi judul, waktu, content dari main content dan ada data dari masing - masing content.



Pada bagian ini masuk ke main content 4, yang berisi judul, waktu, perintah, content dari main content dan ada data dari masing - masing content.  


Pada bagian ini masuk ke main content 5, yang berisi judul, waktu, perintah, content dari main content dan ada data dari masing - masing content, dan selanjutnya, berisi string yang diiringi sebuah perintah dan sebuah pernyataan.   


Pada bagian ini masuk ke main content 6, yang pernyataan dan end.



1. Fitur Program

1. Pendataan Penjualan: Fitur ini memungkinkan pengguna untuk memasukkan data penjualan bahan sembako, seperti tanggal transaksi, nama pembeli, jenis barang, nama barang, jumlah barang, harga barang, dan total pembayaran. Data penjualan tersebut akan disimpan dalam file "sales.txt" untuk pengolahan dan cetak laporan.

2. Pencetakan Laporan Penjualan (Sort Tanggal Transaksi): Fitur ini akan mencetak laporan penjualan bahan sembako berdasarkan tanggal transaksi secara terurut. Data penjualan akan diurutkan berdasarkan tanggal transaksi secara ascending (dari yang terkecil ke terbesar) sebelum dicetak ke layar.

3. Pencetakan Laporan Penjualan (Sort Jumlah Terjual): Fitur ini akan mencetak laporan penjualan bahan sembako berdasarkan jumlah barang terjual secara terurut. Data penjualan akan diurutkan berdasarkan jumlah barang terjual secara descending (dari yang terbesar ke terkecil) sebelum dicetak ke layar.

4. Cari Laporan Penjualan: Fitur ini memungkinkan pengguna untuk mencari laporan penjualan berdasarkan hash key. Pengguna diminta memasukkan hash key yang ingin dicari, dan program akan mencari data penjualan yang sesuai dengan hash key tersebut. Jika ditemukan, data penjualan akan ditampilkan ke layar.

5. Update Laporan Penjualan: Fitur ini memungkinkan pengguna untuk mengubah data penjualan yang sudah ada. Pengguna diminta memasukkan hash key dari data penjualan yang ingin diubah, kemudian pengguna dapat memilih data mana yang ingin diubah, seperti tanggal, nama pembeli, jenis barang, nama barang, jumlah barang, atau harga barang. Setelah data diubah, program akan menyimpan perubahan tersebut ke file "sales.txt".

6. Exit: Fitur ini akan menghentikan program dan keluar dari aplikasi.

1. Detail program

Berikut ini adalah pembahasan detail mengenai program tersebut:

1. Library yang Digunakan:

* stdio.h: Library standar untuk fungsi input/output dalam bahasa C.
* time.h: Library yang digunakan untuk mengakses fungsi-fungsi terkait waktu.
* string.h: Library yang digunakan untuk manipulasi string.
* windows.h: Library khusus untuk sistem operasi Windows yang memberikan akses ke fungsi-fungsi Windows API.
* conio.h: Library yang digunakan untuk fungsi input/output khusus pada console.

1. Struktur Data:

* sembako: Struktur data yang digunakan untuk menyimpan informasi penjualan satu unit sembako.
* hashKey: Kunci hash yang digunakan untuk mengindeks data.
* pembeli: Nama pembeli.
* harga: Harga satu unit barang sembako.
* total\_harga: Total harga dari jumlah barang sembako yang dibeli.
* barang: Nama barang sembako.
* jenis: Jenis barang sembako.
* tanggal: Tanggal transaksi penjualan.
* jumlah\_barang: Jumlah barang sembako yang dibeli.
* available: Status ketersediaan data.
* temp\_data\_info: Struktur data sementara yang digunakan untuk menyimpan informasi penjualan yang akan diupdate. Mirip dengan struktur sembako, digunakan untuk menyimpan data sementara sebelum diupdate.

1. temp\_data\_info temps[MAX\_HASH\_SIZE]; dan sembako data[MAX\_HASH\_SIZE];  
   Variabel temps dan data merupakan dua buah array yang berukuran MAX\_HASH\_SIZE. MAX\_HASH\_SIZE kemungkinan adalah sebuah konstanta yang menentukan ukuran maksimum dari kedua array tersebut.
2. int tempCount = 0; dan int total\_data = 0;  
   Variabel tempCount digunakan untuk menyimpan jumlah data sementara yang telah dimasukkan, sedangkan total\_data digunakan untuk menyimpan jumlah total data yang telah dibaca dari file.
3. FILE\* sales;  
    Variabel sales merupakan pointer ke objek file yang akan digunakan untuk membuka dan menutup file.
4. void dataPenjualan();, void cetakLaporan();, void updateLaporan();, void carilaporan();, void depan();, void menu();, void login();, void loading();, void jam();, void readData();, void updateMenuChoice(int res);  
    Ini adalah deklarasi dari beberapa fungsi yang akan digunakan dalam program. Fungsi-fungsi ini akan didefinisikan di bawah fungsi main().
5. int divisionHash(const char\* str)  
   Ini adalah sebuah fungsi untuk menghitung nilai hash dari sebuah string. Fungsi ini mengiterasi melalui karakter-karakter dalam string dan menjumlahkan nilai ASCII dari setiap karakter. Hasilnya kemudian dimodulo dengan MAX\_HASH\_SIZE untuk menghasilkan nilai hash yang sesuai.
6. int linearProbing(int hashKey, int index)  
   Fungsi linearProbing digunakan untuk melakukan probing secara linier jika terjadi konflik hash. Fungsi ini menerima parameter hashKey, yaitu nilai hash yang digunakan sebagai referensi, dan index, yaitu indeks saat ini. Fungsi ini mengembalikan indeks baru yang dihasilkan setelah melakukan probing linier.
7. int divisionHashKey(const char\* str1, const char\* str2, long int tanggal)  
   Fungsi divisionHashKey digunakan untuk menghitung kunci hash berdasarkan string dan tanggal. Fungsi ini menggabungkan tanggal, str1, dan str2 menjadi satu string unik. Kemudian, menggunakan fungsi divisionHash untuk menghitung nilai hash dari string unik tersebut. Nilai hash tersebut kemudian dimodulo dengan MAX\_HASH\_SIZE untuk mendapatkan indeks awal. Jika terjadi konflik hash, fungsi linearProbing akan digunakan untuk mencari indeks yang valid.
8. int main():  
    Fungsi main() adalah fungsi utama dalam program. Di dalam fungsi ini terdapat beberapa langkah yang dilakukan:

* system("color B"); digunakan untuk mengatur warna latar belakang konsol menjadi biru.
* loading(); adalah pemanggilan fungsi loading(). Fungsi ini mungkin berisi tampilan animasi atau pesan yang akan ditampilkan sebelum program dim
* login(); adalah pemanggilan fungsi login(). Fungsi ini mungkin berisi proses autentikasi pengguna atau login ke sistem.
* readData(); adalah pemanggilan fungsi readData(). Fungsi ini digunakan untuk membaca data dari file dan menyimpannya ke dalam array data. Jumlah total data yang telah dibaca akan disimpan dalam variabel total\_data.
* menu(); adalah pemanggilan fungsi menu(). Fungsi ini mungkin berisi tampilan menu utama dan logika pemrosesan pilihan menu yang diinputkan pengguna.
* fclose(sales); digunakan untuk menutup file yang telah dibuka sebelumnya menggunakan fopen.
* return 0; adalah pernyataan yang mengindikasikan bahwa program berakhir dengan sukses dan nilai kembali 0 akan dikirimkan ke sistem operasi.

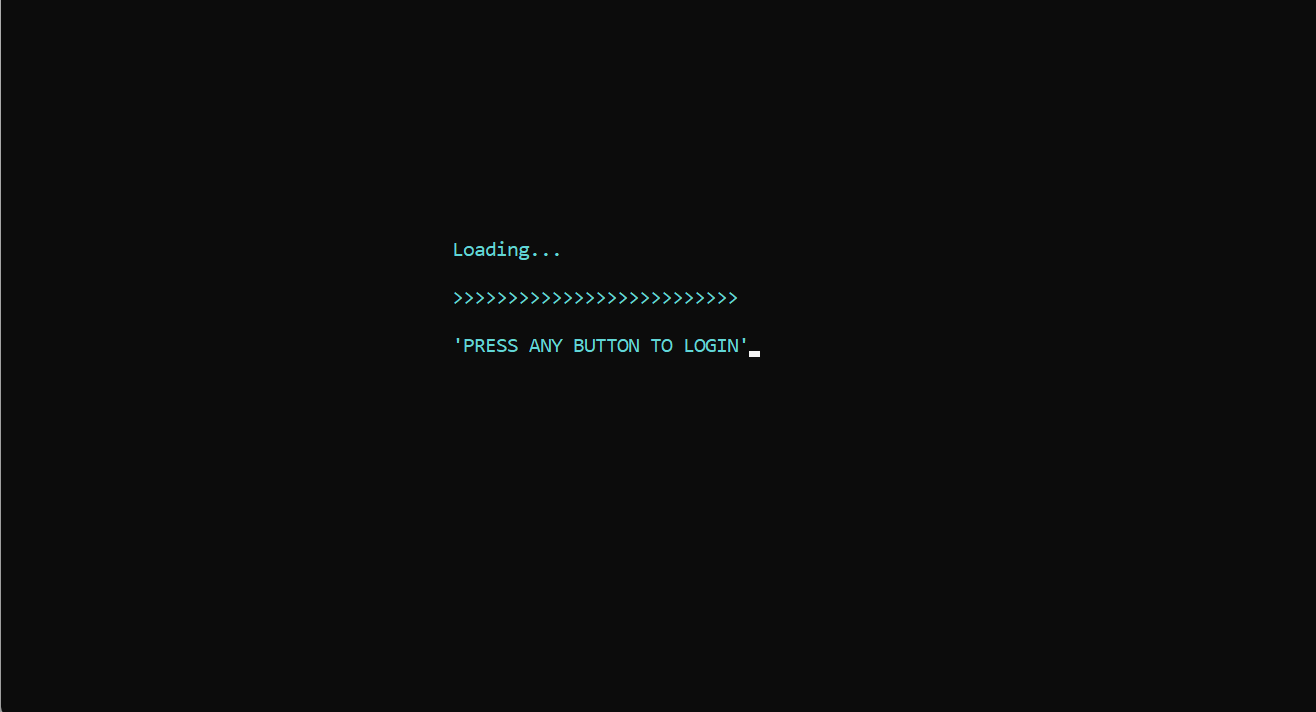
1. void readData()  
   Fungsi readData() digunakan untuk membaca data dari file "sales.txt" dan menyimpannya dalam array data. Fungsi ini membuka file dengan mode "r" (read) menggunakan fopen. Kemudian, menggunakan fscanf, fungsi ini membaca data dari file dengan format tertentu dan menyimpannya ke dalam elemen-elemen array data. Setiap kali data berhasil dibaca, variabel total\_data akan diincrement. Fungsi ini akan terus membaca data hingga mencapai akhir file (EOF).
2. void gotoxy(int x, int y)  
    Fungsi gotoxy() digunakan untuk memindahkan kursor ke posisi tertentu di layar konsol. Koordinat x dan y yang diberikan menentukan posisi kursor yang diinginkan. Fungsi ini menggunakan fungsi SetConsoleCursorPosition() dari Windows API untuk mengatur posisi kursor.
3. void depan()  
   Fungsi depan() digunakan untuk menampilkan tampilan depan atau halaman utama program. Fungsi ini membersihkan layar konsol menggunakan system("cls") dan kemudian mencetak judul program serta opsi menu yang tersedia. Fungsi jam() digunakan untuk mencetak informasi waktu saat ini.
4. void swap(sembako\* xp, sembako\* yp)  
   Fungsi swap() digunakan untuk menukar nilai antara dua elemen dalam array data dengan tipe data sembako. Fungsi ini menerima dua parameter pointer ke elemen-elemen yang akan ditukar. Fungsi ini berguna untuk membantu dalam pengurutan data dalam fungsi cetakLaporanTanggal() dan cetakLaporanJumlah().
5. void cetakLaporanTanggal()  
   Fungsi cetakLaporanTanggal() digunakan untuk mencetak laporan penjualan bahan sembako yang diurutkan berdasarkan tanggal transaksi. Fungsi ini melakukan pengurutan data berdasarkan tanggal menggunakan algoritma selection sort. Setelah data diurutkan, fungsi ini mencetak judul tabel dan data penjualan yang telah diurutkan.
6. void cetakLaporanJumlah()  
   Fungsi cetakLaporanJumlah() digunakan untuk mencetak laporan penjualan bahan sembako yang diurutkan berdasarkan jumlah terjual. Fungsi ini melakukan pengurutan data berdasarkan jumlah barang terjual menggunakan algoritma selection sort. Setelah data diurutkan, fungsi ini mencetak judul tabel dan data penjualan yang telah diurutkan.
7. void readData()  
   Fungsi readData() digunakan untuk membaca data dari file "sales.txt" dan menyimpannya dalam array data. Fungsi ini membuka file dengan mode "r" (read) menggunakan fopen. Kemudian, menggunakan fscanf, fungsi ini membaca data dari file dengan format tertentu dan menyimpannya ke dalam elemen-elemen array data. Setiap kali data berhasil dibaca, variabel total\_data akan diincrement. Fungsi ini akan terus membaca data hingga mencapai akhir file (EOF).
8. void gotoxy(int x, int y)  
   Fungsi gotoxy() digunakan untuk memindahkan kursor ke posisi tertentu di layar konsol. Koordinat x dan y yang diberikan menentukan posisi kursor yang diinginkan. Fungsi ini menggunakan fungsi SetConsoleCursorPosition() dari Windows API untuk mengatur posisi kursor.
9. void depan()  
   Fungsi depan() digunakan untuk menampilkan tampilan depan atau halaman utama program. Fungsi ini membersihkan layar konsol menggunakan system("cls") dan kemudian mencetak judul program serta opsi menu yang tersedia. Fungsi jam() digunakan untuk mencetak informasi waktu saat ini.
10. void swap(sembako\* xp, sembako\* yp)  
    Fungsi swap() digunakan untuk menukar nilai antara dua elemen dalam array data dengan tipe data sembako. Fungsi ini menerima dua parameter pointer ke elemen-elemen yang akan ditukar. Fungsi ini berguna untuk membantu dalam pengurutan data dalam fungsi cetakLaporanTanggal() dan cetakLaporanJumlah().
11. void cetakLaporanTanggal()  
    Fungsi cetakLaporanTanggal() digunakan untuk mencetak laporan penjualan bahan sembako yang diurutkan berdasarkan tanggal transaksi. Fungsi ini melakukan pengurutan data berdasarkan tanggal menggunakan algoritma selection sort. Setelah data diurutkan, fungsi ini mencetak judul tabel dan data penjualan yang telah diurutkan.
12. void cetakLaporanJumlah()  
    Fungsi cetakLaporanJumlah() digunakan untuk mencetak laporan penjualan bahan sembako yang diurutkan berdasarkan jumlah terjual. Fungsi ini melakukan pengurutan data berdasarkan jumlah barang terjual menggunakan algoritma selection sort. Setelah data diurutkan, fungsi ini mencetak judul tabel dan data penjualan yang telah diurutkan.
13. void loading()  
     Fungsi loading() digunakan untuk menampilkan animasi loading saat program sedang dimuat. Fungsi ini mencetak pesan "Loading..." dan kemudian menggunakan perulangan untuk mencetak garis-garis sebagai animasi loading. Setelah selesai, pesan "PRESS ANY BUTTON TO LOGIN" akan ditampilkan dan pengguna dapat menekan tombol apa pun untuk melanjutkan. Fungsi ini menggunakan fungsi Sleep() untuk memberikan jeda antara setiap perubahan dalam animasi dan system("cls") untuk membersihkan layar konsol.
14. void login()  
     Fungsi login() digunakan untuk proses login ke sistem. Fungsi ini memberikan tiga kesempatan login kepada pengguna. Pengguna diminta untuk memasukkan username dan password. Password yang diketikkan oleh pengguna ditampilkan sebagai karakter '\*' untuk keamanan. Jika username dan password yang dimasukkan adalah "admin", maka pesan "LOGIN SUKSES!!!" akan ditampilkan. Jika tidak, pengguna akan diberikan pesan bahwa password atau username yang dimasukkan salah. Jika pengguna gagal login sebanyak tiga kali, program akan keluar (exit(0)).
15. void jam()  
     Fungsi jam() digunakan untuk mencetak waktu saat ini menggunakan fungsi localtime() dan asctime() dari library time.h. Waktu saat ini diperoleh menggunakan fungsi time() dan kemudian diubah menjadi struktur waktu menggunakan localtime(). Fungsi asctime() digunakan untuk mengonversi struktur waktu menjadi string yang dapat dicetak.
16. void dataPenjualan()  
     Fungsi dataPenjualan() digunakan untuk mengambil input dari pengguna tentang data penjualan bahan sembako. Fungsi ini mencetak tampilan untuk pendataan penjualan, termasuk judul, waktu saat ini, dan formulir input. Pengguna diminta untuk memasukkan tanggal transaksi, nama pembeli, jenis barang, nama barang, jumlah barang, dan harga barang. Data yang dimasukkan oleh pengguna akan disimpan dalam elemen-elemen array data. Fungsi juga menghitung hash key menggunakan divisionHash() dan menghitung total pembayaran. Setelah data berhasil dimasukkan ke dalam array data, data juga akan ditulis ke file "sales.txt" menggunakan fprintf().
17. void cariLaporan()  
     Fungsi cariLaporan() digunakan untuk mencari laporan penjualan berdasarkan hash key yang diinputkan pengguna. Pengguna diminta untuk memasukkan hash key dalam format "d-ss" (misalnya: 1-a1). Fungsi ini akan mencari data penjualan yang sesuai dengan hash key yang diinputkan. Jika data ditemukan, fungsi akan mencetak data tersebut. Jika data tidak ditemukan, pesan "Data tidak ditemukan di daftar" akan dicetak.
18. void updateFile()  
    Fungsi updateFile() digunakan untuk memperbarui file "sales.txt" dengan data penjualan yang telah diperbarui. Fungsi ini membuka file dengan mode "w" (write) menggunakan fopen. Jika file berhasil dibuka, fungsi ini akan menulis ulang data penjualan ke dalam file menggunakan fprintf(). Setelah selesai, file ditutup menggunakan fclose().
19. void updateLaporan()  
    Fungsi updateLaporan() digunakan untuk memperbarui laporan penjualan yang ada. Fungsi ini meminta pengguna untuk memasukkan hash key dalam format yang sama seperti fungsi cariLaporan(). Fungsi ini akan mencari data penjualan yang sesuai dengan hash key yang diinputkan. Jika data ditemukan, fungsi akan mencetak data tersebut dan memanggil fungsi updateMenuChoice() untuk memperbarui pilihan menu. Setelah data diperbarui, fungsi updateFile() akan dipanggil untuk memperbarui file "sales.txt".
20. void showUpdateChoices()  
    Fungsi showUpdateChoices() digunakan untuk mencetak pilihan-pilihan yang tersedia untuk memperbarui data penjualan. Pilihan-pilihan tersebut mencakup Tanggal, Nama Pembeli, Jenis Barang, Nama Barang, Jumlah Barang, Harga Barang, dan Cancel.
21. void updateMenuChoice(int i)  
    Fungsi updateMenuChoice() digunakan untuk memproses pilihan yang dipilih oleh pengguna untuk memperbarui data penjualan. Fungsi ini menggunakan perulangan while untuk terus meminta pilihan dari pengguna hingga pilihan valid dipilih. Pilihan yang valid akan memicu pemanggilan fungsi scanf() untuk memperbarui data penjualan yang sesuai. Setelah data diperbarui, fungsi ini akan memanggil updateFile() untuk memperbarui file "sales.txt".
22. void menu()  
    Fungsi menu() adalah fungsi utama yang berisi tampilan menu utama dan logika pemrosesan pilihan menu yang diinputkan pengguna. Fungsi ini menggunakan perulangan while untuk menjaga program tetap berjalan hingga pengguna memilih pilihan "Exit". Pilihan menu diproses menggunakan pernyataan if-else if. Setiap pilihan menu akan memanggil fungsi yang sesuai untuk melaksanakan tugas yang diminta.

**BAB 4**

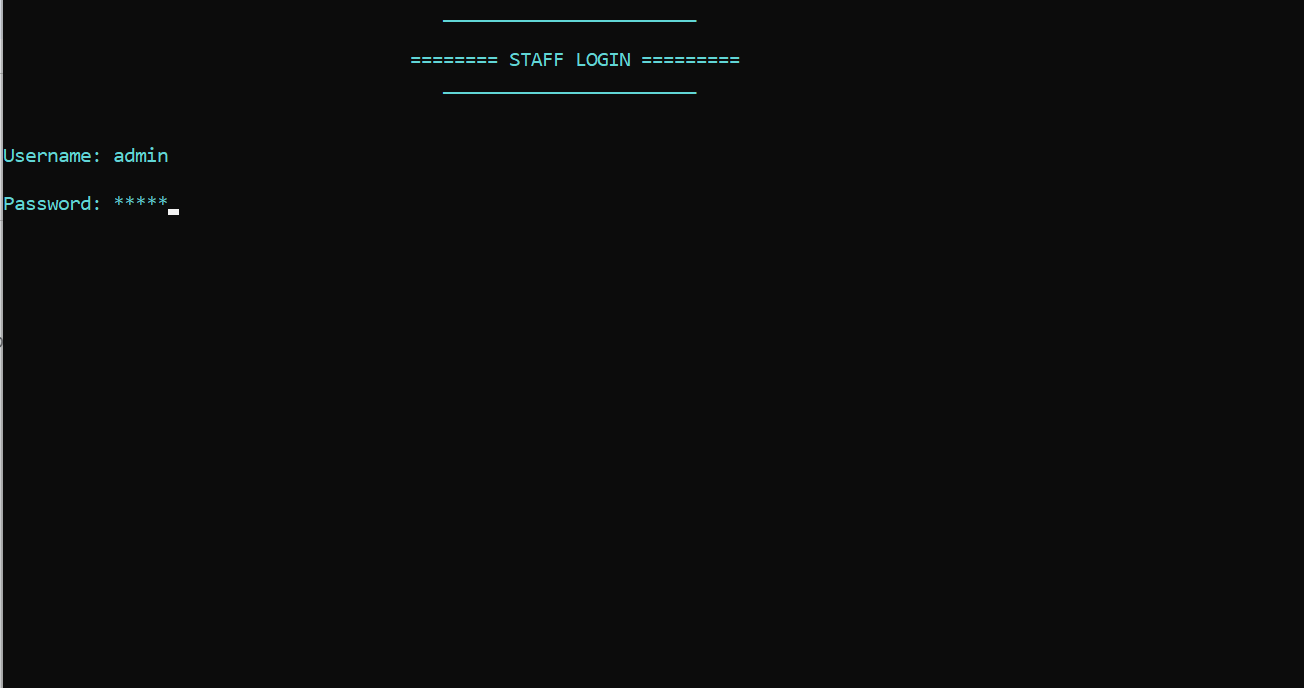
**Hasil Program**

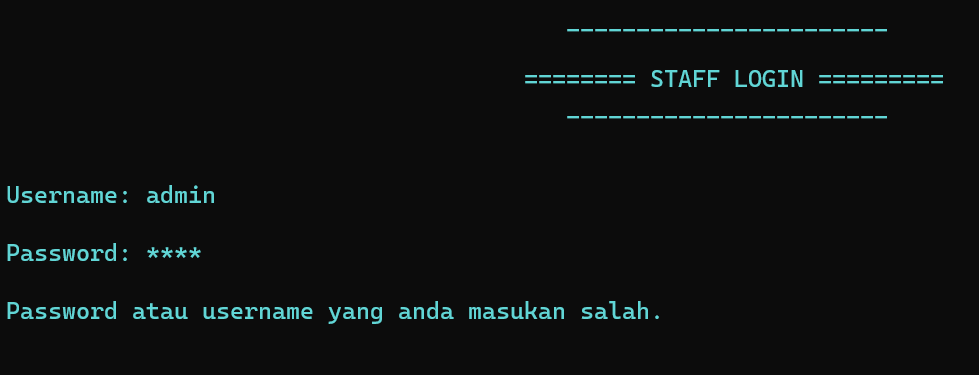
4.1 Hasil Program

Fungsi loading() digunakan untuk menampilkan tampilan loading saat program pertama kali dijalankan.

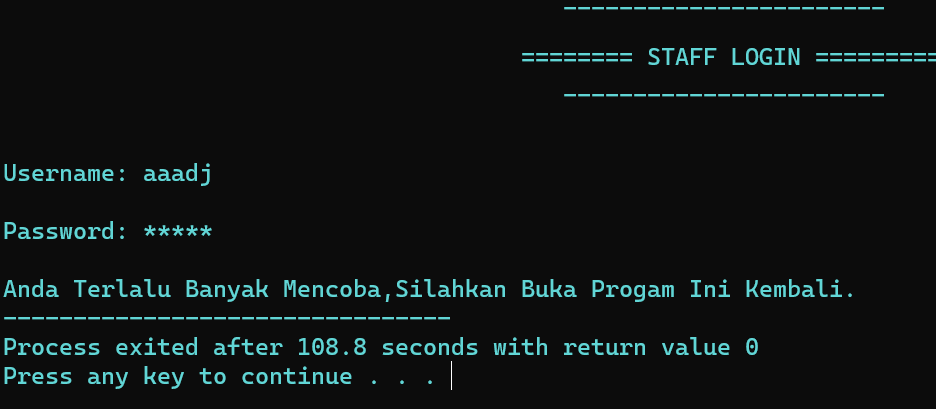


Fungsi login() digunakan untuk melakukan login sebagai admin program.





pada gambar diatas, jika pengguna memasukkan username atau password yang salah, maka akan muncul pesan "Password atau username yang anda masukkan salah."

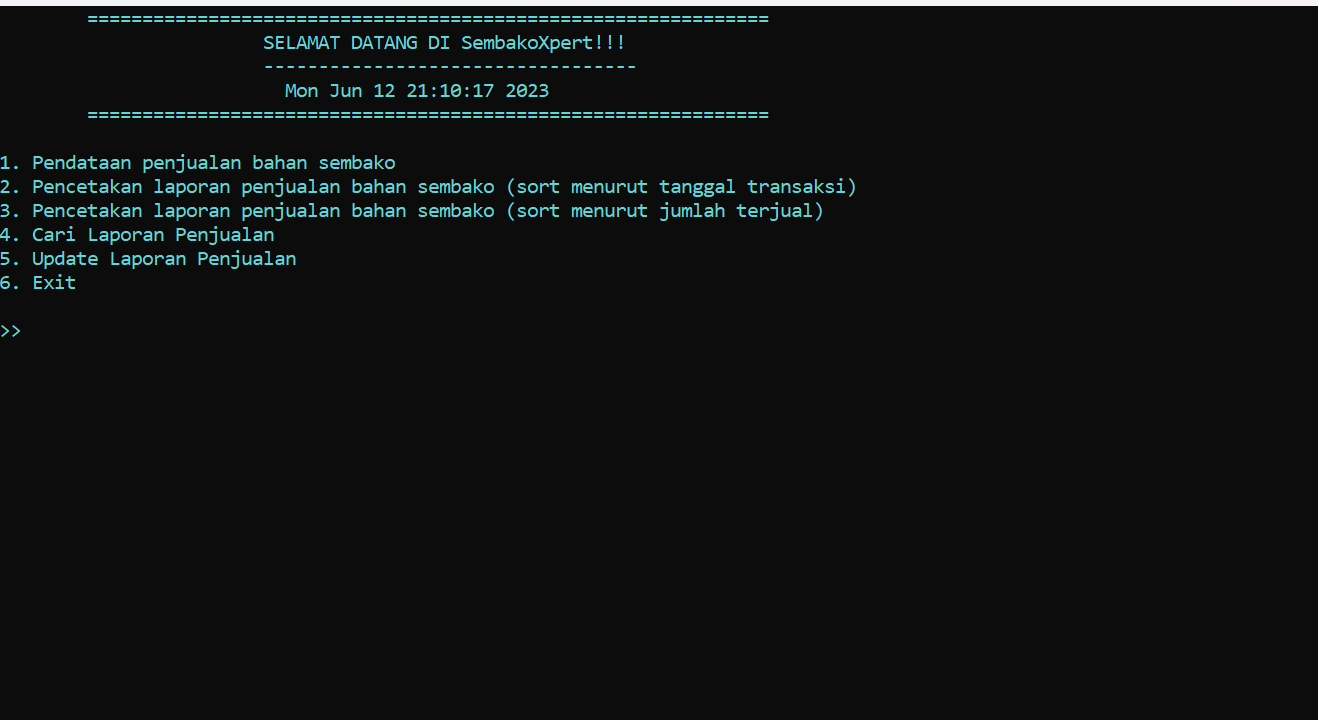


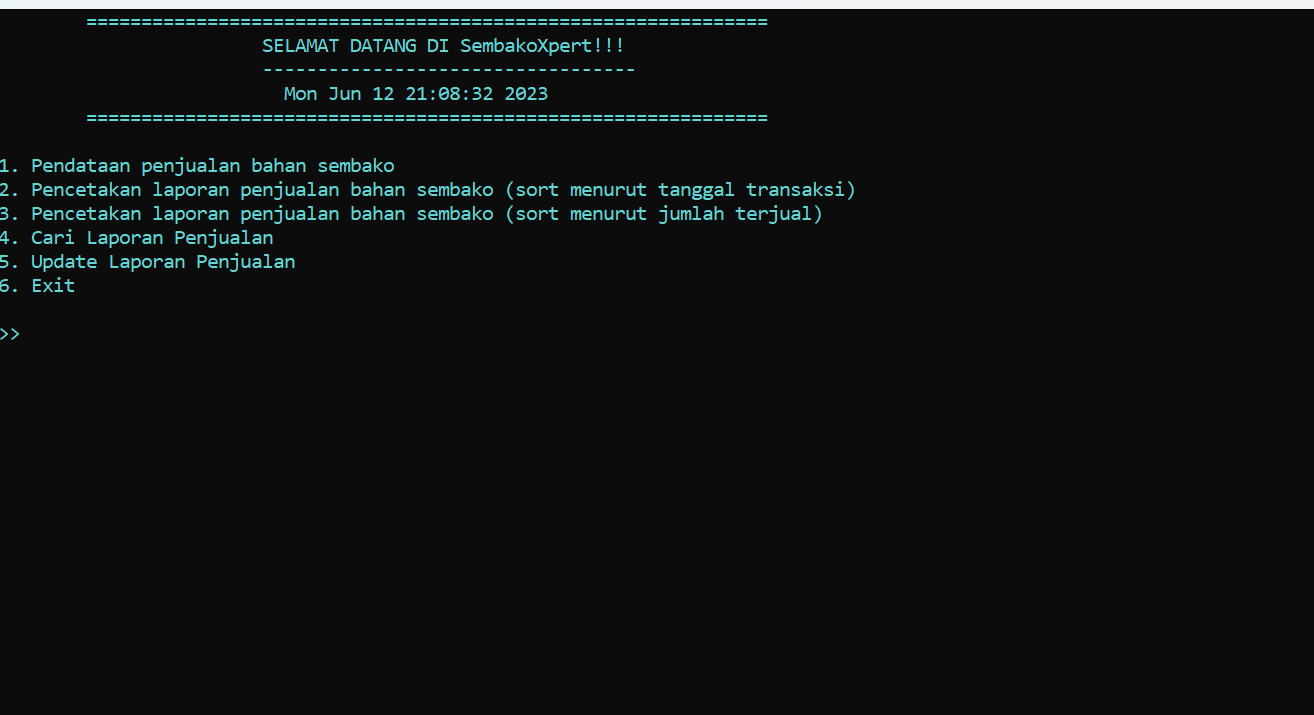
pada gambar diatas, jika pengguna telah mencoba memasukkan username dan password yang salah sebanyak 3 kali, maka akan muncul pesan "Anda Terlalu Banyak Mencoba, Silahkan Buka Program Ini Kembali."

Fungsi jam() digunakan untuk menampilkan waktu saat ini.



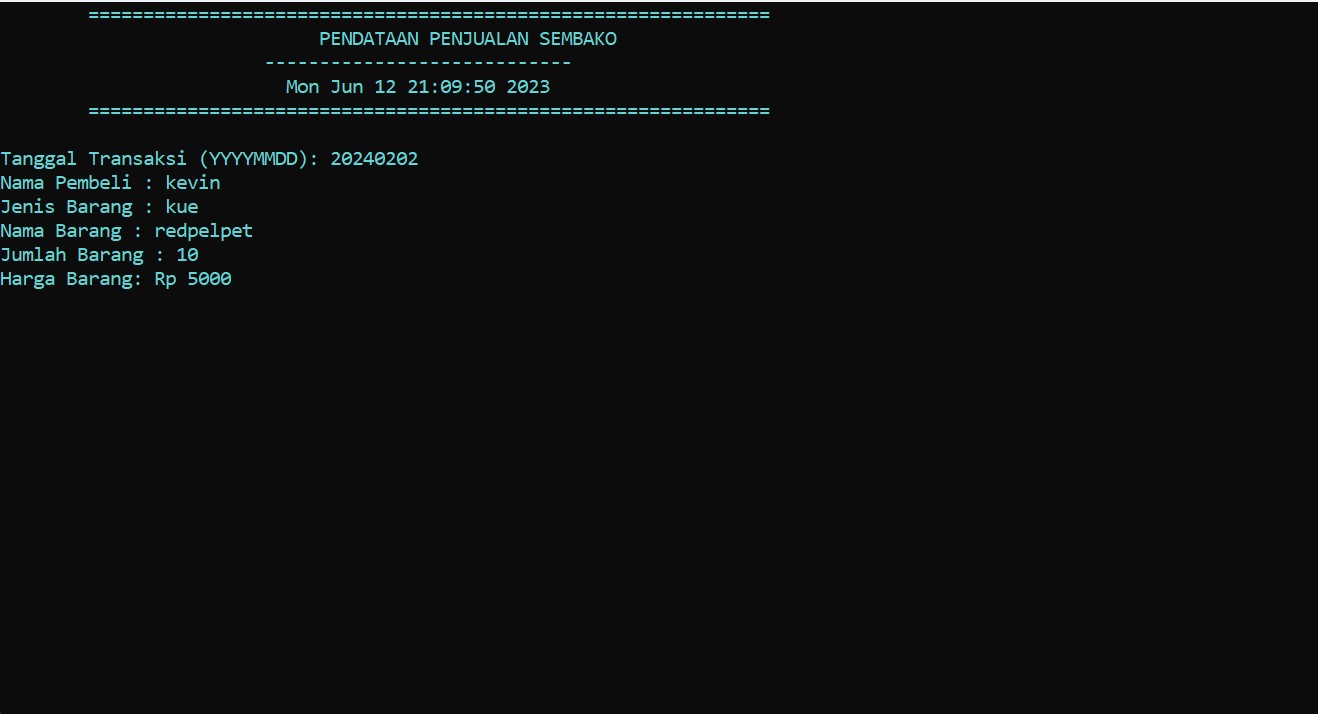
Fungsi depan() digunakan untuk menampilkan tampilan depan program.

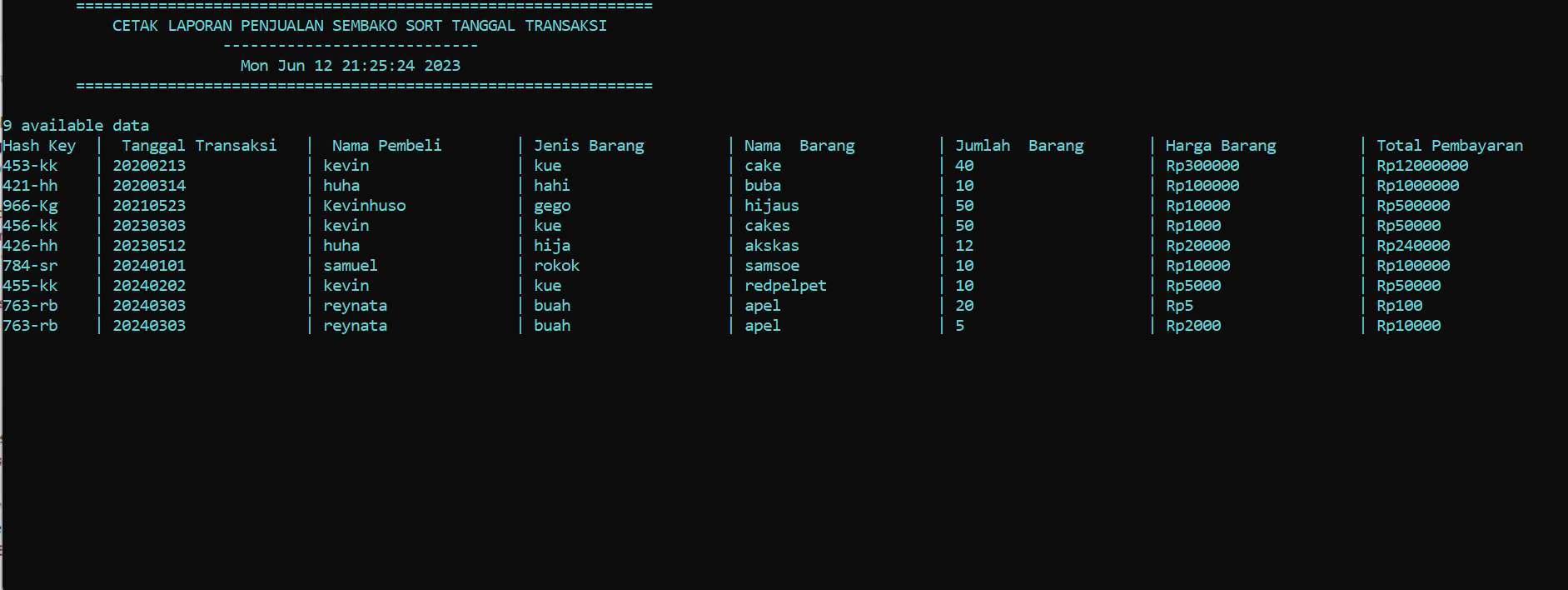




gambar diatas menjadi menu awal SembakoXpert.

Pendataan penjualan bahan bangunan: Fungsi dataPenjualan() digunakan untuk memasukkan data penjualan Sembako ke dalam program. Data penjualan tersebut akan disimpan menggunakan struktur data hash.

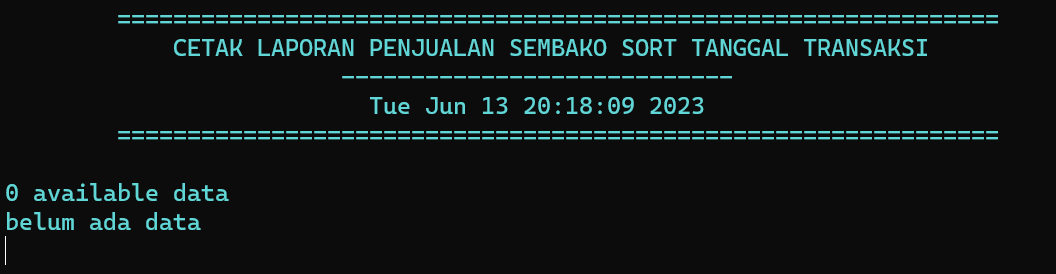




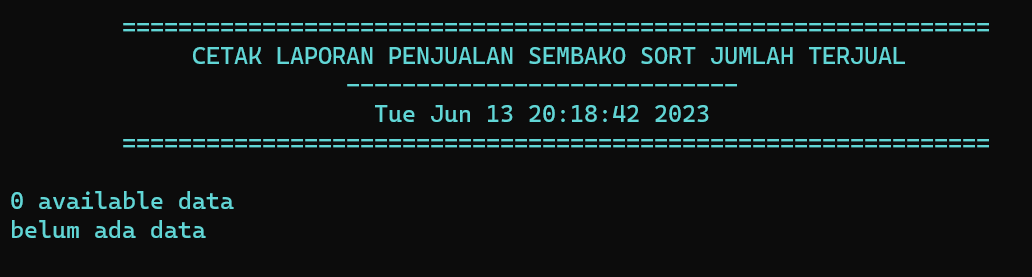
diatas adalah menu yang keluar jika memilih fungsi cetakLaporanTanggal atau menu ke-2. Fungsi cetakLaporanTanggal() digunakan untuk mencetak laporan penjualan Sembako berdasarkan tanggal transaksi.Data penjualan akan diurutkan berdasarkan tanggal transaksi secara descending sebelum dicetak.



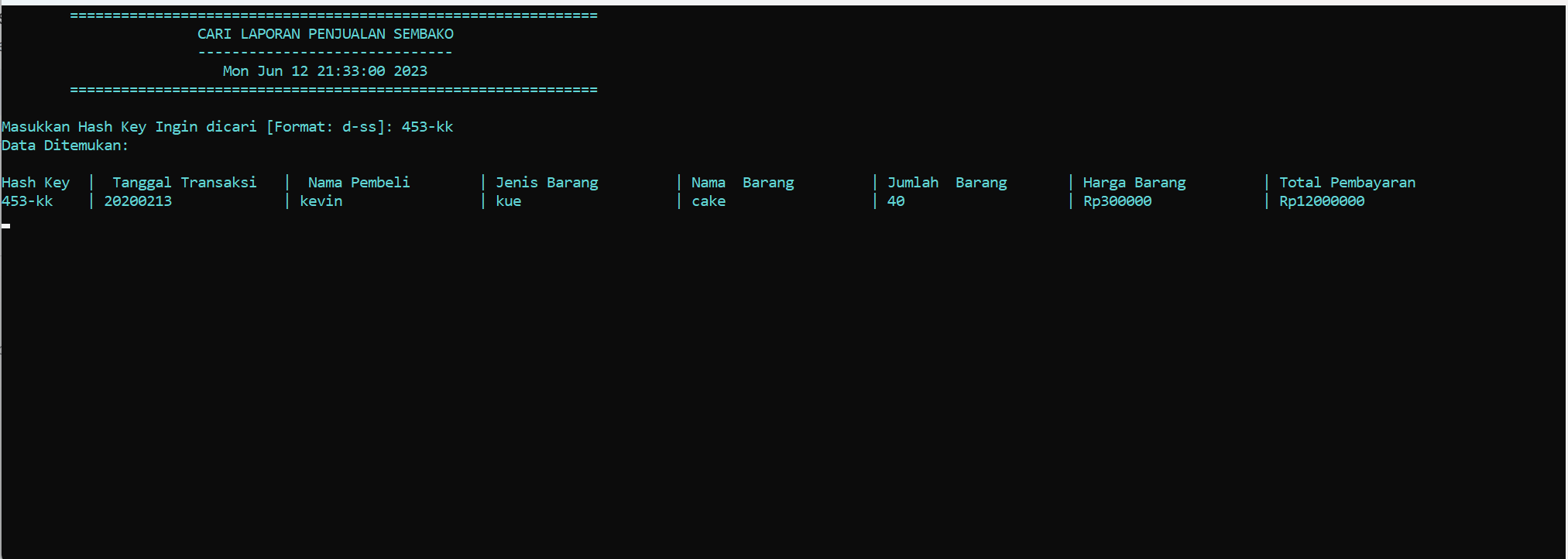
diatas adalah menu yang keluar jika memilih fungsi cetakLaporanJumlah atau menu ke-3. Fungsi cetakLaporanJumlah() digunakan untuk mencetak laporan penjualan Sembako berdasarkan jumlah terjual. Data penjualan akan diurutkan berdasarkan jumlah barang terjual secara descending sebelum dicetak.



gambar diatas adalah tampilan menu cetakLaporanTanggal, sebelum data dimasukan atau data kosong. Menampilkan “ belum ada data”.

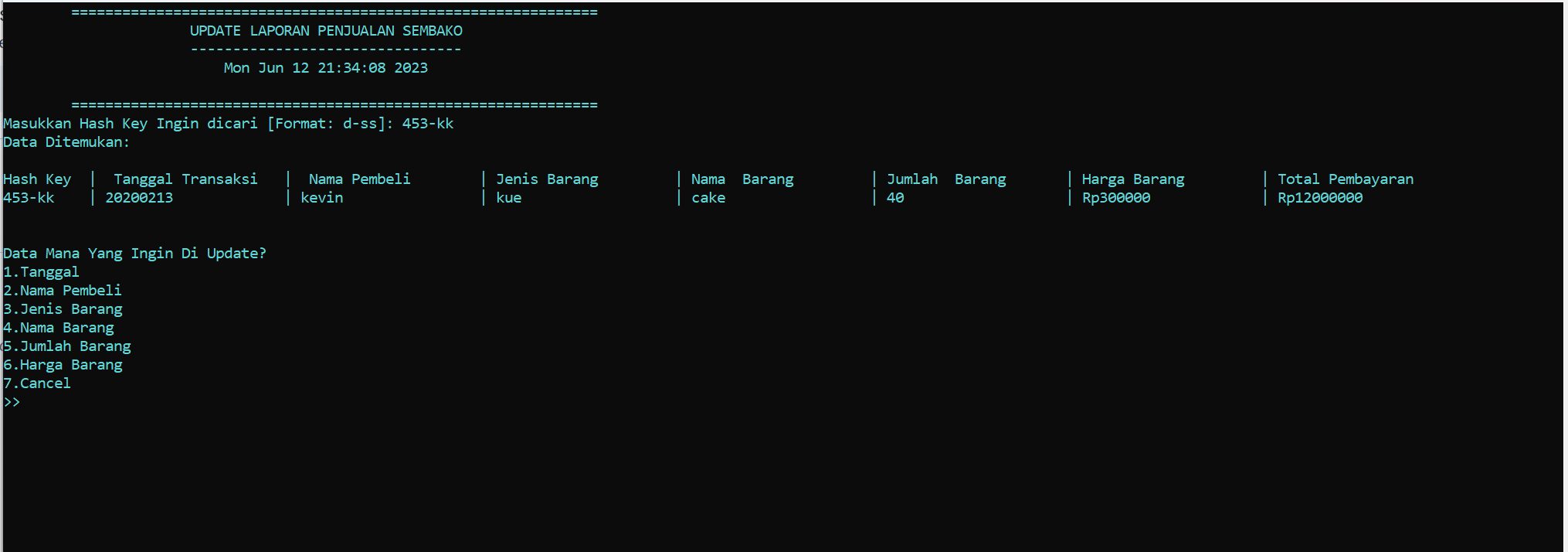


gambar diatas adalah tampilan menu cetakLaporanJumlah, sebelum data dimasukan atau data kosong. Menampilkan “ belum ada data”.



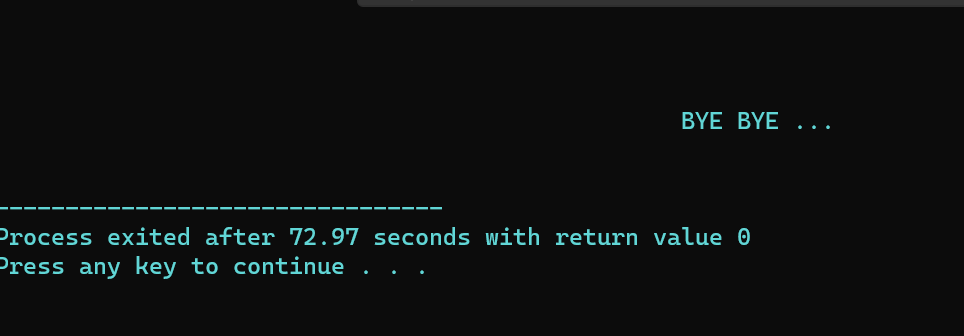
diatas adalah menu yang keluar jika memilih fungsi cariLaporan

Fungsi cariLaporan() digunakan untuk mencari laporan penjualan berdasarkan hash key yang diinputkan. Hash key tersebut digunakan untuk mencari data penjualan yang sesuai dengan hash key tersebut.



diatas adalah menu yang keluar jika memilih fungsi updateLaporan.

Fungsi updateLaporan() digunakan untuk memperbarui laporan penjualan. Fungsi ini akan menampilkan data penjualan yang tersimpan dalam program dan memungkinkan pengguna untuk memperbarui jumlah barang terjual.



gambar diatas adalah tampilan menu exit yang menampilkan kata “ BYE BYE..” dan program akan berhenti.

4.2 Kode Program

#include<stdio.h>

#include <time.h>

#include<string.h>

#include <windows.h>

#include <conio.h>

#define MAX\_HASH\_SIZE 997

struct sembako{

int hashKey;

char pembeli[255];

long long harga;

long long total\_harga;

char barang[255];

char jenis[255];

long int tanggal;

int jumlah\_barang;

bool available;

};

struct temp\_data\_info {

char pembeli[255];

long long harga;

long long total\_harga;

char barang[255];

char jenis[255];

long int tanggal;

int jumlah\_barang;

bool available;

};

temp\_data\_info temps[MAX\_HASH\_SIZE];

sembako data[MAX\_HASH\_SIZE];

int tempCount = 0;

int total\_data = 0;

FILE\* sales;

void dataPenjualan();

void cetakLaporan();

void updateLaporan();

void carilaporan();

void depan();

void menu();

void login();

void loading();

void jam();

void readData();

void updateMenuChoice (int res);

int divisionHash(const char\* str) {

int sum = 0;

int i = 0;

while (str[i] != '\0') {

sum += (int)str[i];

i++;

}

return sum % MAX\_HASH\_SIZE;

}

int linearProbing(int hashKey, int index) {

return (index + 1) % MAX\_HASH\_SIZE;

}

int divisionHashKey(const char\* str1, const char\* str2, long int tanggal) {

char uniqueStr[500];

sprintf(uniqueStr, "%ld\_%s\_%s", tanggal, str1, str2);

int asciiSum = divisionHash(uniqueStr);

int midDigits = (asciiSum % MAX\_HASH\_SIZE);

int index = midDigits;

int probeCount = 0;

while (data[index].available && probeCount < MAX\_HASH\_SIZE) {

if (data[index].hashKey == midDigits) {

index = linearProbing(midDigits, index);

probeCount++;

} else {

break;

}

}

return index;

}

int main(){

system ("color B");

loading();

login();

readData();

menu();

fclose(sales);

return 0;

}

void readData() {

FILE \*sales;

sales = fopen("sales.txt", "r");

while (fscanf(sales, "%ld-%[^-]-%[^-]-%[^-]-%d-%d-%d\r\n", &data[total\_data].tanggal, data[total\_data].pembeli, data[total\_data].jenis,

data[total\_data].barang, &data[total\_data].jumlah\_barang, &data[total\_data].harga, &data[total\_data].total\_harga) != EOF) {

total\_data++;

}

}

void gotoxy(int x,int y){

COORD coord = {0,0};

coord.X = x;

coord.Y = y;

SetConsoleCursorPosition(GetStdHandle(STD\_OUTPUT\_HANDLE), coord);

}

void depan(){

system ("cls");

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

printf("\t\t\tSELAMAT DATANG DI SembakoXpert!!!\t\t\t\n");

printf("\t\t\t----------------------------------\n");

jam();

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

printf("1. Pendataan penjualan bahan sembako\n");

printf("2. Pencetakan laporan penjualan bahan sembako (sort menurut tanggal transaksi)\n");

printf("3. Pencetakan laporan penjualan bahan sembako (sort menurut jumlah terjual)\n");

printf("4. Cari Laporan Penjualan\n");

printf("5. Update Laporan Penjualan\n");

printf("6. Exit\n");

printf(" \n");

}

void swap(sembako\* xp, sembako\* yp)

{

sembako temp = \*xp;

\*xp = \*yp;

\*yp = temp;

}

void cetakLaporanTanggal() {

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

printf("\t CETAK LAPORAN PENJUALAN SEMBAKO SORT TANGGAL TRANSAKSI \t\t\t\n");

printf("\t\t\t----------------------------\n");

jam();

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

printf("%d available data\n", total\_data);

if (!total\_data) {

printf("belum ada data \n");

return;

}

int i, j;

int min\_idx;

for (i = 0; i < total\_data - 1; i++) {

min\_idx = i;

for (j = i + 1; j < total\_data; j++) {

if (data[j].tanggal < data[min\_idx].tanggal) {

min\_idx = j;

}

}

if (min\_idx != i) {

swap(&data[min\_idx], &data[i]);

}

}

char table[][30] = {

"Hash Key ",

" Tanggal Transaksi",

" Nama Pembeli",

"Jenis Barang",

"Nama Barang",

"Jumlah Barang",

"Harga Barang",

"Total Pembayaran"

};

printf("%\*s | %\*s | %\*s | %\*s | %\*s | %\*s | %\*s | %\*s\n",-4, table[0], -20, table[1], -20, table[2], -20, table[3], -20, table[4], -20, table[5], -20, table[6], -20, table[7]);

for (int i = 0; i < total\_data; i++) {

sembako temp = data[i];

int hashKey = divisionHashKey(temp.pembeli, temp.jenis, temp.tanggal);

printf("%d-%c%-4c | %ld %-11c | %-20s | %-20s | %-20s | %-20d | Rp%-18lld | Rp%-20lld\n", hashKey, temp.pembeli[0], temp.jenis[0], temp.tanggal, ' ', temp.pembeli, temp.jenis, temp.barang, temp.jumlah\_barang, temp.harga, temp.total\_harga);

}

}

void cetakLaporanJumlah(){

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

printf("\t CETAK LAPORAN PENJUALAN SEMBAKO SORT JUMLAH TERJUAL \t\t\t\n");

printf("\t\t\t----------------------------\n");

jam();

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

printf("%d available data\n",total\_data);

if(!total\_data){

printf("belum ada data \n");

return;

}

int i, j, max\_idx;

for (i = 0; i < total\_data; i++) {

max\_idx = i;

for (j = i + 1; j < total\_data; j++)

if (data[j].jumlah\_barang > data[max\_idx].jumlah\_barang)

max\_idx = j;

swap(&data[max\_idx], &data[i]);

}

char table[][30] = {

"Hash Key ",

" Tanggal Transaksi",

" Nama Pembeli",

"Jenis Barang",

"Nama Barang",

"Jumlah Barang",

"Harga Barang",

"Total Pembayaran"

};

printf("%\*s | %\*s | %\*s | %\*s | %\*s | %\*s | %\*s | %\*s\n",-4, table[0], -20, table[1], -20, table[2], -20, table[3], -20, table[4], -20, table[5], -20, table[6], -20, table[7]);

for (int i = 0; i < total\_data; i++) {

sembako temp = data[i];

int hashKey = divisionHashKey(temp.pembeli, temp.jenis, temp.tanggal);

printf("%d-%c%-4c | %ld %-11c | %-20s | %-20s | %-20s | %-20d | Rp%-18lld | Rp%-20lld\n", hashKey, temp.pembeli[0], temp.jenis[0], temp.tanggal, ' ', temp.pembeli, temp.jenis, temp.barang, temp.jumlah\_barang, temp.harga, temp.total\_harga);

}

}

void loading(){

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

printf("\n\n\n\t\t\t\t\t Loading...");

printf("\n\n");

printf( "\t\t\t\t\t");

for (int i=0;i<=25;i++)

printf("-");

Sleep(20);

printf("\r");

printf("\t\t\t\t\t ");

for (int i=0; i<= 25; i++)

{

printf(">");

Sleep(40);

}

printf ("\n\n\t\t\t\t\t 'PRESS ANY BUTTON TO LOGIN'");

system ("pause>0");

system ("cls");

}

void login(){

char username[10], password[8];

int ulang = 0;

for(int j=0; j<3; j++){

printf("\t\t\t\t\t\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n");

printf("\n");

printf("\t\t\t\t ======== STAFF LOGIN =========\n");

printf("\t\t\t\t\t\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n");

printf("\n");

printf("\n");

printf("Username: ");

scanf ( "%s", &username);

printf("\n");

printf("Password: ");

fflush(stdin);

char ch;

int i=0;

int x=10;

int y=8;

while((ch = (char) \_getch()) != '\r'){

if (ch == 8){

password[i] = '\0';

gotoxy(x-1,y);

printf(" ");

x--;

i--;

gotoxy(x,y);

}

else{

password[i]=ch;

gotoxy(x,y);

printf("\*");

x++;

i++;

}

}

if ((strcmp(username,"admin")==0)&&(strcmp(password,"admin")==0)) {

printf("\n\n\n\n\n\n\t\t\t\t\t LOGIN SUKSES!!!!");

Sleep(1400);

break;

} else {

ulang++;

if(ulang != 3){

printf("\n\nPassword atau username yang anda masukan salah.");

Sleep(500);

system ("pause>0");

}

else if(ulang == 3){

printf("\n\nAnda Terlalu Banyak Mencoba,Silahkan Buka Progam Ini Kembali.");

Sleep(1400);

exit(0);

}

}

system("cls");

}

}

void jam(){

struct tm \*waktu;

time\_t Timeval;

Timeval = time(0);

waktu = localtime(&Timeval);

printf("\t\t\t %s", asctime(waktu));

}

void dataPenjualan() {

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

printf("\t\t\t PENDATAAN PENJUALAN SEMBAKO \t\t\t\n");

printf("\t\t\t----------------------------\n");

jam();

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

long int tanggal;

printf("Tanggal Transaksi (YYYYMMDD): ");

scanf("%ld", &tanggal);

data[total\_data].tanggal = tanggal;

printf("Nama Pembeli : ");

scanf(" %[^\n]", data[total\_data].pembeli);

printf("Jenis Barang : ");

scanf(" %[^\n]", data[total\_data].jenis);

printf("Nama Barang : ");

scanf(" %[^\n]", data[total\_data].barang);

printf("Jumlah Barang : ");

scanf("%d", &data[total\_data].jumlah\_barang);

printf("Harga Barang: Rp ");

scanf("%lld", &data[total\_data].harga);

char uniqueStr[500];

sprintf(uniqueStr, "%ld\_%s\_%s", tanggal, data[total\_data].pembeli, data[total\_data].jenis);

data[total\_data].hashKey = divisionHash(uniqueStr);

data[total\_data].total\_harga = data[total\_data].jumlah\_barang \* data[total\_data].harga;

printf("Total Pembayaran: Rp %lld\n", data[total\_data].total\_harga);

data[total\_data].available = true;

FILE \*output = fopen ("sales.txt","a+");

if(fprintf(output,"%ld-%s-%s-%s-%d-%lld-%lld\n", data[total\_data].tanggal, data[total\_data].pembeli, data[total\_data].jenis, data[total\_data].barang, data[total\_data].jumlah\_barang, data[total\_data].harga, data[total\_data].total\_harga) > 0) {

printf ("\n\n\t\t\t\t\t 'DATA BERHASIL DI INPUT'");

Sleep(1300);

}

fclose(output);

total\_data++;

}

void cariLaporan() {

int hashKey;

int res = -1;

int start = 0;

int end = total\_data - 1;

char c[200];

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

printf("\t\t CARI LAPORAN PENJUALAN SEMBAKO \t\t\t\n");

printf("\t\t ------------------------------\n");

jam();

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

printf("Masukkan Hash Key Ingin dicari [Format: d-ss]: ");

scanf("%d-%c%c", &hashKey, &c[0], &c[1]);

for (int i = 0; i < total\_data; i++) {

int tempHashKey = divisionHashKey(data[i].pembeli, data[i].jenis, data[i].tanggal);

if (tempHashKey == hashKey && data[i].pembeli[0] == c[0] && data[i].jenis[0] == c[1]) {

res = i;

break;

}

}

if (res == -1)

printf("Data tidak ditemukan di daftar\n");

else {

sembako temp = data[res];

printf("Data Ditemukan:\n\n");

char table[][30] = {

"Hash Key ",

" Tanggal Transaksi",

" Nama Pembeli",

"Jenis Barang",

"Nama Barang",

"Jumlah Barang",

"Harga Barang",

"Total Pembayaran"

};

printf("%\*s | %\*s | %\*s | %\*s | %\*s | %\*s | %\*s | %\*s\n",

-4, table[0], -20, table[1], -20, table[2], -20, table[3], -20, table[4], -20, table[5], -20, table[6], -20, table[7]);

int hashKey = divisionHashKey(temp.pembeli, temp.jenis, temp.tanggal);

printf("%d-%c%-4c | %ld %-11c | %-20s | %-20s | %-20s | %-20d | Rp%-18lld | Rp%-20lld\n",

hashKey, temp.pembeli[0], temp.jenis[0], temp.tanggal, ' ', temp.pembeli, temp.jenis, temp.barang,

temp.jumlah\_barang, temp.harga, temp.total\_harga);

system("pause>0");

}

}

void updateFile() {

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

if (output == NULL) {

printf("Gagal membuka file sales.txt\n");

return;

}

for (int i = 0; i < total\_data; i++) {

fprintf(output, "%ld-%s-%s-%s-%d-%lld-%lld\n", data[i].tanggal, data[i].pembeli, data[i].jenis, data[i].barang, data[i].jumlah\_barang, data[i].harga, data[i].total\_harga);

}

fclose(output);

}

void updateLaporan() {

int start = 0;

int end = total\_data - 1;

int hashKey;

char c[200];

int res = -1;

printf("\t==============================================================\t");

printf("\n\t\t UPDATE LAPORAN PENJUALAN SEMBAKO \t\t\t");

printf("\n\t\t --------------------------------\n");

jam();

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

printf("Masukkan Hash Key Ingin dicari [Format: d-ss]: ");

scanf("%d-%c%c", &hashKey, &c[0], &c[1]);

for (int i = 0; i < total\_data; i++) {

int tempHashKey = divisionHashKey(data[i].pembeli, data[i].jenis, data[i].tanggal);

if (tempHashKey == hashKey && data[i].pembeli[0] == c[0] && data[i].jenis[0] == c[1]) {

res = i;

break;

}

}

if (res == -1)

printf("\nData dengan Hash Key %d tidak ditemukan di daftar\n", hashKey);

else {

sembako temp = data[res];

printf("Data Ditemukan:\n\n");

char table[][30] = {

"Hash Key ",

" Tanggal Transaksi",

" Nama Pembeli",

"Jenis Barang",

"Nama Barang",

"Jumlah Barang",

"Harga Barang",

"Total Pembayaran"

};

printf("%\*s | %\*s | %\*s | %\*s | %\*s | %\*s | %\*s | %\*s\n",-4, table[0], -20, table[1], -20, table[2], -20, table[3], -20, table[4], -20, table[5], -20, table[6], -20, table[7]);

int hashKey = divisionHashKey(temp.pembeli, temp.jenis, temp.tanggal);

printf("%d-%c%-4c | %ld %-11c | %-20s | %-20s | %-20s | %-20d | Rp%-18lld | Rp%-20lld\n", hashKey, temp.pembeli[0], temp.jenis[0], temp.tanggal, ' ', temp.pembeli, temp.jenis, temp.barang, temp.jumlah\_barang, temp.harga, temp.total\_harga);

system ("pause>0");

updateMenuChoice(res);

updateFile();

printf("\n\n\t\t\t\t\t 'DATA BERHASIL DI UBAH'");

Sleep(1300);

return;

}

}

void showUpdateChoices() {

printf("\n\nData Mana Yang Ingin Di Update?\n");

printf("1.Tanggal\n2.Nama Pembeli\n3.Jenis Barang\n4.Nama Barang\n5.Jumlah Barang\n6.Harga Barang\n7.Cancel\n");

}

void updateMenuChoice(int i) {

bool choosingUpdate = true;

while (choosingUpdate) {

showUpdateChoices();

int updateChoice;

do {

printf(">> ");

scanf("%d", &updateChoice);

getchar();

} while (updateChoice < 1 || updateChoice > 7);

if (updateChoice == 1) {

scanf("%ld", &temps[tempCount].tanggal);

getchar();

data[i].tanggal = temps[tempCount].tanggal;

tempCount++;

return;

}

else if (updateChoice == 2) {

scanf("%[^\n]", temps[tempCount].pembeli);

getchar();

strcpy(data[i].pembeli, temps[tempCount].pembeli);

tempCount++;

return;

}

else if (updateChoice == 3) {

scanf("%[^\n]", temps[tempCount].jenis);

getchar();

strcpy(data[i].jenis, temps[tempCount].jenis);

tempCount++;

return;

}

else if (updateChoice == 4) {

scanf("%[^\n]", temps[tempCount].barang);

getchar();

strcpy(data[i].barang, temps[tempCount].barang);

tempCount++;

return;

}

else if (updateChoice == 5) {

scanf("%d", &temps[tempCount].jumlah\_barang);

getchar();

data[i].jumlah\_barang = temps[tempCount].jumlah\_barang;

data[i].total\_harga = data[i].jumlah\_barang \* data[i].harga;

tempCount++;

return;

}

else if (updateChoice == 6) {

scanf("%lld", &temps[tempCount].harga);

getchar();

data[i].harga = temps[tempCount].harga;

data[i].total\_harga = data[i].jumlah\_barang \* data[i].harga;

tempCount++;

return;

}

else if (updateChoice == 7) {

printf("Data tidak berubah\n");

return;

}

}

}

void menu(){

bool onApp = true;

while(onApp) {

depan();

int choice = -1;

do {

printf(">> ");

scanf("%d", &choice); getchar();

system("cls");

}

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

if(choice == 1) {

dataPenjualan();

}else if(choice ==2){

cetakLaporanTanggal();

system ("pause>0");

}else if(choice ==3){

cetakLaporanJumlah();

system ("pause>0");

}else if(choice ==4){

cariLaporan();

}else if(choice ==5){

updateLaporan();

}else if(choice == 6) {

printf("\n\n\n\t\t\t\t\t\t BYE BYE ...");

printf("\n\n");

printf( "\t\t\t\t\t");

onApp = false;

}

}

}

**BAB 5**

**Daftar Pustaka**

1. Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). Introduction to Algorithms (3rd ed.). MIT Press.
2. Aho, A. V., Hopcroft, J. E., & Ullman, J. D. (1983). Data Structures and Algorithms. Addison-Wesley.
3. Agrawal, Anand, Sriram Bhyravarapu, and Nuthalapati Venkata KrishnaChaitanya. ”Matrix hashing with two level of collision resolution.” 20188th International Conference on Cloud Computing, Data Science &Engineering (Conﬂuence). IEEE, 2018.
4. Chatterjee, S., & Dutta, S. (2016). Collision handling techniques in Hash Tables: A survey. International Journal of Computer Science and Information Technologies, 7(6), 2536-2541.

LEMBAR PENILAIAN

**<Implementasi Program Laporan Toko Sembako dengan Metode Hashing>**

MATA KULIAH COMP6362004 – DATA STRUCTURES

KELAS < BD20 - LAB >

Semester [Ganjil, Genap] <2022,2023>

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **DAFTAR MAHASISWA** | **NILAI** | | | | **BOBOT** | | | | **KREDIT** | | | | **TOTAL KREDIT** |
| **1** | **2** | **3** | **4** | **1** | **2** | **3** | 4 | 1 | 2 | 3 | 4 |
| <2602126896> - <KEVIN HUSODO> |  |  |  |  | 20% | 30% | 30% | 20% |  |  |  |  |  |
| <2602080636> - < KEVIN FARRELL FEBRIANUS TANAMATRAS > |  |  |  |  | 20% | 30% | 30% | 20% |  |  |  |  |  |
| **TOTAL** | | | | | | | | | | | | |  |

**KETERANGAN :**

** Skala Penilaian : 0 sd 100**

** Komponen**

1 : Laporan

2 : Produk

3 : Pengetahuan dan Solusi

4 : Presentasi

**Malang, …… - ……………. ……..**

**(<ELIZABETH PASKAHLIA GUNAWAN, S.Kom., M.Cs.>)**

**<D5707>**

**Pembagian Tugas Final Projek**

**Program :** KEVIN HUSODO dan KEVIN FARRELL FEBRIANUS TANAMATRAS

**Laporan :   
Flowchart :** KEVIN FARRELL FEBRIANUS TANAMATRAS

**Layout design :** KEVIN HUSODO

**Pseudocode :** KEVIN HUSODO

**Untuk sisanya bersama sama dikerjakan.**