Aplikasi Kasir Sederhana   
untuk Coffeeshop

LAPORAN STUDI KASUS TUGAS AOL  
MATA KULIAH COMP6360004 – ALGORITHM & PROGRAMMING  
KELAS LC20

Text

Description automatically generated

DISUSUN OLEH :  
2602172233 – RAYNALDY DWI KHARISMA

COMPUTER SCIENCE

SEMESTER GANJIL 2022/2023  
MALANG

Daftar Isi

[BAB I PENDAHULUAN 3](#_Toc125623374)

[1. LATAR BELAKANG 3](#_Toc125623375)

[2. KEBUTUHAN APLIKASI 3](#_Toc125623376)

[BAB II DESAIN PROGRAM 4](#_Toc125623377)

[BAB III SOURCE CODE 5](#_Toc125623378)

[BAB IV TAMPILAN HASIL 24](#_Toc125623379)

[1. Tampilan Awal 24](#_Toc125623380)

[2. Tampilan Input Order 24](#_Toc125623381)

[3. Tampilan Rekap Order 24](#_Toc125623382)

[4. Tampilan Checkout 25](#_Toc125623383)

[5. Tampilan Menu Admin 25](#_Toc125623384)

[6. Tampilan View Sales 25](#_Toc125623385)

[7. Tampilan Sort Sales 25](#_Toc125623386)

[8. Tampilan Sort Sales Berdasarkan Nama Produk 26](#_Toc125623387)

[9. Tampilan Sort Sales Berdasarkan Qty Produk Terjual 26](#_Toc125623388)

[10. Tampilan Sort Sales Berdasarkan Total Pembayaran 27](#_Toc125623389)

[11. Tampilan Search Sales 27](#_Toc125623390)

[12. Tampilan Search Berdasarkan Nama Produk 27](#_Toc125623391)

[13. Tampilan Search Berdasarkan Qty Penjualan 28](#_Toc125623392)

[14. Tampilan Search Berdasarkan Total Pembayaran 28](#_Toc125623393)

[15. Tampilan Search Berdasarkan Metode Pembayaran 28](#_Toc125623394)

# BAB I PENDAHULUAN

## 1. LATAR BELAKANG

Di era yang semakin modern ini, saya sering kali melihat *coffeeshop* atau warung kopi yang masih manual dalam mencatat, merekap, dan memproses order pelanggan. Namun, sudah banyak juga *coffeeshop* yang sudah menggunakan mesin kasir otomatis yaitu aplikasi *Point of Sales* (POS). Hal ini membuat saya tertarik untuk mereplika aplikasi tersebut dalam skala kecil tetapi masih memiliki fungsi yang sama, yaitu otomatisasi order pelanggan.

## 2. KEBUTUHAN APLIKASI

Adapun beberapa hal yang saya butuhkan untuk membuat aplikasi tersebut:

1. Struktur kendali perulangan
   1. Do while loop
   2. For loop
2. Seleksi
   1. If else
   2. Switch case
3. Tipe data
   1. Empat buah struct untuk mencatat menu, order, sales, waktu, dan tanggal order
   2. Pointer bertipe const char untuk menyimpan format data read file
   3. Int, char, float, void untuk menyimpan sesuatu dalam variable
   4. Pointer FILE untuk membaca dan menulis file
4. Media penyimpanan
   1. File menu.txt untuk menyimpan menu *coffeeshop*
   2. File sales.txt untuk menyimpan sales *coffeeshop*
5. Algoritma sorting
   1. qsort *built-in* yang terdapat di library <stdlib.h>
   2. comparer nama – untuk sorting berdasarkan nama produk
   3. comparer qty – untuk sorting berdasarkan kuantitas produk terjual
   4. comparer total – untuk sorting berdasarkan total pembayaran
6. Algoritma searching
   1. Linear search menggunakan for loops dan if else statement untuk mencari data yang sesuai dengan kriteria

# BAB II DESAIN PROGRAM

Diagram

Description automatically generated

# BAB III SOURCE CODE

#include <stdio.h>

#include <stdlib.h>

#include <ctype.h>

#include <time.h>

#include <string.h>

const char \*FORMAT\_DATA\_READ = "%d/%d/%d, %[^,], %c, %d, %d, %s\n";

struct Sales{

int date[4];

char nama[50];

char size;

int qty;

int total\_pembayaran;

char metode\_pembayaran[20];

} array\_sales[100], temp;

struct Menu{

char nama\_menu[50];

float harga\_regular;

float harga\_large;

} array\_menu[50];

struct Order{

char date[20];

int index\_order;

char ukuran\_order;

int qty\_order;

} array\_order[50];

void input\_order();

void load\_menu();

int index\_counter(char FILE\_DIR[]);

void print\_menu\_txt();

void scan\_order(int index\_temp);

void rekap\_order(int index\_temp);

void ubah\_order();

void print\_receipt(char sub\_input3, int index\_temp);

void write\_to\_file(char sub\_input3, int index\_temp);

void menu\_admin();

void sort\_sales();

void search\_sales();

void print\_sales();

int main(){

char input;

printf("Aplikasi Kasir\n");

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

printf("1. Input Order\n");

printf("2. Admin\n");

printf("0. Exit\n");

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

printf("Input: "); scanf("%d", &input); getchar();

switch(input){

case 1:

input\_order();

break;

case 2:

menu\_admin();

break;

case 0:

exit(0);

break;

default:

printf("Input salah!\n");

system("pause");

system("cls");

main();

}

return 0;

}

void load\_menu(){

char buffer[256];

FILE \*fp;

fp = fopen("menu.txt", "r");

if(fp == NULL){

printf("File menu.txt tidak ditemukan!\n");

}

fgets(buffer, sizeof(buffer), fp);

int index=0;

while(!feof(fp)){

fscanf(fp, "%[^,], %f, %f\n",

&array\_menu[index].nama\_menu,

&array\_menu[index].harga\_regular,

&array\_menu[index].harga\_large);

index++;

}

fclose(fp);

}

int index\_counter(char FILE\_DIR[]){

FILE \*fp;

int count = 0;

char c;

fp = fopen(FILE\_DIR, "r");

if (fp == NULL){

return 0;

}

for (c = getc(fp); c != EOF; c = getc(fp)){

if (c == '\n'){

count = count + 1;

}

}

fclose(fp);

return count;

}

void print\_menu\_txt(){

int index = index\_counter("menu.txt");

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

printf("| No | Nama | Regular | Large |\n");

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

for(int i=0; i<index; i++){

printf("| %-2d | %-22s | %.0f | %.0f |\n",

i+1,

array\_menu[i].nama\_menu,

array\_menu[i].harga\_regular,

array\_menu[i].harga\_large);

}

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

}

void scan\_order(int index\_temp){

printf("Pesan (No Produk): "); scanf("%d", &array\_order[index\_temp].index\_order); getchar();

printf("Ukuran (R/L): "); scanf("%c", &array\_order[index\_temp].ukuran\_order); getchar();

printf("Qty: "); scanf("%d", &array\_order[index\_temp].qty\_order); getchar();

}

void rekap\_order(int index\_temp){

float total=0;

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

printf("| No | Nama | Ukuran | Harga | Qty | Total |\n");

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

for(int i=0; i<index\_temp; i++){

if(array\_order[i].ukuran\_order == 'R'){

printf("| %-2d | %-22s | %-3c | %.0f | %-2d | %-6.0f |\n",

i+1,

array\_menu[array\_order[i].index\_order-1].nama\_menu,

array\_order[i].ukuran\_order,

array\_menu[array\_order[i].index\_order-1].harga\_regular,

array\_order[i].qty\_order,

(array\_menu[array\_order[i].index\_order-1].harga\_regular \* array\_order[i].qty\_order));

total+=(array\_menu[array\_order[i].index\_order-1].harga\_regular \* array\_order[i].qty\_order);

}

else if(array\_order[i].ukuran\_order == 'L'){

printf("| %-2d | %-22s | %-3c | %.0f | %-2d | %-6.0f |\n",

i+1,

array\_menu[array\_order[i].index\_order-1].nama\_menu,

array\_order[i].ukuran\_order,

array\_menu[array\_order[i].index\_order-1].harga\_large,

array\_order[i].qty\_order,

(array\_menu[array\_order[i].index\_order-1].harga\_large \* array\_order[i].qty\_order));

total+=(array\_menu[array\_order[i].index\_order-1].harga\_large \* array\_order[i].qty\_order);

}

}

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

printf("\t\t\t\t Total Pembayaran = %.0f\n", total);

}

void ubah\_order(){

int temp;

printf("Pilih nomor pesanan yang ingin diubah!\n");

printf("Input: "); scanf("%d", &temp); getchar();

print\_menu\_txt();

scan\_order(temp-1);

}

void print\_receipt(char sub\_input3, int index\_temp){

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

printf("|\t\t\tIP Cafe 2\t\t\t |\n");

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

time\_t t = time(NULL);

struct tm tm = \*localtime(&t);

printf("| %d/%02d/%02d\t\t", tm.tm\_mday, tm.tm\_mon + 1, tm.tm\_year + 1900);

if(sub\_input3 == '1'){

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

}

else if(sub\_input3 == '2'){

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

}

else if(sub\_input3 == '3'){

printf("\t\t\t E-Wallet |\n");

}

printf("| %02d:%02d:%02d\t\t\t\t\t\t |\n", tm.tm\_hour, tm.tm\_min, tm.tm\_sec);

rekap\_order(index\_temp);

}

void write\_to\_file(char sub\_input3, int index\_temp){

time\_t t = time(NULL);

struct tm tm = \*localtime(&t);

FILE \*fp;

fp = fopen("sales.txt", "a");

for(int i=0; i<index\_temp; i++){

if(toupper(array\_order[i].ukuran\_order) == 'R' && sub\_input3 == '1'){

fprintf(fp, "%d/%d/%d, %s, %c, %d, %.0f, Cash\n",

tm.tm\_mday, tm.tm\_mon + 1, tm.tm\_year + 1900,

array\_menu[array\_order[i].index\_order-1].nama\_menu,

array\_order[i].ukuran\_order,

array\_order[i].qty\_order,

(array\_menu[array\_order[i].index\_order-1].harga\_regular \* array\_order[i].qty\_order));

}

else if(toupper(array\_order[i].ukuran\_order) == 'R' && sub\_input3 =='2'){

fprintf(fp, "%d/%d/%d, %s, %c, %d, %.0f, QRIS\n",

tm.tm\_mday, tm.tm\_mon + 1, tm.tm\_year + 1900,

array\_menu[array\_order[i].index\_order-1].nama\_menu,

array\_order[i].ukuran\_order,

array\_order[i].qty\_order,

(array\_menu[array\_order[i].index\_order-1].harga\_regular \* array\_order[i].qty\_order));

}

else if(toupper(array\_order[i].ukuran\_order) == 'R' && sub\_input3 =='3'){

fprintf(fp, "%d/%d/%d, %s, %c, %d, %.0f, E-Wallet\n",

tm.tm\_mday, tm.tm\_mon + 1, tm.tm\_year + 1900,

array\_menu[array\_order[i].index\_order-1].nama\_menu,

array\_order[i].ukuran\_order,

array\_order[i].qty\_order,

(array\_menu[array\_order[i].index\_order-1].harga\_regular \* array\_order[i].qty\_order));

}

else if(toupper(array\_order[i].ukuran\_order) == 'L' && sub\_input3 == '1'){

fprintf(fp, "%d/%d/%d, %s, %c, %d, %.0f, Cash\n",

tm.tm\_mday, tm.tm\_mon + 1, tm.tm\_year + 1900,

array\_menu[array\_order[i].index\_order-1].nama\_menu,

array\_order[i].ukuran\_order,

array\_order[i].qty\_order,

(array\_menu[array\_order[i].index\_order-1].harga\_large \* array\_order[i].qty\_order));

}

else if(toupper(array\_order[i].ukuran\_order) == 'L' && sub\_input3 =='2'){

fprintf(fp, "%d/%d/%d, %s, %c, %d, %.0f, QRIS\n",

tm.tm\_mday, tm.tm\_mon + 1, tm.tm\_year + 1900,

array\_menu[array\_order[i].index\_order-1].nama\_menu,

array\_order[i].ukuran\_order,

array\_order[i].qty\_order,

(array\_menu[array\_order[i].index\_order-1].harga\_large \* array\_order[i].qty\_order));

}

else if(toupper(array\_order[i].ukuran\_order) == 'L' && sub\_input3 =='3'){

fprintf(fp, "%d/%d/%d, %s, %c, %d, %.0f, E-Wallet\n",

tm.tm\_mday, tm.tm\_mon + 1, tm.tm\_year + 1900,

array\_menu[array\_order[i].index\_order-1].nama\_menu,

array\_order[i].ukuran\_order,

array\_order[i].qty\_order,

(array\_menu[array\_order[i].index\_order-1].harga\_large \* array\_order[i].qty\_order));

}

}

fclose(fp);

}

void input\_order(){

char sub\_input;

char sub\_input2;

char sub\_input3;

int index\_temp=0;

do{

system("cls");

load\_menu();

print\_menu\_txt();

scan\_order(index\_temp);

index\_temp++;

printf("Pesan lagi? (Y/N)\n");

printf("Input: "); scanf("%c", &sub\_input); getchar();

}while(toupper(sub\_input)!='N');

do{

system("cls");

rekap\_order(index\_temp);

printf("\n");

printf("1. Ubah pesanan\n");

printf("2. Checkout\n");

printf("Input: "); scanf("%c", &sub\_input2); getchar();

if(sub\_input2 == '1'){

ubah\_order();

}

}while(sub\_input2!='2');

printf("Pilih metode pembayaran\n");

printf("1. Cash\n");

printf("2. QRIS\n");

printf("3. E-Wallet\n");

printf("Input: "); scanf("%c", &sub\_input3); getchar();

system("cls");

print\_receipt(sub\_input3, index\_temp);

write\_to\_file(sub\_input3, index\_temp);

system("pause");

system("cls");

main();

}

void print\_array\_sales(int i){

printf("| %-2d | %d/%d/%d | %-22s | %-3c | %-3d | %-6d | %-8s |\n",

i+1,

array\_sales[i].date[0],

array\_sales[i].date[1],

array\_sales[i].date[2],

array\_sales[i].nama,

array\_sales[i].size,

array\_sales[i].qty,

array\_sales[i].total\_pembayaran,

array\_sales[i].metode\_pembayaran);

}

void initialize\_data(){

FILE \*fp;

int file\_index = index\_counter("sales.txt");

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

for(int i=0; i<file\_index; i++){

fscanf(fp, FORMAT\_DATA\_READ,

&array\_sales[i].date[0],

&array\_sales[i].date[1],

&array\_sales[i].date[2],

&array\_sales[i].nama,

&array\_sales[i].size,

&array\_sales[i].qty,

&array\_sales[i].total\_pembayaran,

&array\_sales[i].metode\_pembayaran);

}

fclose(fp);

}

void view\_sales(){

int file\_index = index\_counter("sales.txt");

initialize\_data();

system("cls");

print\_sales();

}

void print\_sales(){

int file\_index = index\_counter("sales.txt");

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

printf("| No | Tanggal | Nama | Size | Qty | Total | Metode |\n");

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

for(int i=0; i<file\_index; i++){

print\_array\_sales(i);

}

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

}

typedef int (\*compfn)(const void\*, const void\*);

int compare\_nama(struct Sales \*elem1, struct Sales \*elem2){

if(strcasecmp(elem1->nama, elem2->nama) < 0){

return -1;

}

else if(strcasecmp(elem1->nama, elem2->nama) > 0){

return 1;

}

return 0;

}

int compare\_qty(struct Sales \*elem1, struct Sales \*elem2){

if(elem1->qty < elem2->qty){

return -1;

}

else if(elem1->qty > elem2->qty){

return 1;

}

return 0;

}

int compare\_total(struct Sales \*elem1, struct Sales \*elem2){

if(elem1->total\_pembayaran < elem2->total\_pembayaran){

return -1;

}

else if(elem1->total\_pembayaran > elem2->total\_pembayaran){

return 1;

}

return 0;

}

void sort\_sales(){

int file\_index = index\_counter("sales.txt");

int sub\_input;

initialize\_data();

system("cls");

printf("Sort berdasarkan:\n");

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

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

printf("2. Qty\n");

printf("3. Total\n");

printf("0. Back\n");

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

printf("Input: "); scanf("%d", &sub\_input); getchar();

switch(sub\_input){

case 1:

qsort((void \*) &array\_sales, file\_index, sizeof(struct Sales), (compfn)compare\_nama);

system("cls");

printf("Sort Berdasarkan Nama Produk\n");

print\_sales();

system("pause");

sort\_sales();

break;

case 2:

qsort((void \*) &array\_sales, file\_index, sizeof(struct Sales), (compfn)compare\_qty);

system("cls");

printf("Sort Berdasarkan Qty Produk Terjual\n");

print\_sales();

system("pause");

sort\_sales();

break;

case 3:

qsort((void \*) &array\_sales, file\_index, sizeof(struct Sales), (compfn)compare\_total);

system("cls");

printf("Sort Berdasarkan Nama Produk\n");

print\_sales();

system("pause");

sort\_sales();

break;

case 0:

menu\_admin();

break;

default:

printf("Input salah!\n");

system("pause");

sort\_sales();

break;

}

}

void search\_by\_name(){

char key[25];

int found;

int file\_index=index\_counter("sales.txt");

system("cls");

printf("Masukkan nama produk yang akan dicari : ");

scanf("%s", key);

getchar();

found=0;

for(int i=0; i<file\_index; i++){

if(strstr(array\_sales[i].nama, key) != NULL){

found = 1;

}

}

if(found==0){

printf("Data tidak ditemukan\n");

system("pause");

search\_sales();

}

else{

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

printf("| No | Tanggal | Nama | Size | Qty | Total | Metode |\n");

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

for(int i=0; i<file\_index; i++){

if(strstr(array\_sales[i].nama, key) != NULL){

print\_array\_sales(i);

}

}

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

}

system("pause");

search\_sales();

}

void search\_by\_qty(){

char key[25];

int found, high, low;

int file\_index=index\_counter("sales.txt");

system("cls");

printf("Masukkan qty terendah : ");

scanf("%d", &low);

printf("Masukkan qty tertinggi : ");

scanf("%d", &high);

for(int i = 0; i < file\_index; i++){

if(array\_sales[i].qty >= low && array\_sales[i].qty <= high){

found=1;

};

}

if (found == 0){

printf("Data tidak ditemukan\n");

system("pause");

search\_sales();

}

else{

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

printf("| No | Tanggal | Nama | Size | Qty | Total | Metode |\n");

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

for(int i = 0; i < file\_index; i++){

if(array\_sales[i].qty >= low && array\_sales[i].qty <= high){

print\_array\_sales(i);

}

}

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

}

}

void search\_by\_sales(){

char key[25];

int found, high, low;

int file\_index=index\_counter("sales.txt");

system("cls");

printf("Masukkan pembayaran terendah : ");

scanf("%d", &low);

printf("Masukkan pembayaran tertinggi : ");

scanf("%d", &high);

for(int i = 0; i < file\_index; i++){

if(array\_sales[i].total\_pembayaran >= low && array\_sales[i].total\_pembayaran <= high){

found=1;

};

}

if (found == 0){

printf("Data tidak ditemukan\n");

system("pause");

search\_sales();

}

else{

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

printf("| No | Tanggal | Nama | Size | Qty | Total | Metode |\n");

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

for(int i = 0; i < file\_index; i++){

if(array\_sales[i].total\_pembayaran >= low && array\_sales[i].total\_pembayaran <= high){

print\_array\_sales(i);

}

}

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

}

}

void search\_by\_method(){

char key[25];

int found;

int file\_index=index\_counter("sales.txt");

system("cls");

printf("Masukkan metode pembayaran yang akan dicari : ");

scanf("%s", key);

getchar();

found=0;

for(int i=0; i<file\_index; i++){

if(strstr(array\_sales[i].metode\_pembayaran, key) != NULL){

found = 1;

}

}

if(found==0){

printf("Data tidak ditemukan\n");

system("pause");

search\_sales();

}

else{

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

printf("| No | Tanggal | Nama | Size | Qty | Total | Metode |\n");

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

for(int i=0; i<file\_index; i++){

if(strstr(array\_sales[i].metode\_pembayaran, key) != NULL){

print\_array\_sales(i);

}

}

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

}

system("pause");

search\_sales();

}

void search\_sales(){

int file\_index = index\_counter("sales.txt");

int sub\_input;

char key[25];

int found;

int high, low;

initialize\_data();

system("cls");

printf("Search berdasarkan:\n");

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

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

printf("2. Qty\n");

printf("3. Total Pembayaran\n");

printf("4. Metode Pembayaran\n");

printf("0. Back\n");

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

printf("Input: "); scanf("%d", &sub\_input); getchar();

switch(sub\_input){

case 1:

search\_by\_name();

break;

case 2:

search\_by\_qty();

break;

case 3:

search\_by\_sales();

break;

case 4:

search\_by\_method();

break;

case 0:

menu\_admin();

break;

default:

printf("Input salah!\n");

system("pause");

sort\_sales();

break;

}

}

void menu\_admin(){

char sub\_input;

system("cls");

printf("Menu Admin\n");

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

printf("1. View Sales\n");

printf("2. Sort Sales\n");

printf("3. Search Sales\n");

printf("0. Back\n");

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

printf("Input: "); scanf("%d", &sub\_input); getchar();

switch(sub\_input){

case 1:

view\_sales();

system("pause");

menu\_admin();

break;

case 2:

sort\_sales();

break;

case 3:

search\_sales();

break;

case 0:

system("cls");

main();

break;

default:

printf("Input salah!\n");

system("pause");

menu\_admin();

break;

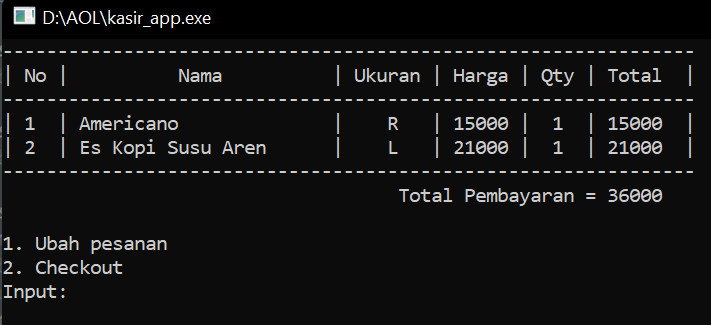
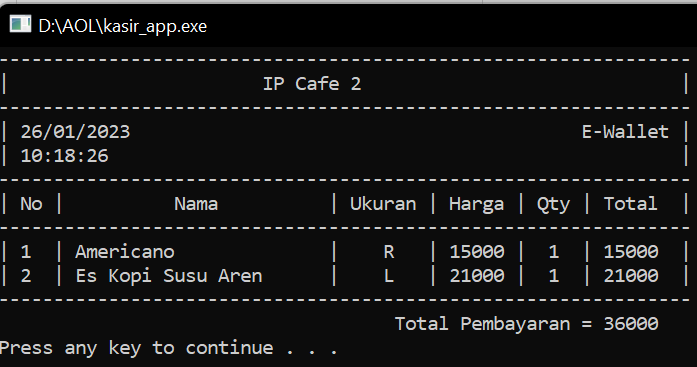
}

}

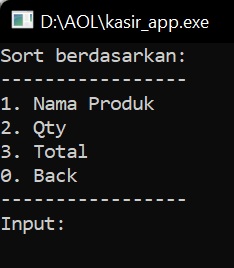
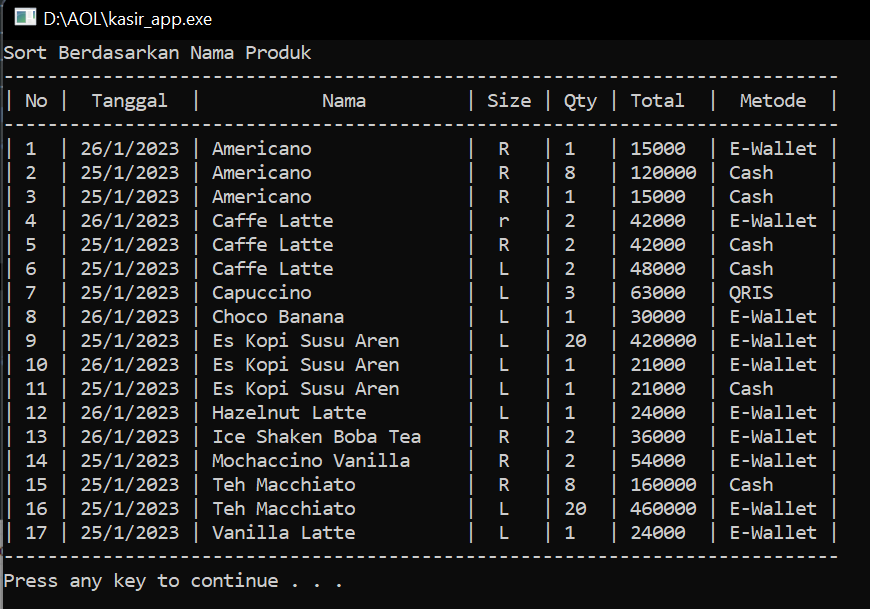
# BAB IV TAMPILAN HASIL

1. Tampilan Awal  
   Text

   Description automatically generated
2. Tampilan Input Order  
   A screenshot of a computer

   Description automatically generated with low confidence
3. Tampilan Rekap Order  
   
4. Tampilan Checkout  
   
5. Tampilan Menu Admin  
   Text

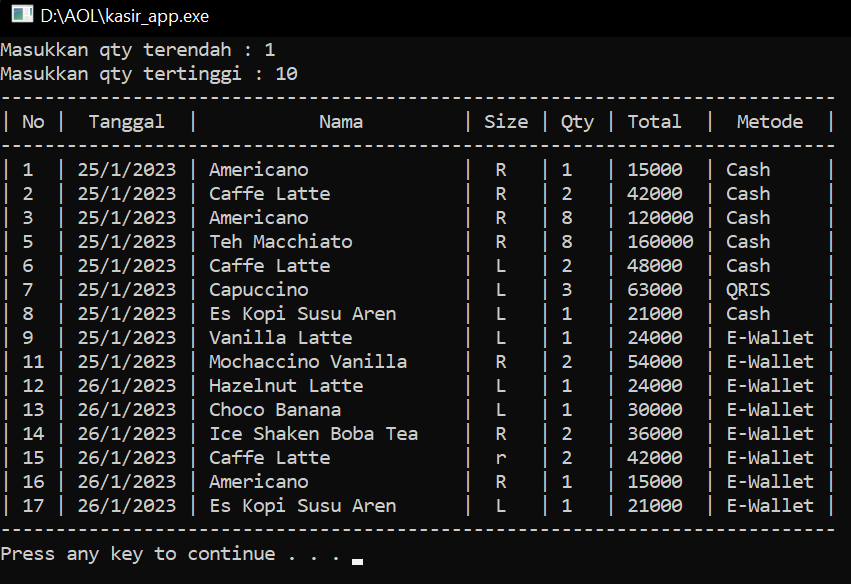
   Description automatically generated
6. Tampilan View Sales  
   Text

   Description automatically generated
7. Tampilan Sort Sales  
   
8. Tampilan Sort Sales Berdasarkan Nama Produk  
   
9. Tampilan Sort Sales Berdasarkan Qty Produk Terjual  
   Text

   Description automatically generated
10. Tampilan Sort Sales Berdasarkan Total Pembayaran  
    A screenshot of a computer

    Description automatically generated with medium confidence
11. Tampilan Search Sales  
    Text

    Description automatically generated
12. Tampilan Search Berdasarkan Nama Produk  
    Text

    Description automatically generated
13. Tampilan Search Berdasarkan Qty Penjualan  
    
14. Tampilan Search Berdasarkan Total Pembayaran  
    Text

    Description automatically generated
15. Tampilan Search Berdasarkan Metode Pembayaran  
    Text

    Description automatically generated