FINAL PROJECT
STRUKTUR DATA A081

Dosen Pengampu:

Fawwaz Ali Akbar, S.Kom, M.Kom

Disusun Oleh:

Kelompok 6:

1. Dafauzan Bilal Syaifulloh (21081010069)
2. Achmad Rozy Priambodo (21081010070)
3. Bima Ahmad Dionfaka (21081010075)
4. Mahardika Virgo Wuryantoro (21081010077)
5. Hamida Wefi Niamaputri (21081010080)

INFORMATIKA

FAKULTAS ILMU KOMPUTER

UPN "VETERAN" JAWA TIMUR

2022

A. Studi Kasus

Dengan adanya internet, di masa sekarang seseorang tidak perlu membutuhkan sebuah toko fisik untuk mulai berbisnis. Banyaknya media sosial dan platform penjualan online memungkinkan munculnya berbagai macam toko online. Di toko online, pembeli tidak perlu datang secara langsung ke toko, melainkan hanya perlu memilih produk yang diinginkan melalui gawai masing-masing, dan melakukan order atau pemesanan kepada penjual. Produk kemudian akan dikirimkan penjual ke alamat pembeli.

Dari sisi penjual juga terdapat banyak kemudahan. Penjual tidak perlu memiliki toko fisik dan memajang barang-barang, melainkan hanya perlu mengiklankan produk di toko online. Setelah itu, penjual hanya perlu menunggu pemesanan dari pelanggan dan nantinya mengirimkan pesanan tersebut. Namun, dengan tidak adanya toko fisik juga sangat dimungkinkan penyimpanan barang menjadi tidak terorganisir. Oleh karena itu, diperlukan sistem yang dapat mencatat semua barang-barang yang dijual oleh toko online. Selain itu, diperlukan juga sistem untuk menangani pemesanan dari pembeli, sebab tentu saja ada batasan mengenai seberapa banyak pesanan yang dapat ditangani oleh pembeli dalam satu hari.

B. Rancangan Program

Dari studi kasus di atas, dibuat sebuah program untuk menangani penyimpanan barang yang dijual serta pemesanan barang. Sistem penyimpanan barang akan ditangani dengan menggunakan struktur data linked list. Sistem pemesanan barang akan ditangani dengan menggunakan struktur data queue.

Program akan terdiri dari dua sisi, yakni sisi penjual dan sisi pembeli. Sisi penjual akan memuat menu seperti melihat daftar produk, menambah produk ke daftar, menghapus produk dari daftar, melihat daftar antrian, serta memroses antrian terdepan. Sisi pembeli akan memuat menu melihat daftar produk dan memesan produk (menambahkan antrian).

Antrian pemesanan akan dibatasi dengan jumlah yang sudah ditentukan sebelumnya. Apabila panjang antrian sudah mencapai batas, maka otomatis pembeli tidak bisa menambah pemesanan baru.

Program akan memiliki tiga buah struct. Struktur data linked list akan menggunakan sebuah struct sebagai masing-masing node yang ada di dalamnya. Struktur data queue akan menggunakan dua buah struct. Sebuah struct sebagai masing-masing node yang terdapat dalam queue dan sebuah struct sebagai antrian itu sendiri.

C. Algoritma

1. Fungsi main
    1) Mulai
    2) Deklarasi struct ListNode dengan member:
        - nama dengan tipe data string
        - stok dengan tipe data integer

- pointer next dengan tipe data ListNode

3) Deklarasi struct QueueNode dengan member:
   - namaBarang dengan tipe data string
   - namaPembeli dengan tipe data string
   - pointer next dengan tipe data QueueNode
4) Deklarasi struct Queue dengan member:
   - length dengan tipe data integer
   - pointer front dengan tipe data QueueNode
   - pointer rear dengan tipe data QueueNode
5) Deklarasi variabel pointer pHead dengan tipe data struct ListNode dan inisialisasi dengan NULL
6) Deklarasi variabel pQueue dengan tipe data struct Queue
   a) Inisialisasi member length dengan 0
   b) Inisialisasi member front dan rear dengan NULL
7) Looping: selama pilihan != ‘q’ lakukan:
   a) Tampilkan menu penjual atau pembeli
   b) Minta masukkan untuk variabel pilihan
   c) Periksa:
      - Jika pilihan == ‘1’, looping: selama option != ‘q’ lakukan:
        1. Tampilkan menu untuk penjual
        2. Minta masukkan untuk variabel option
        3. Panggil fungsi yang sesuai dengan option berdasarkan menu yang ada
      - Jika pilihan == ‘2’, looping: selama option != ‘q’ lakukan:
        1. Tampilkan menu untuk pembeli
        2. Minta masukkan untuk variabel option
        3. Panggil fungsi yang sesuai dengan option berdasarkan menu yang ada
8) Selesai

2. Fungsi addBarang
   1) Mulai, menerima parameter pointer to pointer pHead
   2) Deklarasi variabel nama dan masukkanSetelah dengan tipe data string
   3) Deklarasi variabel stok dengan tipe data integer
   4) Deklarasi variabel pointer pNew dan pointer pCur dengan tipe data struct ListNode
   5) Minta masukkan untuk variabel namaBarang
   6) Minta masukkan untuk variabel stok
   7) Alokasikan memori ke pNew
   8) Jika pNew == NULL, tampilkan pesan error bahwa memori tidak cukup dan kembalikan fungsi
   9) Simpan nilai dari variabel nama ke member nama dari pNew
   10) Simpan nilai dari variabel stok ke member stok dari pNew
   11) Inisialisasi member next dari pNew dengan NULL
   12) Dereference pHead dan simpan nilainya ke pCur
   13) Jika pCur == NULL:
      - Simpan nilai dari pNew ke dereference dari pHead

- Kembalikan fungsi

14) Panggil fungsi traverseList dan masukkan dereference dari pHead sebagai parameter
15) Minta masukkan untuk variabel masukkanSetelah
16) Looping: selama pCur != NULL dan member nama dari pCur != masukkan setelah lakukan:
    - Simpan nilai member next dari pCur ke variabel pCur
17) Jika pCur == NULL tampilkan pesan error bahwa barang tidak ditemukan dan kembalikan fungsi
18) Jika member next dari pCur == NULL, maka masukkan NULL ke member next dari pNew
19) Jika kondisi pada poin (18) salah, maka masukkan member next dari pCur ke member next dari pNew
20) Masukkan pNew ke member next dari pCur
21) Selesai

3. Fungsi hapusBarang
    1) Mulai, menerima variabel pointer to pointer pHead dengan tipe data struct ListNode
    2) Deklarasi variabel pointer pPrev dan variabel pointer pCur dengan tipe data struct ListNode
    3) Deklarasi variabel namaBarang dengan tipe data string
    4) Inisialisasi pCur dengan dereference dari pHead
    5) Inisialisasi pPrev dengan NULL
    6) Jika pCur == NULL tampilkan pesan error bahwa daftar produk kosong dan kembalikan fungsi
    7) Panggil fungsi traverseList, masukkan dereference dari pHead sebagai parameter
    8) Minta masukkan untuk variabel namaBarang
    9) Looping: selama pCur != NULL dan member nama dari pCur != namaBarang lakukan:
        a) Masukkan nilai dari pCur ke pPrev
        b) Masukkan nilai member next dari pCur ke pCur
    10) Jika pCur == NULL tampilkan pesan error bahwa nama barang tidak ditemukan dan kembalikan fungsi
    11) Jika pCur == dereference dari pHead, maka:
        a) Masukkan nilai member next dari pCur ke dereference pHead
        b) Bebaskan memori pCur
        c) Kembalikan fungsi
    12) Jika member next dari pCur == NULL, maka:
        a) Masukkan NULL ke member next dari pPrev
        b) Bebaskan memori pCur
        c) Kembalikan fungsi
    13) Masukkan member next dari pCur ke member next dari pPrev
    14) Bebaskan memori pCur
    15) Kembalikan fungsi
    16) Selesai

4. Fungsi enQueue
    1) Mulai, menerima parameter variabel pointer pQueue dengan tipe data struct Queue dan variabel pointer pHead dengan tipe data struct ListNode
    2) Deklarasi variabel MAX dengan tipe data integer dan inisialisasi dengan 5
    3) Deklarasi variabel pointer pNew dengan tipe data struct QueueNode
    4) Deklarasi variabel pointer pCur dengan tipe data struct ListNode
    5) Deklarasi variabel namaBarang dan namaPembeli dengan tipe data string
    6) Periksa, jika member length dari pQueue >= MAX, maka tampilkan pesan error bahwa antrian telah penuh dan kembalikan fungsi
    7) Panggil fungsi traverseList, masukkan pHead sebagai parameter
    8) Minta masukkan untuk variabel namaBarang
    9) Minta masukkan untuk variabel namaPembeli
    10) Inisialisasi pCur dengan nilai dari pHead
    11) Looping, selama pCur != NULL dan member nama dari pCur != namaBarang lakukan:
        - Masukkan nilai member next dari pCur ke pCur
    12) Jika pCur == NULL, maka tampilkan pesan error bahwa produk tidak tersedia dan kembalikan fungsi
    13) Alokasikan memori ke pNew
    14) Jika pNew == NULL, tampilkan pesan error bahwa memori tidak cukup dan kembalikan fungsi
    15) Masukkan nilai dari namaBarang ke member namaBarang dari pNew
    16) Masukkan nilai dari namaPembeli ke member namaPembeli dari pNew
    17) Jika member front dari pQueue == NULL, maka:
        a) Masukkan pNew ke member front dari pQueue
        b) Masukkan pNew ke member rear dari pQueue
        c) Inisialisasi member next dari pNew dengan NULL
        d) Tambah member length dari pQueue sebanyak 1
        e) Kembalikan fungsi
    18) Masukkan pNew ke member next dari member rear dari pQueue
    19) Masukkan pNew ke member rear dari pQueue
    20) Inisialisasi member next dari pNew dengan NULL
    21) Tambah member length dari pQueue sebanyak 1
    22) Selesai
5. Fungsi deQueue
    1) Mulai, menerima parameter variabel pointer pQueue dengan tipe data struct Queue
    2) Deklarasi variabel pointer pCur dengan tipe data struct QueueNode
    3) Inisialisasi pCur dengan member front dari pQueue
    4) Jika pCur == NULL tampilkan pesan error bahwa antrian kosong dan kembalikan fungsi
    5) Jika member next dari pCur == NULL, lakukan:
        a) Masukkan NULL ke member front dari pQueue
        b) Masukkan NULL ke member rear dari pQueue
    6) Jika poin (5) bernilai salah, maka masukkan nilai member next dari pCur ke member front dari pQueue

7) Bebaskan memori pada pCur
8) Kurangi member length dari pQueue sebanyak 1
9) Selesai

6. Fungsi traverseList
   1) Mulai, menerima parameter pointer pHead dengan tipe data struct ListNode
   2) Deklarasi variabel pointer pCur dengan tipe data struct ListNode dan inisialisasi dengan nilai pada pHead
   3) Jika pCur != NULL, looping selama pCur != NULL, lakukan:
      a) Tampilkan member nama dan stok dari pCur
      b) Masukkan nilai member next dari pCur ke variabel pCur
   4) Selesai
7. Fungsi traverseQueue
   1) Mulai, menerima parameter variabel pQueue dengan tipe data struct Queue
   2) Deklarasi variabel pointer pCur dengan tipe data struct QueueNode dan inisialisasi dengan nilai member front dari pQueue
   3) Jika pCur != NULL:
      a. Looping selama pCur != NULL, lakukan:
         a) Tampilkan member namaBarang dan namaPembeli dari pCur
         b) Masukkan nilai dari member next dari pCur ke variabel pCur
      b. Tampilkan member namaBarang dan namaPembeli dari pCur
   4) Tampilkan panjang antrian
   5) Selesai

D. Tautan GitHub

mdikavw/fp-strudat (github.com)

E. Kode Program

```
#include <stdio.h>

#include <stdlib.h>

#include <string.h>


struct ListNode {

    char nama[30];

    int stok;

    ListNode * next;

};

typedef struct listNode listNode;

void addBarang(ListNode **Head);


struct QueueNode {

    char namaBarang[30];
```

```
    char namaPembeli[30];
    QueueNode * next;
};
typedef struct queueNode queueNode;

struct Queue {
    int length;
    QueueNode * front;
    QueueNode * rear;
};
typedef struct Queue Queue;

void traverseList(ListNode * pHead){
    ListNode * pCur;
    pCur = pHead;
    if(pCur != NULL){
        while(pCur != NULL){
            printf("%s(%i) -> ", pCur->nama, pCur->stok);
            pCur = pCur->next;
        }
    }
    printf("NULL\n");
    return;
}

void addBarang(ListNode ** pHead){
    char nama[30], masukkanSetelah[30];
    int stok;
    ListNode * pNew, * pCur;
    printf("Masukkan nama barang: ");
    scanf(" %s", nama);
    printf("Masukkan stok barang: ");
    scanf("%i", &stok);
```

```
    pNew = (ListNode *)malloc(sizeof(ListNode));
    if(pNew == NULL){
        printf("Memori tidak cukup\n");
        system("pause");
        return;
    }
    strcpy(pNew->nama, nama);
    pNew->stok = stok;
    pNew->next = NULL;

    pCur = *pHead;

    if(pCur == NULL){
        *pHead = pNew;
        return;
    }
    traverseList(*pHead);
    printf("Masukkan barang setelah (ketik 'depan' jika ingin memasukkan
barang di depan): ");
    scanf(" %s", masukkanSetelah);

    if(strcmp(masukkanSetelah, "depan") == 0){
        pNew->next = *pHead;
        *pHead = pNew;
        return;
    }

    while(pCur != NULL && strcmp(pCur->nama, masukkanSetelah) != 0){
        pCur = pCur->next;
    }

    if(pCur == NULL){
        printf("Barang tidak ditemukan\n");
        system("pause");
```

```
        return;
    }

    if(pCur->next == NULL){
        pNew->next = NULL;
    } else {
        pNew->next = pCur->next;
    }
    pCur->next = pNew;
}

void hapusBarang(ListNode ** pHead){
    ListNode * pPrev, * pCur;
    char namaBarang[30];

    pCur = *pHead;
    pPrev = NULL;
    if(pCur == NULL){
        printf("Daftar produk kosong\n");
        system("pause");
        return;
    }
    traverseList(*pHead);
    printf("Masukkan nama barang yang akan dihapus: ");
    scanf(" %s", namaBarang);

    while(pCur != NULL && strcmp(pCur->nama, namaBarang) != 0){
        pPrev = pCur;
        pCur = pCur->next;
    }
    if(pCur == NULL){
        printf("Nama barang tidak ditemukan\n");
        system("pause");
        return;
```

```
    }
    if(pCur == *pHead){
        *pHead = pCur->next;
        free(pCur);
        return;
    }
    if(pCur->next == NULL){
        pPrev->next = NULL;
        free(pCur);
        return;
    }
    pPrev->next = pCur->next;
    free(pCur);
    return;
}


void enQueue(Queue * pQueue, ListNode * pHead){
    int MAX = 5;
    QueueNode * pNew;
    ListNode * pCur;
    char namaBarang[30], namaPembeli[30];
    if(pQueue->length >= MAX){
        printf("Antrian telah melebihi batas\n");
        system("pause");
        return;
    }

    printf("Masukkan nama barang: ");
    scanf(" %s", namaBarang);
    printf("Masukkan nama pembeli: ");
    scanf(" %s", namaPembeli);

    pCur = pHead;
```

```
    while(pCur != NULL && strcmp(strlwr(pCur->nama), strlwr(namaBarang))
!= 0){
        pCur = pCur->next;
    }
    if(pCur == NULL){
        printf("Produk tidak tersedia\n");
        system("pause");
        return;
    }

    pNew = (QueueNode *)malloc(sizeof(QueueNode));
    if(pNew == NULL){
        printf("Memori tidak cukup\n");
        system("pause");
        return;
    }
    strcpy(pNew->namaBarang, namaBarang);
    strcpy(pNew->namaPembeli, namaPembeli);
    if(pQueue->front == NULL){
        pQueue->front = pNew;
        pQueue->rear = pNew;
        pNew->next = NULL;
        pQueue->length++;
        system("pause");
        return;
    }
    pQueue->rear->next = pNew;
    pQueue->rear = pNew;
    pNew->next = NULL;
    pQueue->length++;
}

void deQueue(Queue * pQueue){
    QueueNode * pCur;
```

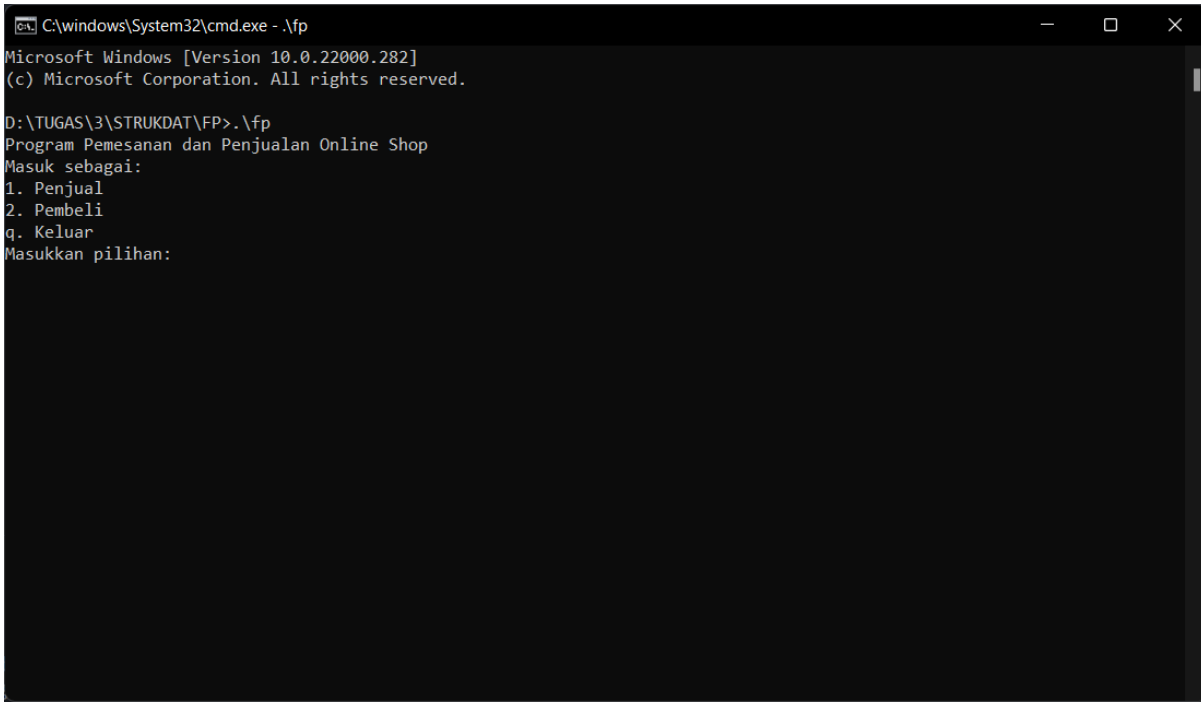
```
    pCur = pQueue->front;
    if(pCur == NULL){
        printf("Antrian kosong\n");
        system("pause");
        return;
    }
    if(pCur->next == NULL){
        pQueue->front = NULL;
        pQueue->rear = NULL;
        system("pause");
    } else {
        pQueue->front = pCur->next;
    }
    free(pCur);
    pQueue->length--;
    system("pause");
    return;
}

void traverseQueue(Queue pQueue){
    QueueNode * pCur;
    pCur = pQueue.front;
    if(pCur != NULL){
        while(pCur != pQueue.rear){
            printf("%s (%s) -> ", pCur->namaPembeli, pCur->namaBarang);
            pCur = pCur->next;
        }
        printf("%s (%s) -> ", pCur->namaPembeli, pCur->namaBarang);
    }
    printf("NULL\n");
    printf("Panjang antrian: %i\n", pQueue.length);
}

int main(){
```

```
char pilihan, option;

ListNode * pHead;
pHead = NULL;

Queue pQueue;
pQueue.length = 0;
pQueue.front = NULL;
pQueue.rear = NULL;

printf("Program Pemesanan dan Penjualan Online Shop\n");
do{
    printf("Masuk sebagai: \n");
    printf("1. Penjual\n");
    printf("2. Pembeli\n");
    printf("q. Keluar\n");
    printf("Masukkan pilihan: ");
    scanf("%c", &pilihan);
    system("cls");
    if(pilihan == '1'){
        do{
            printf("Pilihan Menu: \n");
            printf("1. Lihat daftar produk\n");
            printf("2. Tambah produk baru\n");
            printf("3. Hapus produk\n\n");
            printf("4. Lihat antrian\n");
            printf("5. Proses antrian terdepan\n");
            printf("q. Keluar\n");
            printf("Masukkan pilihan: ");
            scanf(" %c", &option);
            if(option == '1'){
                traverseList(pHead);
                system("pause");
            } else if(option == '2'){
```

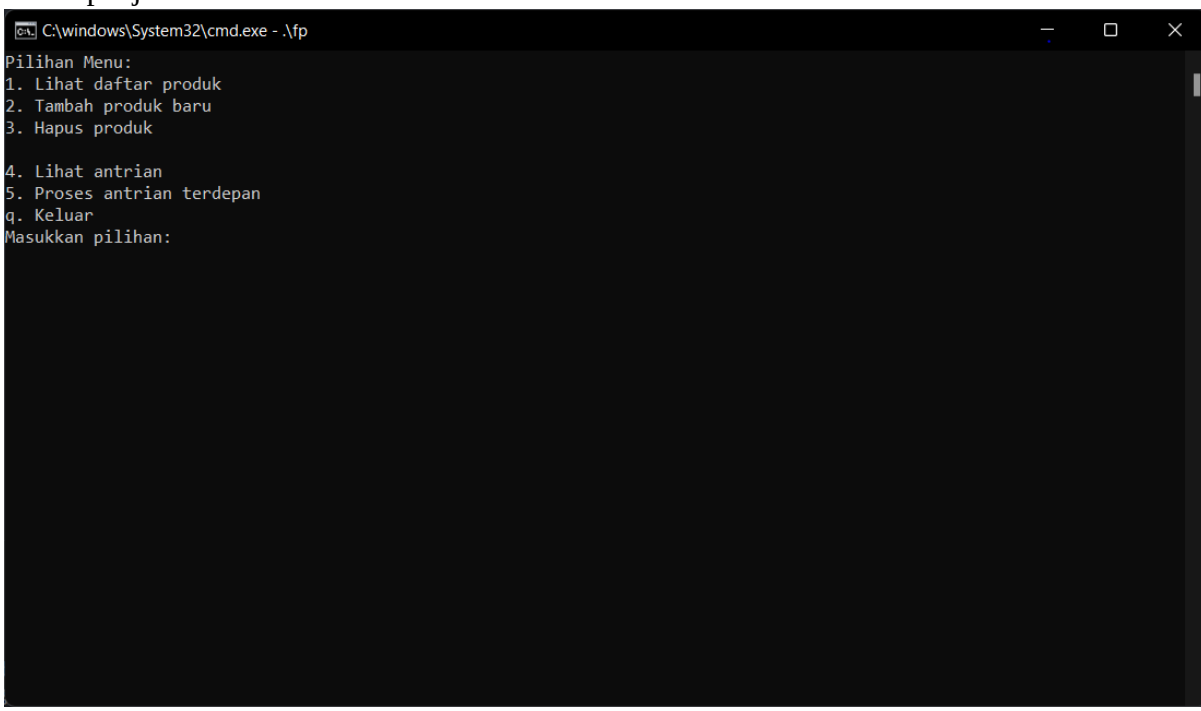
```
                    addBarang(&pHead);
                } else if(option == '3'){
                   hapusBarang(&pHead);
                } else if(option == '4'){
                    traverseQueue(pQueue);
                    system("pause");
                } else if(option == '5'){
                    deQueue(&pQueue);
                }
                system("cls");
            } while (option != 'q');
        } else if(pilihan == '2'){
            do{
                printf("Pilihan Menu: \n");
                printf("1. Lihat daftar produk\n");
                printf("2. Pesan produk\n");
                printf("q. Keluar\n");
                printf("Masukkan pilihan: ");
                scanf(" %c", &option);
                if(option == '1'){
                    traverseList(pHead);
                } else if(option == '2'){
                    enQueue(&pQueue, pHead);
                }
                system("cls");
            } while (option != 'q');
        }
    } while (pilihan != 'q');

    return 0;
}
```

F. Screenshot Program

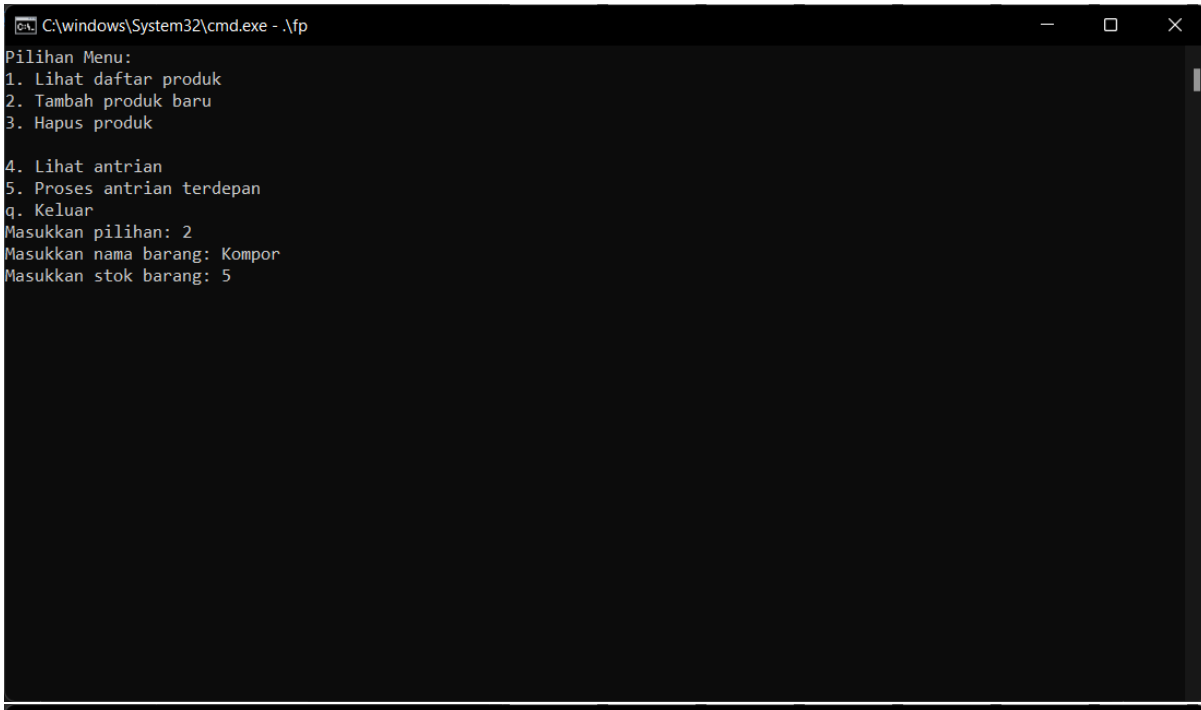
1. Tampilan awal

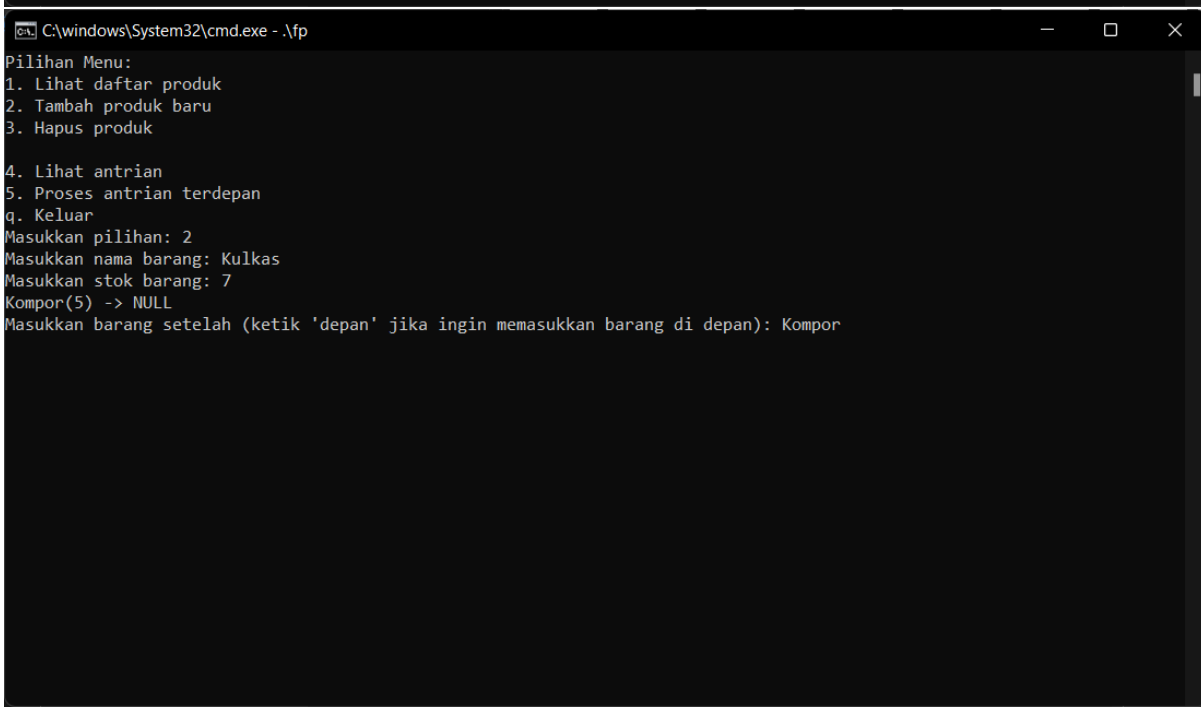


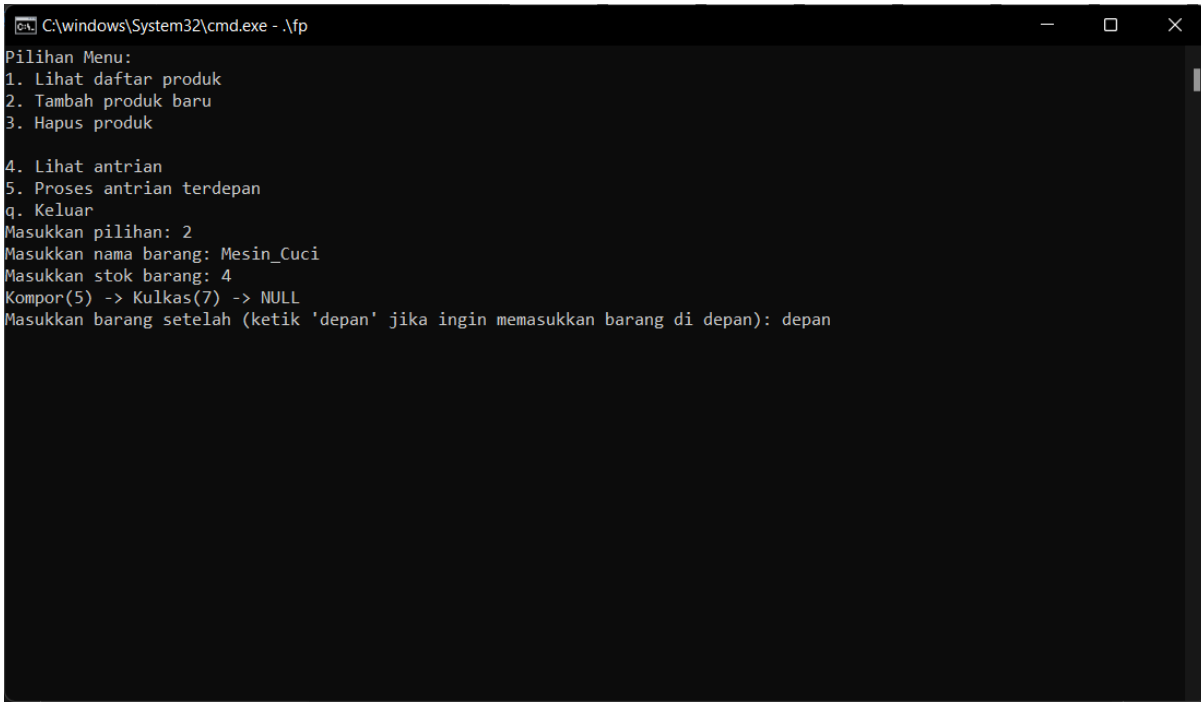
2. Menu penjual



3. Menambahkan produk


C:\windows\System32\cmd.exe - .\fp
Pilihan Menu:
1. Lihat daftar produk
2. Tambah produk baru
3. Hapus produk
4. Lihat antrian
5. Proses antrian terdepan
q. Keluar
Masukkan pilihan: 2
Masukkan nama barang: Kompor
Masukkan stok barang: 5


C:\windows\System32\cmd.exe - .\fp
Pilihan Menu:
1. Lihat daftar produk
2. Tambah produk baru
3. Hapus produk
4. Lihat antrian
5. Proses antrian terdepan
q. Keluar
Masukkan pilihan: 2
Masukkan nama barang: Kulkas
Masukkan stok barang: 7
Kompor(5) -> NULL
Masukkan barang setelah (ketik 'depan' jika ingin memasukkan barang di depan): Kompor

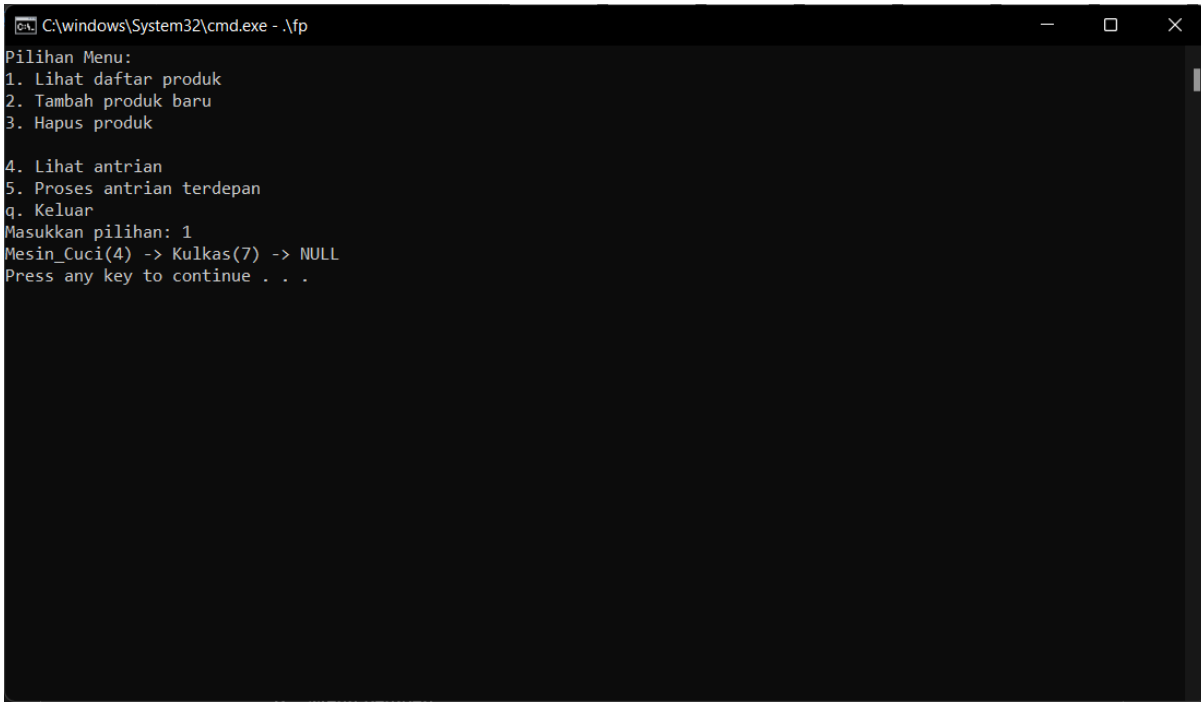


4. Melihat daftar produk

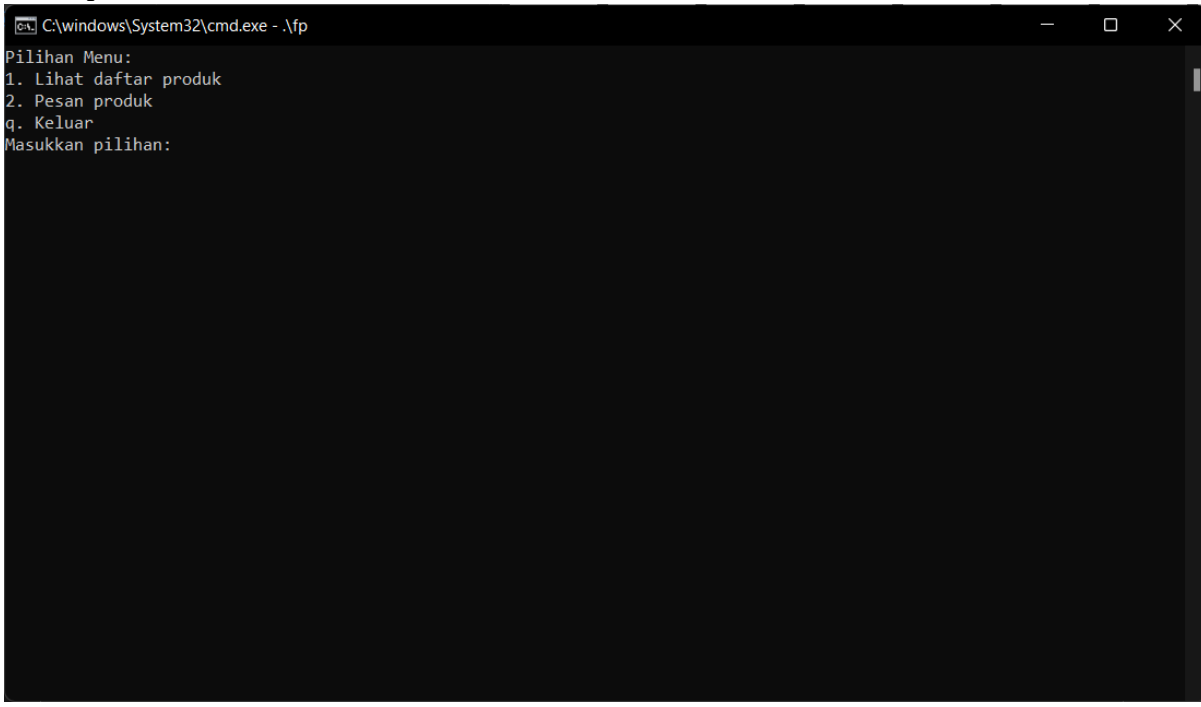
5. Menghapus produk


C:\windows\System32\cmd.exe - .\fp
Pilihan Menu:
1. Lihat daftar produk
2. Tambah produk baru
3. Hapus produk
4. Lihat antrian
5. Proses antrian terdepan
q. Keluar
Masukkan pilihan: 3
Mesin_Cuci(4) -> Kompor(5) -> Kulkas(7) -> NULL
Masukkan nama barang yang akan dihapus: Treadmill
Nama barang tidak ditemukan
Press any key to continue . . .

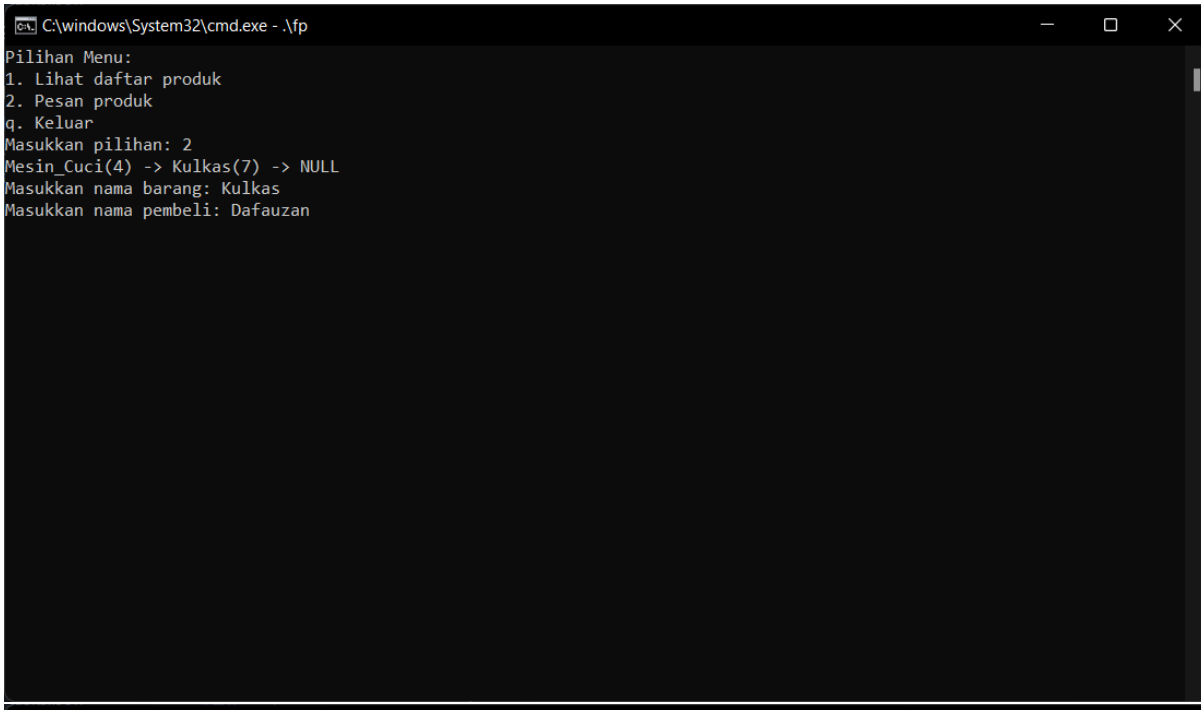

C:\windows\System32\cmd.exe - .\fp
Pilihan Menu:
1. Lihat daftar produk
2. Tambah produk baru
3. Hapus produk
4. Lihat antrian
5. Proses antrian terdepan
q. Keluar
Masukkan pilihan: 3
Mesin_Cuci(4) -> Kompor(5) -> Kulkas(7) -> NULL
Masukkan nama barang yang akan dihapus: Kompor

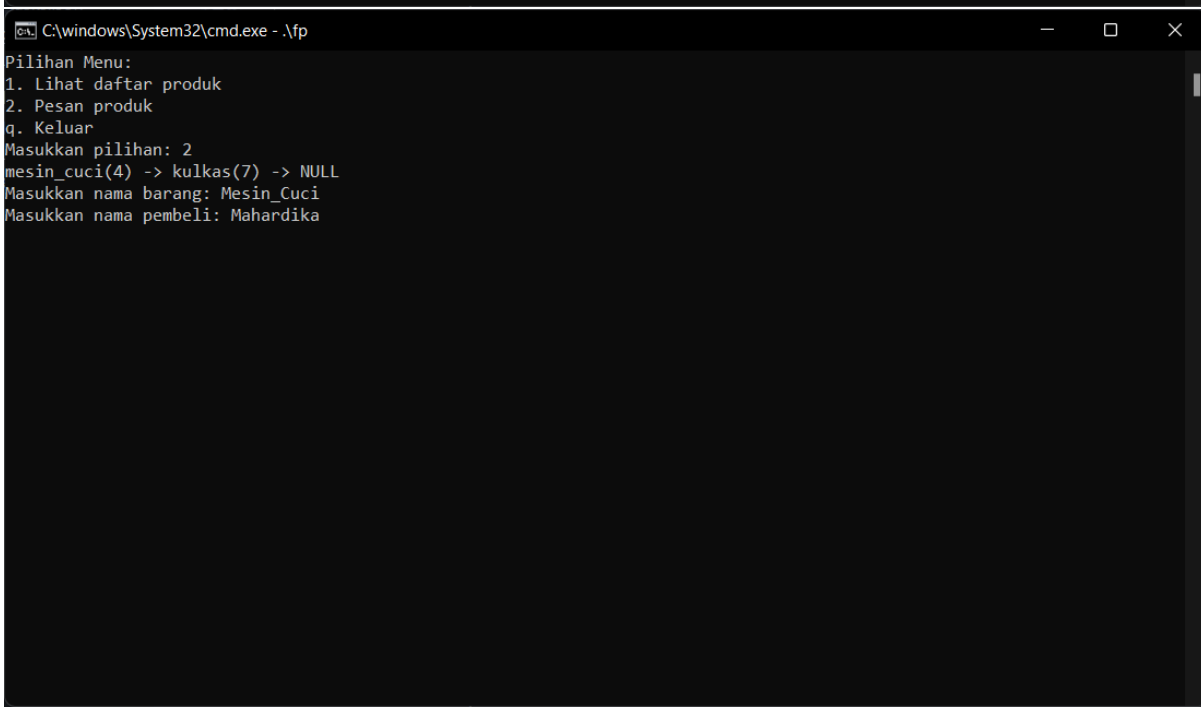


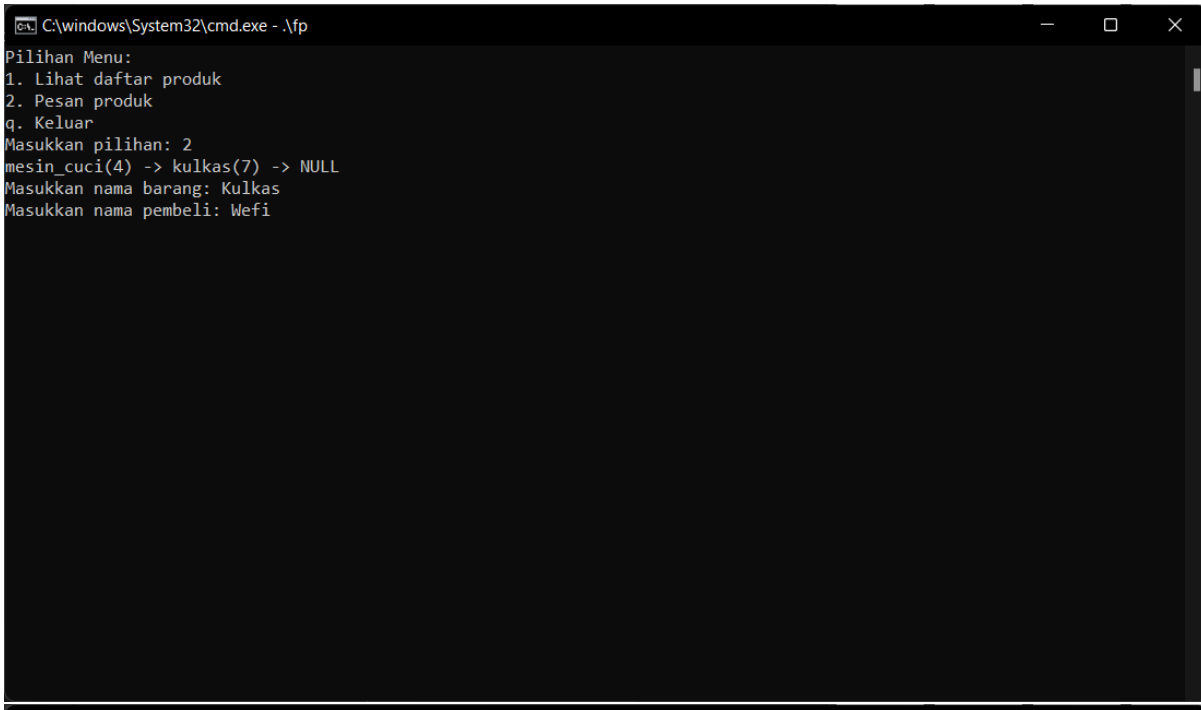
6. Menu pembeli

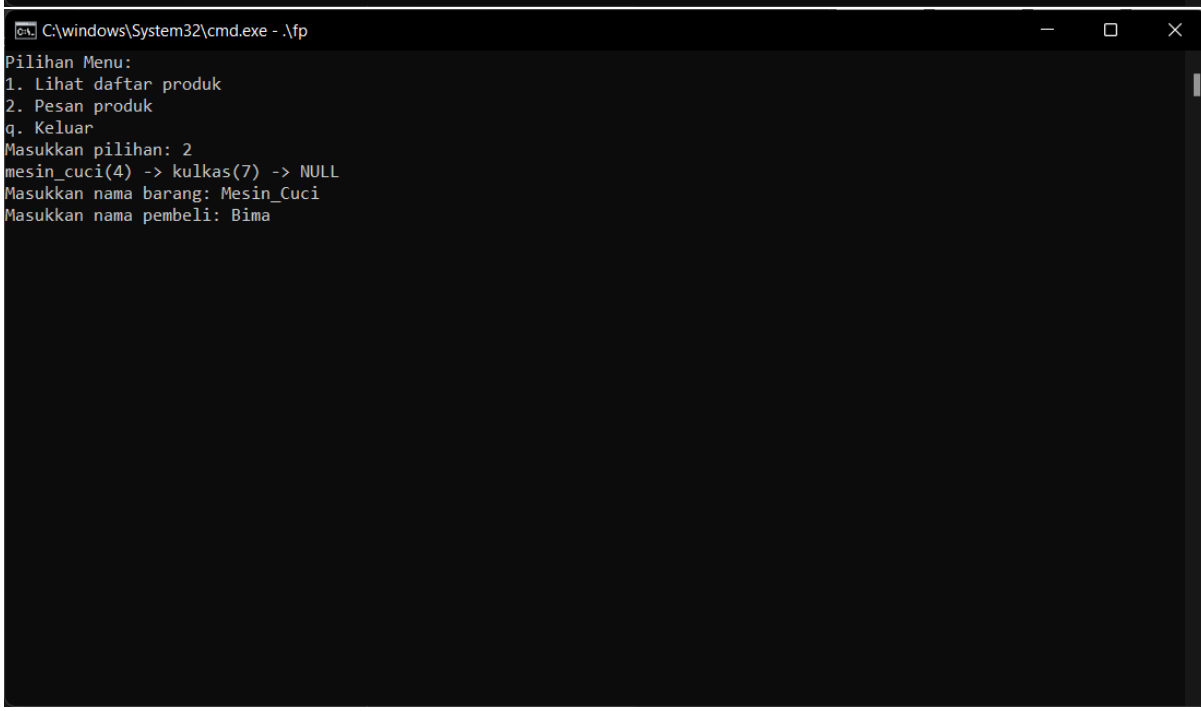


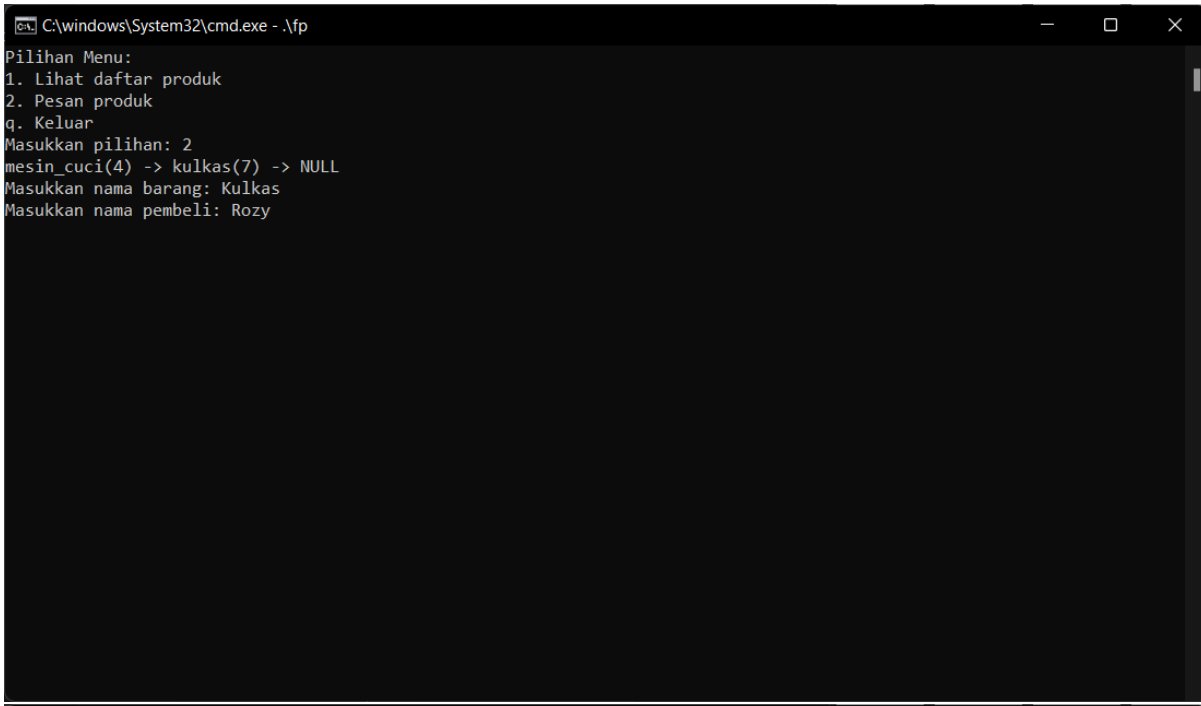
7. Memesan produk (enqueue)

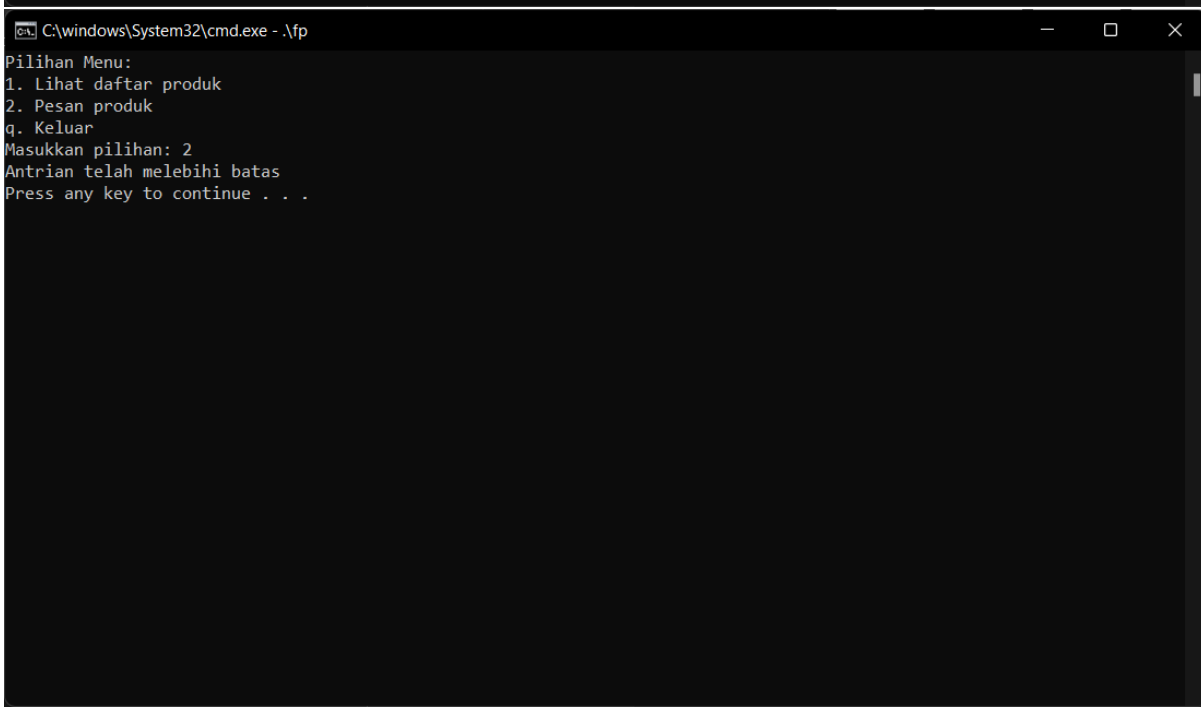

C:\windows\System32\cmd.exe - .\fp
Pilihan Menu:
1. Lihat daftar produk
2. Pesan produk
q. Keluar
Masukkan pilihan: 2
Mesin_Cuci(4) -> Kulkas(7) -> NULL
Masukkan nama barang: Kulkas
Masukkan nama pembeli: Dafauzan


C:\windows\System32\cmd.exe - .\fp
Pilihan Menu:
1. Lihat daftar produk
2. Pesan produk
q. Keluar
Masukkan pilihan: 2
mesin_cuci(4) -> kulkas(7) -> NULL
Masukkan nama barang: Mesin_Cuci
Masukkan nama pembeli: Mahardika

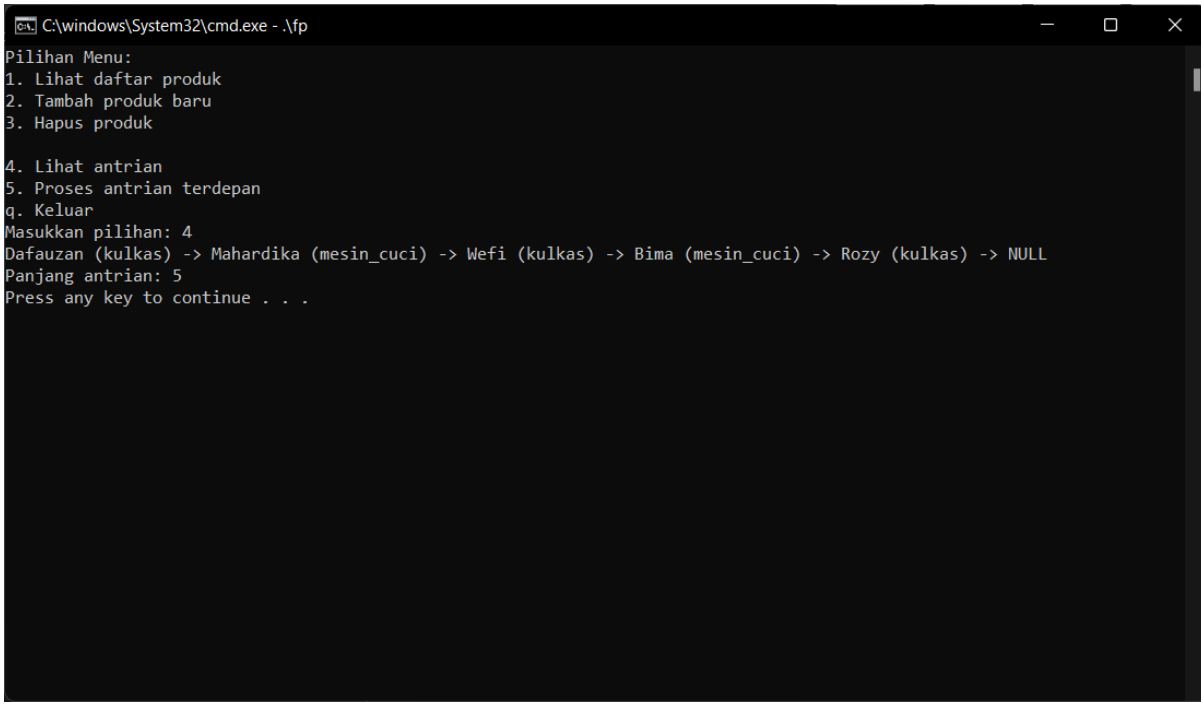

C:\windows\System32\cmd.exe - .\fp
Pilihan Menu:
1. Lihat daftar produk
2. Pesan produk
q. Keluar
Masukkan pilihan: 2
mesin_cuci(4) -> kulkas(7) -> NULL
Masukkan nama barang: Kulkas
Masukkan nama pembeli: Wefi


C:\windows\System32\cmd.exe - .\fp
Pilihan Menu:
1. Lihat daftar produk
2. Pesan produk
q. Keluar
Masukkan pilihan: 2
mesin_cuci(4) -> kulkas(7) -> NULL
Masukkan nama barang: Mesin_Cuci
Masukkan nama pembeli: Bima

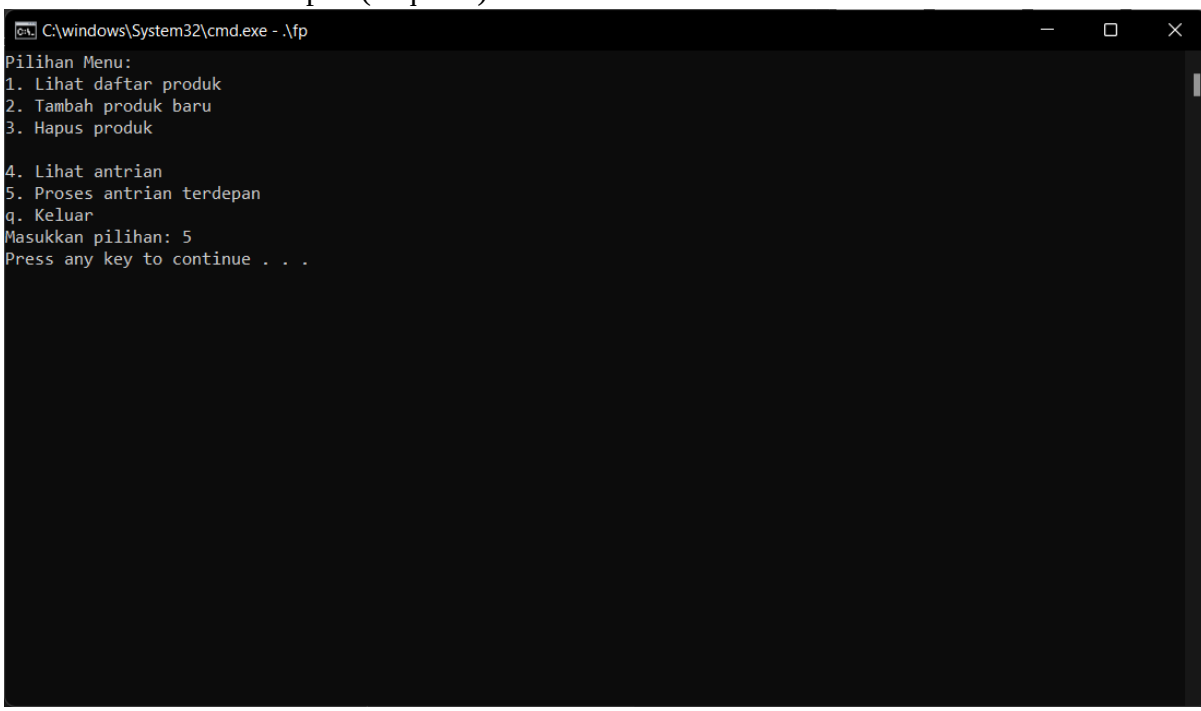




8. Melihat antrian



9. Memroses antrian terdepan (dequeue)




C:\windows\System32\cmd.exe - .\fp
Pilihan Menu:
1. Lihat daftar produk
2. Tambah produk baru
3. Hapus produk
4. Lihat antrian
5. Proses antrian terdepan
q. Keluar
Masukkan pilihan: 4
Mahardika (mesin_cuci) -> Wefi (kulkas) -> Bima (mesin_cuci) -> Rozy (kulkas) -> NULL
Panjang antrian: 4
Press any key to continue . . .