**PRAKTIKUM**

**STRUKTUR DATA C081**

**“SISTEM OPERASIONAL WAHANA KERETA HANTU”**



*ditujukan untuk memenuhi salah satu tugas mata kuliah Struktur Data C081 yang diampu oleh Fawwaz Ali Akbar, S.Kom, M.Kom*

**Disusun oleh** :

**Alfian Bima Prastyo 21081010101**

**Muhammad Bagus Satrio 21081010102**

**Raisah Nurul Faridah 21081010105**

**Muhamad Raihan Ramadhani Isworo 21081010106**

**Marcellinus Aditya Vitro Darmawan 21081010107**

**PROGRAM STUDI INFORMATIKA**

**FAKULTAS ILMU KOMPUTER**

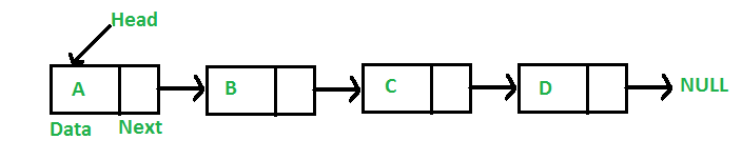
**UNIVERSITAS PEMBANGUNAN NASIONAL “VETERAN” JAWA TIMUR**

**SURABAYA**

**2022**

Struktur data adalah salah satu aspek yang cukup penting bagi setiap orang yang memulai bidang pemrograman dan komputer. Dengan memahami prinsip dan cara kerja sebuah struktur data kita dapat mengimplementasikan ke dalam program dengan lebih baik dan sesuai fungsinya. Salah satunya adalah struktur data *linked list* dan *stack*.

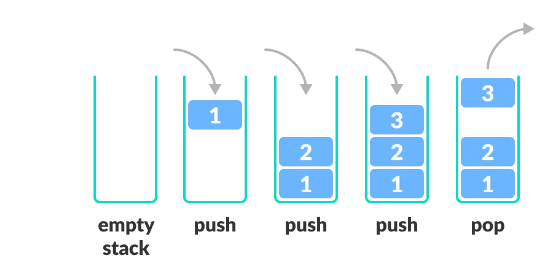
*Linked list* merupakan struktur data linier yang berbentuk rantai simpul dimana setiap simpul menyimpan 2 item, yaitu nilai data dan pointer ke simpul elemen berikutnya. Berbeda dengan array, elemen *linked list* tidak ditempatkan dalam alamat memori yang berdekatan melainkan elemen ditautkan menggunakan pointer.



Simpul pertama dari *linked list* disebut sebagai *head* atau simpul kepala. Apabila *linked list* berisi elemen kosong, maka nilai pointer dari *head* menunjuk ke NULL. Begitu juga untuk pointer berikutnya dari simpul terakhir atau simpul ekor (*tail*) akan menunjuk ke NULL.

Ukuran elemen dari *linked list* dapat bertambah secara dinamis dan mudah untuk menyisipkan dan menghapus elemen karena tidak seperti array, kita hanya perlu mengubah pointer elemen sebelumnya dan elemen berikutnya untuk menyisipkan atau menghapus elemen.

Sedangkan *stack* atau biasa diartikan tumpukan, adalah struktur data linier yang mengikuti prinsip *Last In First Out* (LIFO). Dimana elemen yang terakhir disimpan dalam *stack*, menjadi elemen yang pertama diambil. Untuk meletakkan sebuah elemen pada bagian atas dari *stack*, maka dilakukan operasi *push*. Sedangkan untuk memindahkan sebuah elemen dari tempat atas tersebut dalam sebuah *stack*, maka dilakukan operasi *pop*.



Dari gambar di atas, bisa dilihat bahwa meskipun elemen ke-3 adalah yang paling terakhir ditambahkan, namun elemen tersebut justru yang pertama dihapus. Operasi inilah yang kemudian disebut sebagai prinsip operasi LIFO (*Last In First Out*).

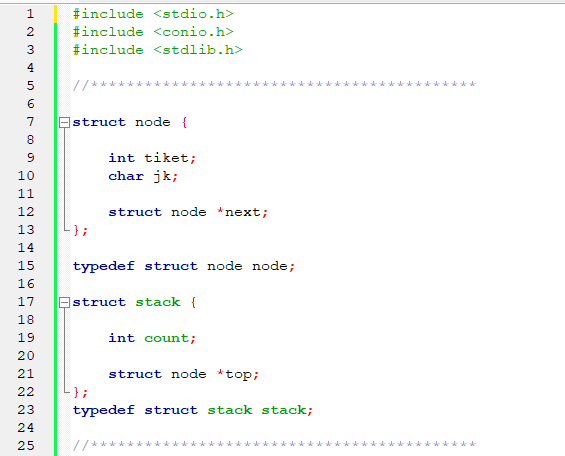
Pada program kami dengan judul “Sistem Operasional Wahana Kereta Hantu” menerapkan dua struktur data, yaitu *linked list* yang digunakan untuk mengatur antrian pembelian tiket sebelum memasuki wahana kereta hantu, dan *stack* yang digunakan untuk memenuhi beberapa peraturan yang ada pada wahana kereta hantu.

Beberapa peraturan yang ada pada wahana kereta hantu:

1. Jumlah maksimum penumpang adalah 12, yaitu jumlah yang sama dengan jumlah maksimum antrian.
2. Dari antrian, akan didahulukan penumpang perempuan sesuai urutan terdepan.
3. Penumpang yang masuk ke wahana akan mengisi kursi paling belakang terlebih dahulu.
4. Kereta hantu dapat dijalankan meskipun penumpang kurang dari jumlah maksimum.
5. Saat wahana selesai, penumpang yang keluar adalah penumpang paling depan terlebih dahulu kemudian dilanjut penumpang di belakangnya.

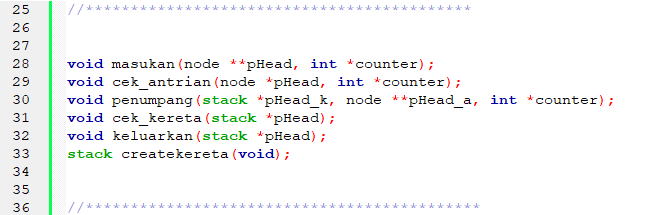
Source Code

1. Deklarasi Library dan Struct



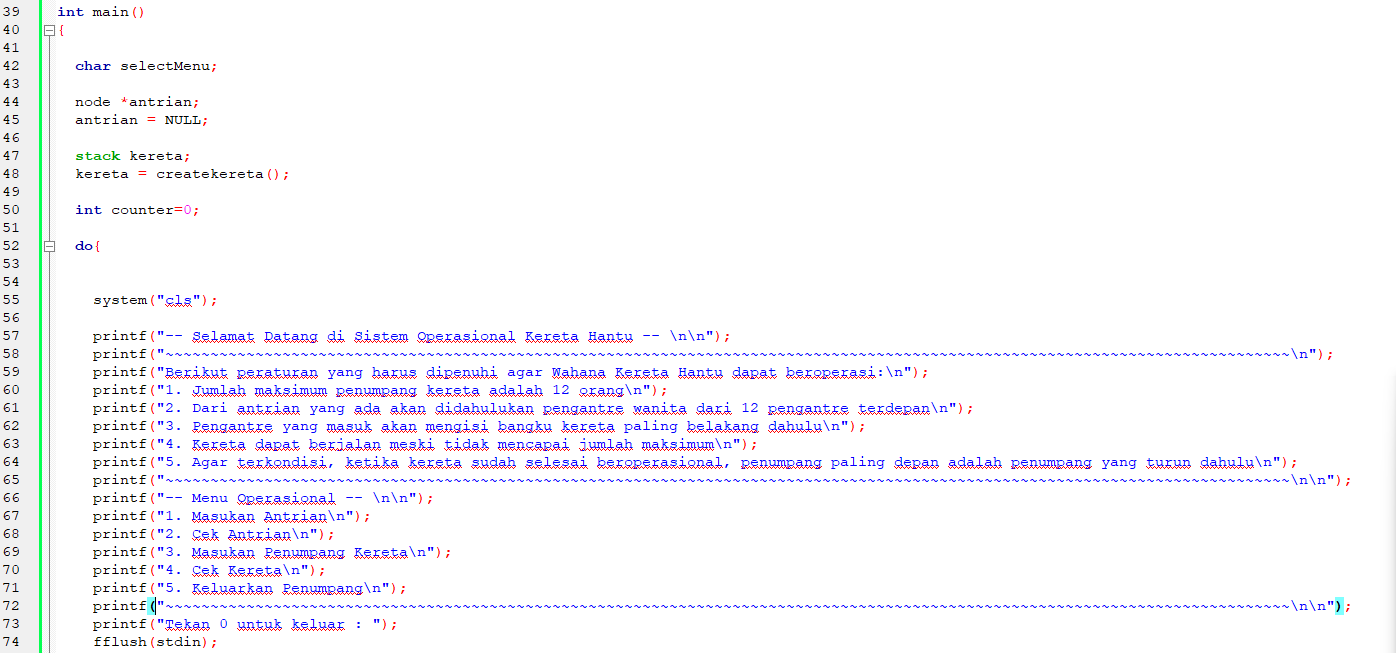
Disini Terdapat Deklarasi dari library dan struct node, struct stack

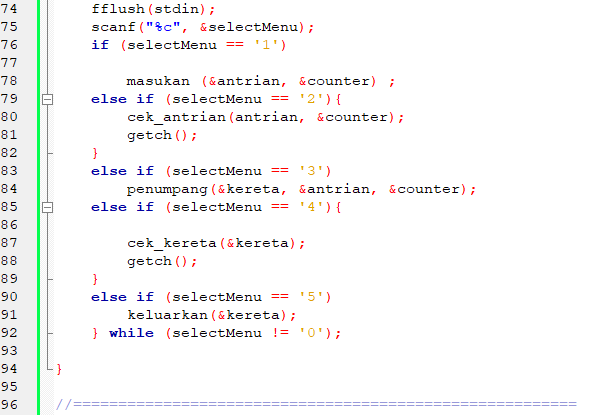
1. Deklarasi Void



Disini ada deklarasi dari beberapa void yaitu void masukan(untuk user menginput data/antrian penumpang), cek\_antrian(untuk mengecek antrian), Penumpang(Untuk memasukan penumpang ke kereta), cek\_kereta(untuk mengecek isi dari kereta) , keluarkan(Untuk mengeluarkan penumpang yang ada di dalam kereta) dan deklarasi untuk membuat stack dari createkereta

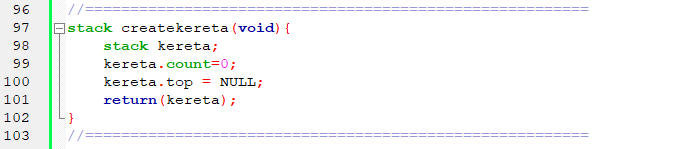
1. Int Main atau inti program





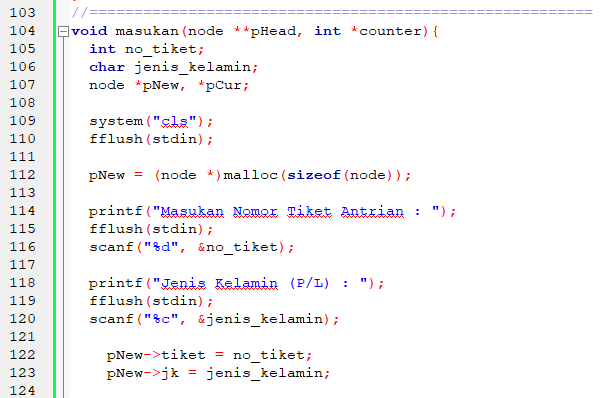
Disini ada kode untuk menampilkan output dari menu program kereta dan ada if else looping untuk pemilihan dari menu

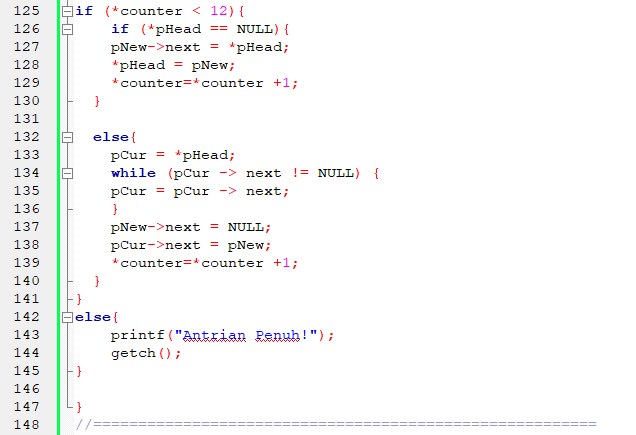
1. Fungsi pembuatan stack createkereta



Disini ada void createkereta yang berfungsi untuk membuat kereta

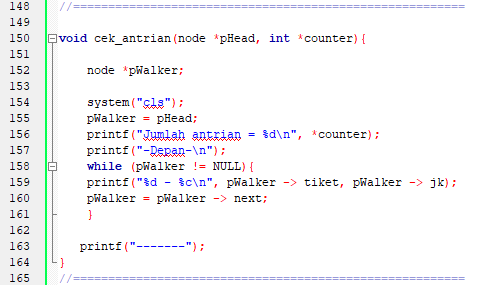
1. Fungsi Memasukkan Data





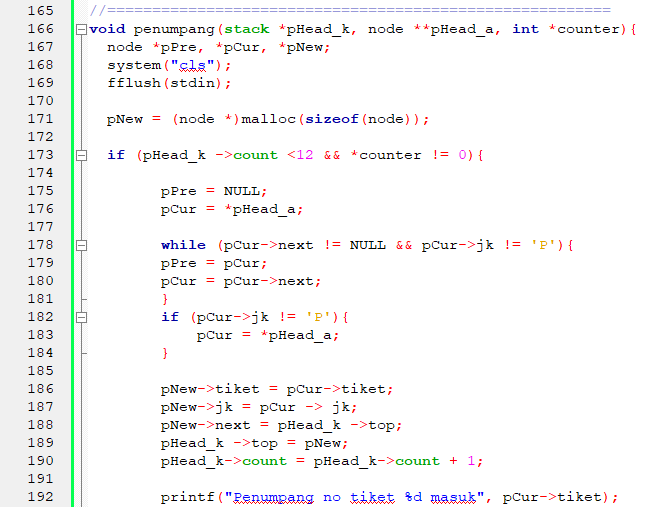
Disini ada void Masukan untuk memasukan data dari penumpang atau Nomor antrian dan apabila antiran telah penuh maka akan mengeluarkan output “Antrian Penuh”

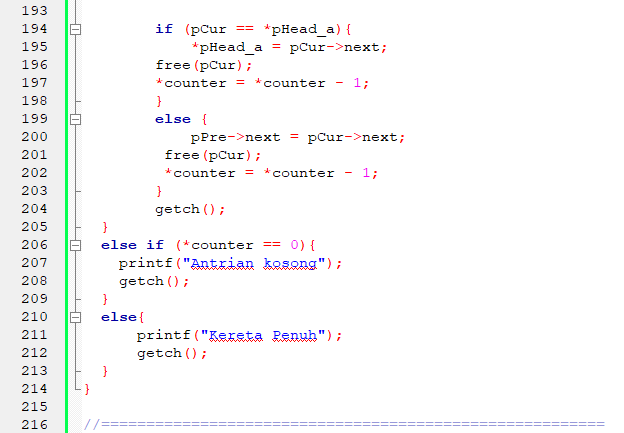
1. Fungsi cek\_antrian



Disini ada void cek\_antrian yang berfungsi untuk menampilkan antrian dari penumpang kereta

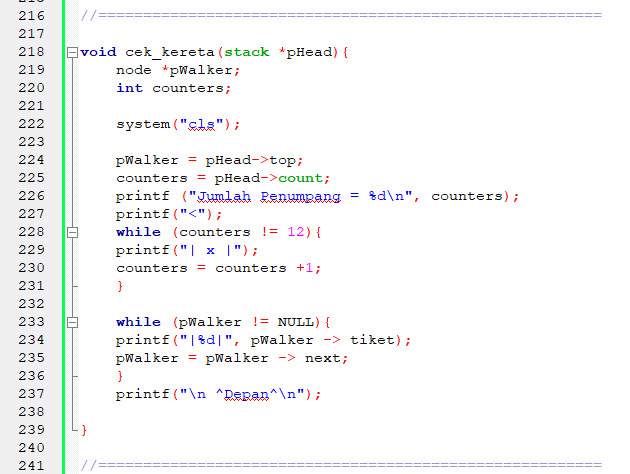
1. Fungsi penumpang





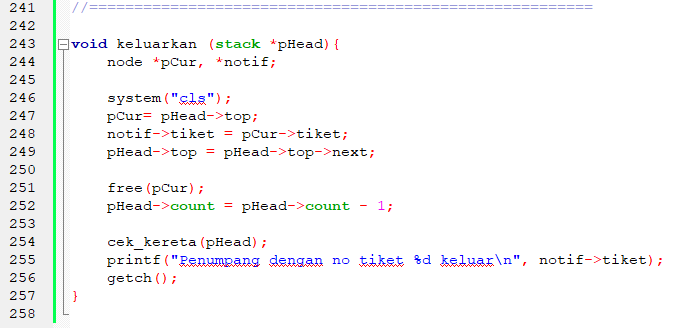
Disini ada void penumpang yang berfungsi untuk memasukan penumpang yang berada di antrian kedalam kereta dan jika di antrian kosong akan menampilkan “Antrian kosong” dan apabila kereta sudah penuh maka akan menampilkan “Kereta Penuh’

1. Fungsi cek\_kereta



Disini ada void cek\_kereta yang berfungsi untuk menampilkan isi dari kereta atau penumpang yang ada di dalam kereta

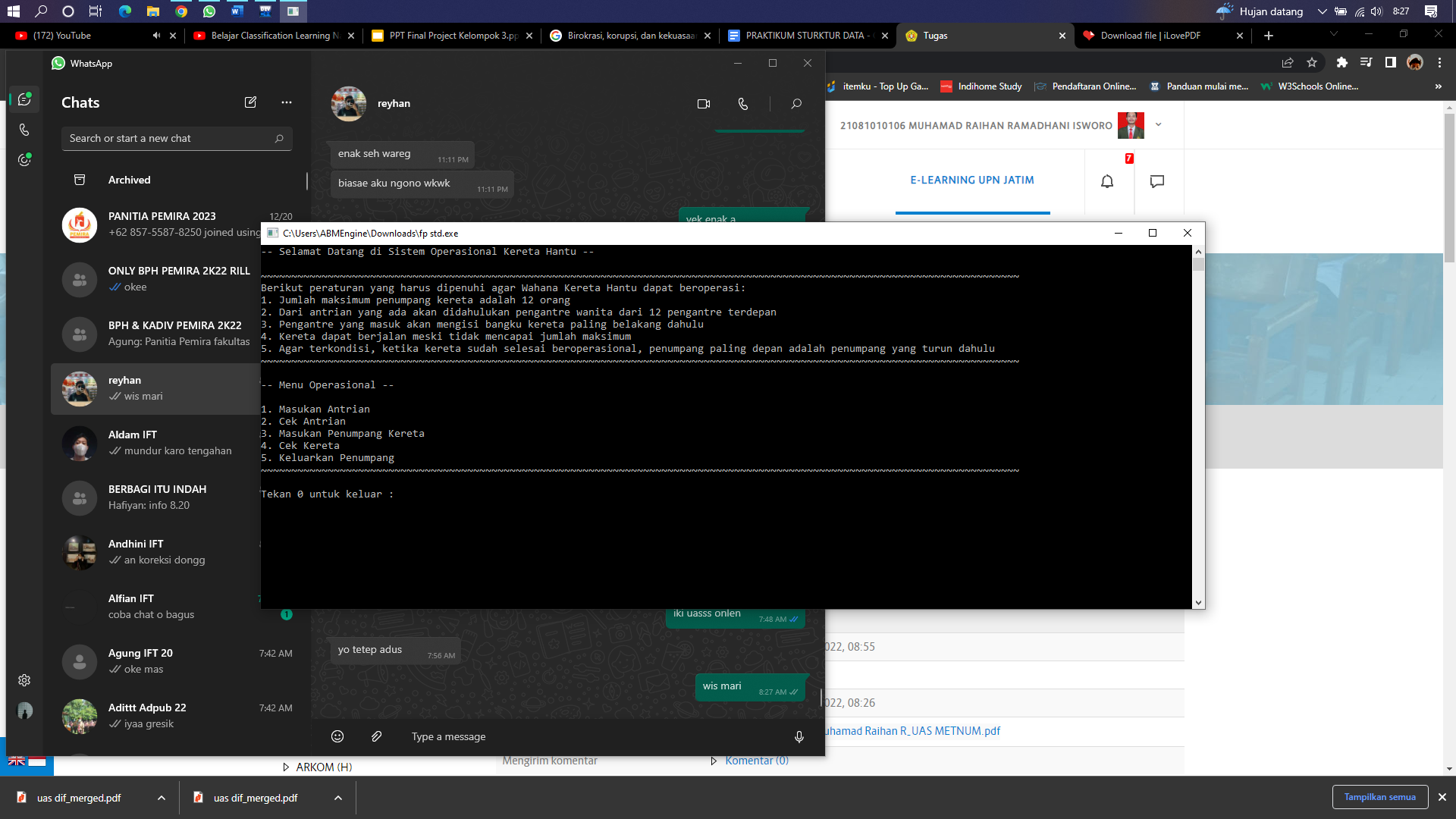
1. Fungsi keluarkan



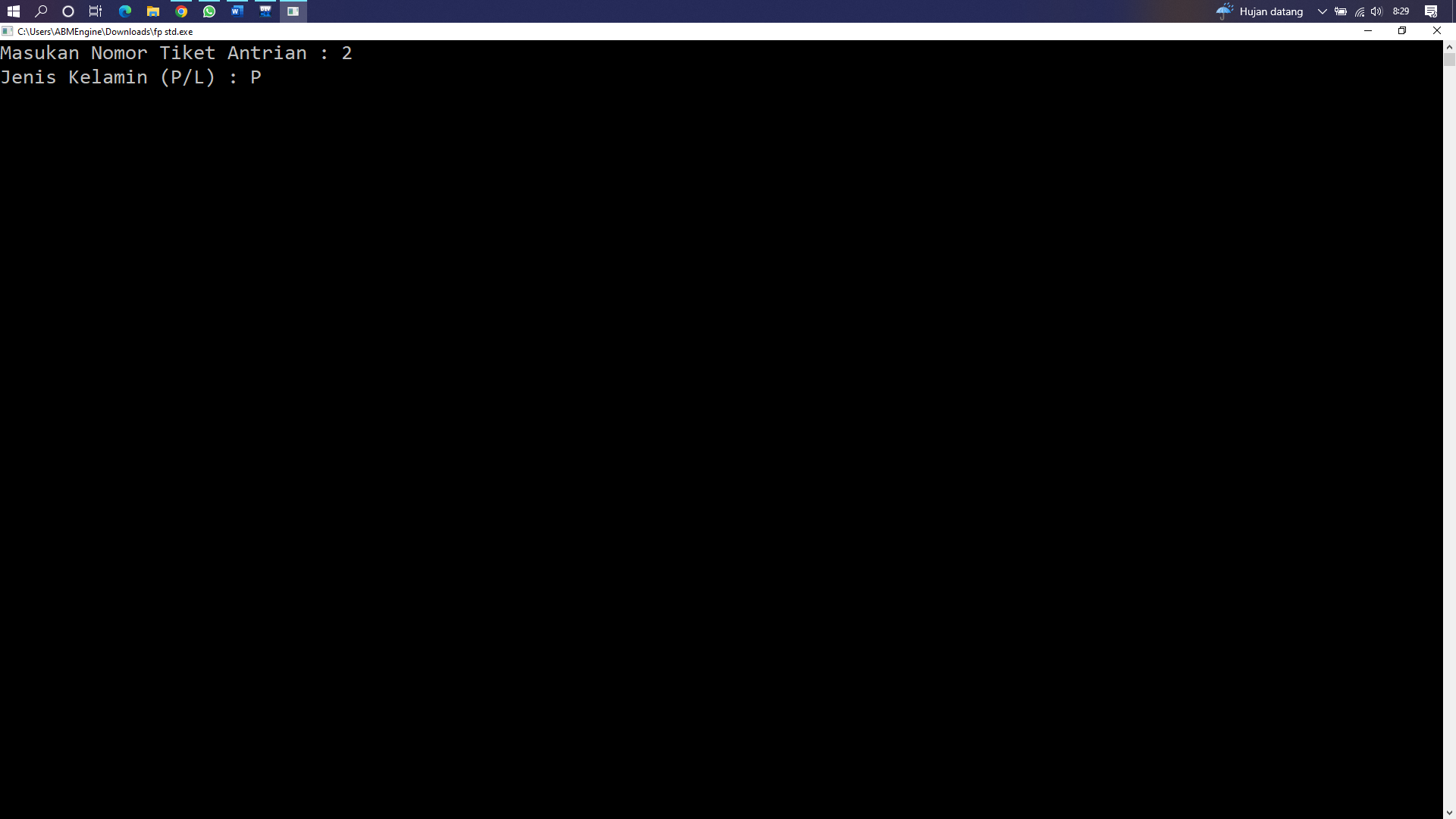
Disini ada void keluarkan yang berfungsi untuk mengeluarkan penumpang yang ada di dalam kereta

Output

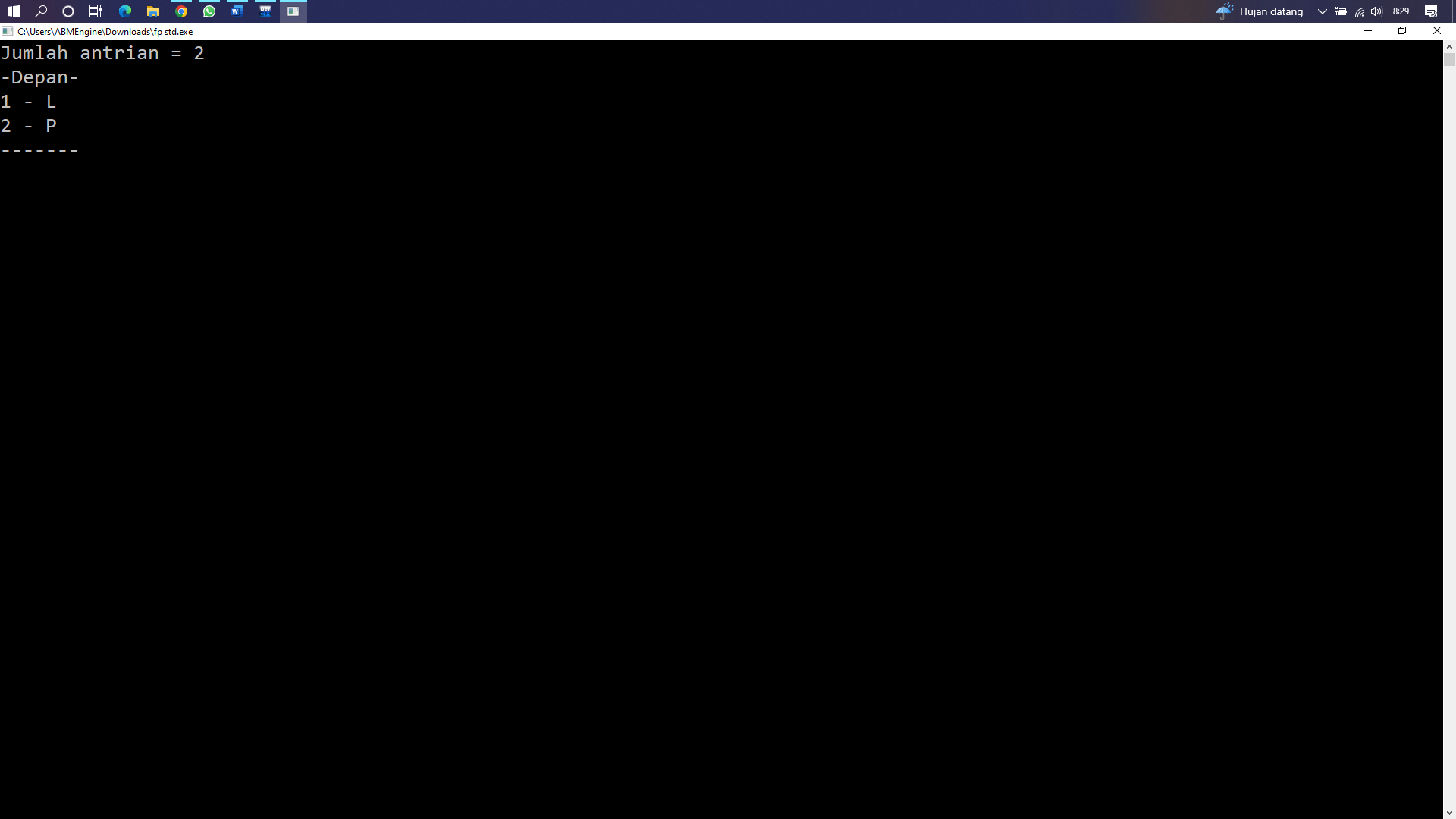
1. Output Menu



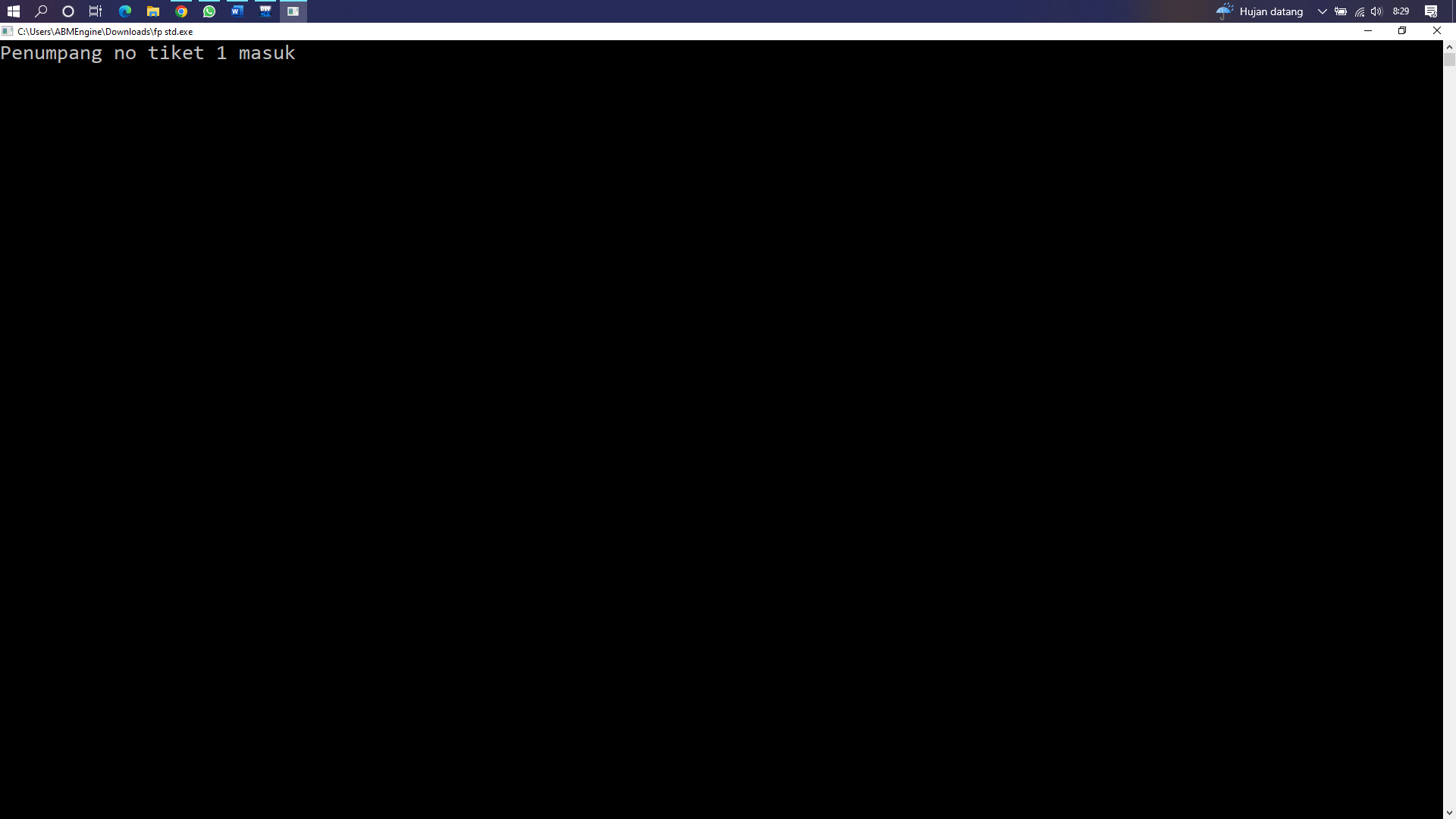
1. Memasukkan Penumpang



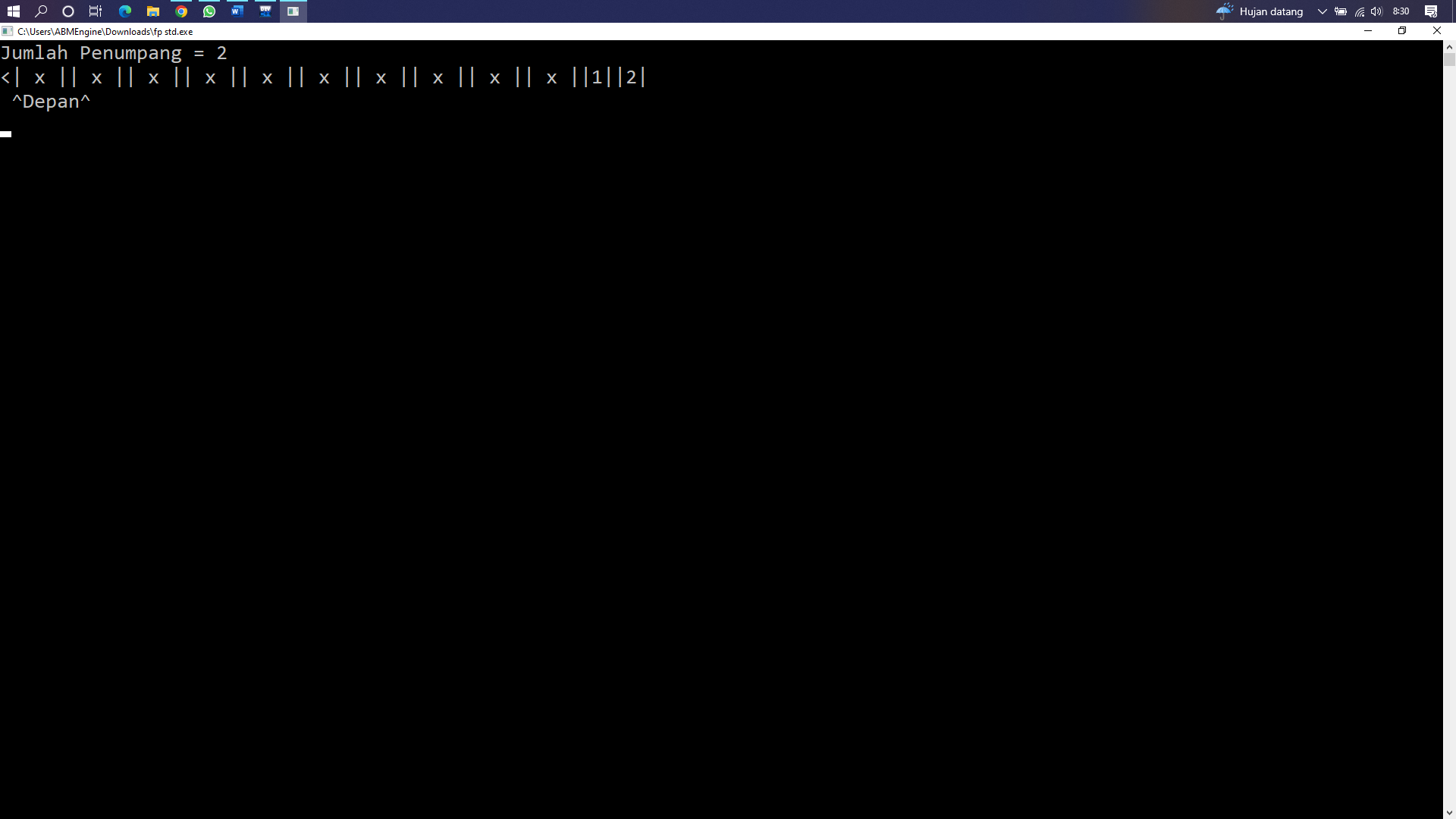
1. Tampilan Antrian



1. Memasukkan antrian ke kereta



1. Tampilan Kereta



1. Mengeluarkan Penumpang dari kereta

