Taller Preparcial-Peaje

Cristian Camilo Morales Robles

Código: 1054552530

Docente: Jhan Carlos Martínez Ceballos

Universidad del Quindío

Facultad de Ingeniería

Ingeniería de Sistemas y Computación

Programación I

Armenia, Quindío

2025

TALLER PREPARCIAL-PEAJE

PENSAMIENTO COMPUTACIONAL

1. **Abstracción**

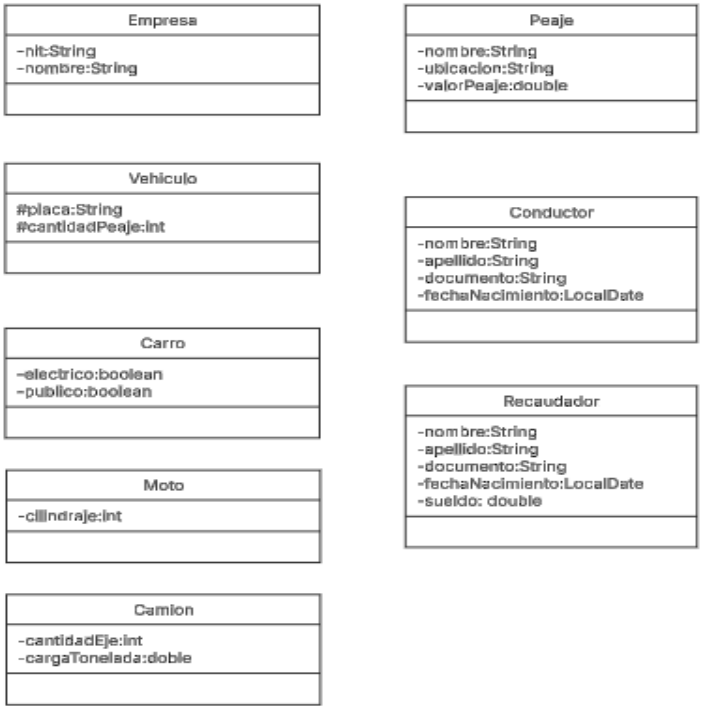
¿Qué se solicita finalmente?

Programa para desarrollar un sistema de peaje

¿Qué información es relevante dado el problema?

Empresa: nit, nombre
Peaje: nombre, ubicación, valor peaje
Vehículo: placa, cantidad peaje
Carro: eléctrico, publico
Moto: cilindraje
Camión: eje, carga tonelada
Conductor: nombre, apellido, documento, fecha nacimiento
Recaudador: nombre, apellido, documento, fecha nacimiento, sueldo

¿Cómo se agrupa la información relevante?



¿Qué funcionalidades se solicitan?

- Almacenar la información de empresa, peaje, vehículo, conductor y recaudador.
- Obtener la información de empresa, peaje, vehículo, conductor y recaudador.
- Actualizar la información de empresa, peaje, vehículo, conductor y recaudador.
- Eliminar la información de empresa, peaje, vehículo, conductor y recaudador.

2. Descomposición

¿Cómo se distribuyen las funcionalidades?

https://lucid.app/lucidchart/f211fbc7-a45d-4e44-92a8-3ef20bd58bc2/edit?viewport_loc=-692%2C1140%2C6652%2C3092%2CHWEp-vi-RSFO&invitationId=inv_6502d55d-85bb-44c1-92d5-d3c9e0adbeb5

¿Qué debo hacer para probar las funcionalidades?

Prueba	Entrada	Salida
Asignar un vehículo al conductor assertTrue(true)	Empresa empresa = new Empresa (“8100” , “PeajeUQ”); Conductor conductor= new Conductor (“Juan”,” Restrepo”,”1024” …); Carro carro = new Carro (“AZE879”,5, false, true); empresa. agregarConductor(conductor); empresa. asignarVehiculo (“1024”, carro”)	**True** (el carro fue asignado a la lista del conductor

Reporte detallado de los vehículos (carro, moto, camión)	Empresa empresa = new Empresa (“8100” , “PeajeUQ”);
	Carro carro=new Carro (“MZD720”…);
	Moto moto=new Moto (“IGS71E” …);
	Camion camion =new Camion (“IST250”….);
	empresa.agregarVehiculo(carro);
	empresa.agregarVehiculo(moto);
	empresa.agregarVehiculo(camion);
		Muestra la información detallada del carro, moto y camion con su recaudo total

Generar reporte de vehiculo donde muestre la descripción de	Empresa empresa = new Empresa (“8100” , “PeajeUQ”); Carro carro=new Carro(“MZD720”..); Moto moto=new Moto(“IGS71E”…);	Muestra la información de la descripción de los vehículos (carro, moto, camión)

cada vehiculo (carro, moto, camion)	Camion camion =new Camion(“IST250”….);
	empresa.agregarVehiculo(carro);
	empresa.agregarVehiculo(moto);
	empresa.agregarVehiculo(camion);	

Consultar total peaje por persona con un assertEqual(Total del peaje de la persona)	Empresa empresa = new Empresa (“8100” , “PeajeUQ”); Conductor conductor= new Conductor (“Juan”,” Restrepo”,”1024” …); empresa.agregarConductor(conductor); Carro carro=new Carro (“MZD720”…); Moto moto=new Moto (“IGS71E” …); Camion camion =new Camion (“IST250”….); empresa. asignarVehiculo (“1024”, carro”); empresa. asignarVehiculo (“1024”, moto”); double totalCalculado=empresa.consultarTotal PeajePorPersona(“1025”);	**Equal** (realiza una suma del calculo de los peajes del método para carro y moto) y me compara si el método mencionado si es igual

Obtener vehiculo por conductor, assertEqual y assertTrue para verificar si es igual el documento y el tipo del vehiculo	Empresa empresa = new Empresa (“8100” , “PeajeUQ”);
	Conductor conductor= new Conductor (“Juan”,” Restrepo”,”1024” …);
	empresa.agregarConductor(conductor);
	Carro carro=new Carro (“MZD720”…);
	Moto moto=new Moto (“IGS71E” …);
	Camion camion =new Camion (“IST250”….);
	empresa.agregarVehiculo(carro);
	empresa.agregarVehiculo(moto);
	empresa.agregarVehiculo(camion);
	LinkendList<Vehiculo>vehiculos= empresa.obtenerVehiculoPorConductor(“1024”,”carro”);
	assertEqual(1,vehículos.size());
	assertTrue(vehilo.contains(carro);
		assertEqual (Es donde va igual dentro la lista de vehículos con el método, si existe un carro dentro la lista) **True** (me sale un verdadero, donde verifica la lista si contiene un carro, si existe me muestra un verdadero, de lo contrario me arroja un **false**)

Registrar paso de vehículo (carro, moto, camion) con assertTrue	Empresa empresa = new Empresa (“8100” , “PeajeUQ”); Carro carro=new Carro (“MZD720”…); empresa.agregarVehiculo(carro); empresa.registrarPasoVehiculo (“MZD720”);	Si registramos la placa nos va a arroja un verdadero, pero si el vehículo no ah pasado por registro no arroja un falso
Buscar recaudador por nombre con un assertNotNull	Empresa empresa = new Empresa (“8100” , “PeajeUQ”);	Primero que existe dentro de la variable resultado un

y con un assertEqual	Recaudador recaudador = new Recaudador (“Adriana”, “Rodriguez”, “2530”…);
	empresa.agregarRecaudador(recaudador);
	Recaudador resultado=empresa.buscarRecaudadorPor Nombre(“AdrianaRodriguez”);
	assertNotNull(resultado);
	assertEquals(“2530”,resultado.get Documento());
		recaudador, y si no existe nos arroje un error que no existe, y si existe que lo compare con el documento del recaudador para que arroje un true

Obtener conductor con camion alto tonelaje con un assertEquals y assertTrue	Empresa empresa = new Empresa (“8100” , “PeajeUQ”);
	Conductor conductor1= new Conductor (“Anderson”,” Rodriguez”,”1057” …);
	Conductor conductor2= new Conductor (“Ana”,” Benavidez”,”1007” …);
	Conductor conductor3= new Conductor (“Carlos”,” Rodriguez”,”7474” …);
	empresa.agregarConductor(conductor1);
	empresa.agregarConductor(conductor2);
	empresa.agregarConductor(conductor3);
	empresa.asignarVehiculo(camion1);
	empresa.asignarVehiculo(camion2);
	empresa.asignarVehiculo(camion3);
	LinkedList<Conductor>resultado= empresa.obtenerConductorConCamion AltoTonelaje();
	assertEquals(3,resultado.size());
	assertTrue(resultado.contains(conductor1));
	assertTrue(resultado.contains(conductor2));
	assertTrue(resultado.contains(conductor3));
		Como resultado se agrego 3 conductores a una lista, donde verifica si existen los conductores y cumplen con la condición del tonelaje mayor a 10 nos arroja un verdadero

3. **Reconocimiento de patrones**
   ¿Qué puedo reutilizar de la solución de otros problemas?

4. **Codificación**

   https://github.com/cristianm-98/TallerPreparcial-Peaje.git