Proyecto programado #1

Nicole de los Ángeles Araya Ballestero

EIF 204 - Programación II

NRC 41722 - Grupo 03

Ingeniería en Sistemas, Universidad Nacional de Costa Rica

Profesor Cristopher Montero Jiménez

14 de mayo del 2022

Objetivos del programa

Lograr crear un programa de venta de tiquetes para una empresa de Buses, en la cual se inserta varias clases que permiten su buen funcionamiento. Y así el usuario pueda lograr hacer lo que necesite para su buen uso.

Limitaciones del programa

- Si quiere ingresar varios buses debe ingresar a la opción varias veces para insertar la cantidad de buses que se quiere.
- No se cargan los tiquetes que están guardados en el programa.
- En la opción Reservar de Tiquetes, si hay una ruta registrada pero no tiene buses, igual aparecerá, pero no se podrá reservar ningún tiquete y se devolverá al menú.
- Si atrapa un error el programa siempre se devolverá al menú Principal.
- El identificador del tiquete será la cedula que la persona ingrese.
- Si ingresa más de una letra en los menús, deberá darle “enter” hasta que se logre limpiar el buffer por la cantidad digitada.
- Si no cierra el programa a través de la opción “4. Salir” no se guardará los nuevos datos que se hayan digitado.
- Poner los identificadores de Ruta y de Transporte diferentes, ya que a la hora de cargar los datos al programa, se puede confundir la Ruta con un Transporte.

Alcances del programa

- El programa no permite que haya un transporte repetido en otra ruta.
- El programa no permite que exista un identificador repetido en ninguna clase.
- Si el programa tiene algún problema devolverá una excepción, un ejemplo, que no encuentre lo digitado.
- El programa muestra toda la información que requiera la persona para realizar lo que necesite según en el menú en el que este.
- Si no existen los nombres de los archivos aun, donde guardara la información deseada, el programa se encarga de crear los archivos para su buen funcionamiento.
- Se guarda toda la información vital del programa que sea digitada, lo que seria las rutas y sus buses vinculados, los buses que existan y sus tiquetes comprados.
- Al iniciar el programa se carga toda la información almacenada respecto a las rutas y sus buses vinculados y los buses que existan.
- A la hora de elegir una opción en los menús, y se ingresa una letra, el programa no se caerá y tirará una excepción.
- Al reservar tiquetes, el usuario puede digitar la cantidad de espacios que desea obtener, y si al asignar esos espacios ocupa agarrar también espacio del otro bus, el tiquete procederá a almacenar esas placas que se están utilizando. Obteniendo un tiquete con n placas ocupadas por el usuario.

Descripción de la solución planteada

Justificación de clases involucradas

- La clase Object contienen sus atributos específicos y un método virtual puro, siendo la superclase, heredando a Ruta, Tiquete, Transporte.
- La clase Transporte contienen sus atributos específicos y un vector de sus espacios, y le hereda a “RestricciónTransporte” para poner restricciones a los Transportes.
- Clase RutaReservar contienen métodos relacionados a la reserva de espacios de asientos.
- Las clases que terminan en ...BDatos se crearon para guardar información.
- La Listas contienen a sus propias clases y se relacionan con su respectiva “base de datos” (…BDatos) para guardar.
- La clase Excepción contiene un método string que será heredado a sus otras clases para que devuelvan un mensaje especifico cada una.
- Clase Archivos se encarga de leer y cargar la información.
- Clase BusCar es la controladora.

Descripción de la funcionalidad y objetivo de cada clase

- Object: Superclase que hereda a sus hijos, contiene solo un atributo y sus métodos que devuelven o cambian al atributo, tanto como un método puro.

- Transporte: Tiene los atributos necesarios y sus métodos que devuelven o cambia esos atributos y hereda de su padre Object. También tiene un vector de asientos, inicializando su memoria en ‘D’ (disponible). Entre otros métodos vacíos que serán reescritos por la clase RestriccionTransporte.

- RestriccionTransporte: Clase que se encarga de la restricción sanitaria y de rescribir los métodos de la clase padre, para ocupar los asientos.

- Tiquete: Tiene todos los atributos para la información de Tiquete y sus métodos que devuelven o cambia esos atributos, y hereda de su padre Object.

- TiqueteBDatos, TransporteBDatos, RutaBDatos: Son clases que reciben por parámetro un tiquete, ruta o transporte para guardar los datos y sus métodos son statics

- Archivo: Recibe por parámetros la ListaRuta y ListaTransporte, para leer y cargar los datos guardados, tanto como crear los archivos si no existen y sus métodos son statics

- Ruta: Tiene los atributos necesarios y sus métodos que devuelven o cambia esos atributos, tanto como una lista de transportes que contiene. Y hereda de su padre Object.

- RutaReserva: Es una clase que recibe por parámetro una Ruta para reservar en ella y saber sobre la disponibilidad y sus métodos son statics.

- ListaRuta, ListaTransporte, ListaTiquete: Son colecciones que pueden contener información de rutas, transportes y tiquetes sin tener un tamaño limite. Sus métodos hacen todo lo que permita un buen manejo de su información.

- BusCar: Contiene punteros a las todas las Listas, y se encarga de crear, mostrar, cargar y guardar la información. Sus métodos son la controladora, por lo que son menús.

- main: Donde se ejecuta el programa.

Descripción relaciones incluidas

- Clase Object le hereda a Ruta, Tiquete y Transporte.

- La clase RutaReserva usa a Ruta.

- La clase RutaBDatos usa a la Clase Ruta.

- La clase TiqueteBDatos usa a la Clase Tiquete.
- La clase TransporteBDatos usa a la Clase Transporte.
- La clase Transporte le hereda a Restricción Transporte.
- La clase Ruta tiene una clase Lista Transporte.
- La clase Excepción le hereda a NotFound, Found, Downloading, LetterOption y Existence.
- BusCar tiene relación con las Listas y usa la clase Excepción y Archivos.