Programming 122 – F22 - Final

You are tasked to write an app for a client that will allow them to keep track of cars for their customers. The customer should be able to display all their cars, add cars, filter cars by different criteria, see all of their customer’s cars, and display the average mileage of any cars listed.

or GitHub Classroom:

Graphical user interface, application

Description automatically generated

Rubric

|  |
| --- |
| Basics – 20 Points |
| Commented header at top of code ( name, date, assignment )  Comments throughout code explaining functionality  Proper use of try / catch  Proper use of TryParse() |

|  |
| --- |
| GUI – 60 points *All use proper naming convention* |
| Add Car   * Make – Radio Buttons ( 1 for each make ) * Model – Text box * Year – Combo Box ( Populated with a range of years from 1970 to 2022 ) * Features – Check Boxes ( 1 for each feature, GPS and Backup Cam ) * Mileage – Text box * Owners – List Box ( Display names of all owners )   Filter By Make – Radio Buttons ( 1 for each make )  Filter By Range of Year – 2 Combo Boxes, populated with a year range from 1970 to 2022  Filter By Mileage ( Above ) – Combo box, populated from 0 to 100,000, in 10,000 increments  Display Cars by Owner – ListBox : Displays all owners, includes number of cars and average miles  Reset Button – Displays all the cars  ListView – Displays all cars. Headers for ( make, model, year, GPS, Backup Cam, Mileage )  Average Miles of Displayed Cars – Displays Average Miles for the current list of cars, even filtered |

|  |  |
| --- | --- |
| Methods Required – 60 points | |
| ***Display***  void DisplayCars(List<Car>)  void DisplayAverageMileage(List<Car>())  void DisplayOwnersDetailed(List<Owner))  ***Filter***  List<Car> ByMake(Car.Makes, List<Car>())  List<Car> ByYear(int, int, List<Car>())  List<Car> ByMileage(int, List<Car>())  ***Classes***  Owner – double AverageCarMileage()  Owner – void AddCar(Car) | ***Functionality***  void FormatCarInfo(Car)  void AddCar(Car)  ***Preloads***  void Preload()  void GenerateCars(int)  void PopulateOwners()  ***Gui Population***  void FillYearComboBox()  void FillMileageComboBox() |

|  |
| --- |
| *Car class –20 points* |
| Enumerator: Makes – {4 car makes, your choice}  Fields:  \_make: Makes  \_model: String  \_year: int  \_hasGPS: bool  \_hasBackUpCam: bool  \_mileage: int |

|  |
| --- |
| Owner *class – 20 points* |
| Fields:  \_name: string  \_cars: List<Car>  Method:  double AverageMileage() void AddCar(Car) |

|  |
| --- |
| *Data Management – 20 Points* |
| One car list in Form1  One Owner list in Form1  Each Owner will have their own list of cars.  All cars and owners will be associated with these lists   * All Cars are added to the main Car list AND an Owners car list * All Owners are added to the Owner list |

Classes

*Car*

Enumerator: Makes { Add 4 car makes of your choice }   
Ex. Toyota, Honda, Nissen, Subaru

Fields

* Make : Makes
* Model: string
* Year: int
* HasGps: bool
* HasBackUpCam: bool
* Mileage: int

Constructor - Takes all Fields

Properties ( getters and setters for all )

No Methods

*Owner*

Fields

* Name: string
* Cars: List<Car>

Constructor - Name

Properties: Name ( getter / setter ), Cars ( getter )

*Void AddCar(Car car)*: Adds a car to the Cars list

*Double AverageMileage()*: Calculates the average mileage for the Owners cars

Display

*void DisplayCars(List<Car>)*

Write a method that takes a list of cars and display the information to the list view. This should work with any list of cars you pass it, including your list of filtered car results. All List View formatting should be done with your *FormatCarInfo(Car)* listed method down below. You should just loop through the list with this method.

*void DisplayAverageMileage(List<Car>())*

Write a method that takes a list of cars, and displays the average mileage of the cars displayed on the list to a label placed around the list view ( you can choose where. See image for example).

Tip: Think about the best place to call this method. Is there another method that is called that takes a car list and displays a new list of cars?

*void DisplayOwnersDetailed(List<Owner))*

Write a method that populates a Listbox from the owner’s list. This should also format the information to display the owner’s name, the number of cars they own, and the average mileage of all their cars.

Tip: What variable correlates with how many cars they have on their list?

Filter

*List<Car> ByMake(Car.Makes, List<Car>())*

Write a method that takes a **Makes** enumerator, and a list of cars. Return a list of all cars of that makes.

Ex. If you pass in Car.Makes.Ford, a list of all Ford Cars should be returned.

*List<Car> ByYear(int from, int to, List<Car>())*

Write a method that takes two ints, both representing years, low and high. Your method should return a list of cars whose years fall between those values.

Ex. Low = 1982, High = 2004. A list of cars between the years of 1982 – 2004 should be returned.

*List<Car> ByMileage(int, List<Car>())*

Write a method that takes a single int, this represents a mileage. Returns a list of cars whose mileage is OVER that amount.

Ex. ByMileage(50000, cars) should return a list of call cars whose mileage is 50,000 or above

Gui Population

*void FillYearComboBox()*

Write a method that populates your year combo boxes on load. There should be a range from 1970 to 2022. This should be done with a loop.

*void FillMileageComboBox()*

Write a method that populates your Mileage Combo box with values starting at 0 and going up to 100,000, in 10,000 increments. Use a loop for this.

Functionality

*void FormatCarInfo(Car)*

Write a method that takes a car parameter. Then formats and displays the information correctly to the list view display. This method should only take a single car. But you can call this method other methods to display more cars.

*void AddCar()*

Write this method to gather all the information from the winform, create a new car object, and add that object to the selected owner ( from the list box ) and the car list. The ListView and the Owners ListBox should both update to reflect the new car.

Preloads

*void Preload()*

When this method is called, it will call all other methods that populate your lists and fields when your first run your application. This includes

* *PopulateOwners*
* *GenerateCars*
* *FillYearComboBox*
* *FillMileageComboBox*

Make sure these are called in the correct order, otherwise things may not load properly.

*void GenerateCars(int)*

Write a method that will generate a list of cars and place them both in the car List and a randomly selected owner. **We shall discuss this more in class**

*void PopulateOwners()*

Write a method to populate the owner’s list with a minimum of 5 owners. These can be manually created and you can use whomever you want.