Antonio Ennis

Colorado State University – Global Campus

ITS320 – Basic Programming

Professor Russel Frith

September 10, 2023

For the portfolio project students are required to complete three main objectives. The third objective is creating a final program using the skills learned over the last two months. The final program’s main objective is to create an automobile class that will be used by a dealership as a vehicle inventory program. The program students are asked to make is simple and could be considered a prototype with its limited functionality.

**Source Code:**

**# Define the Vehicle class to represent individual vehicles**

class Vehicle:

def \_\_init\_\_(self, make, model, color, year, mileage):

**# Initialize attributes with the values passed in**

self.\_\_make = make

self.\_\_model = model

self.\_\_color = color

self.\_\_year = year

self.\_\_mileage = mileage

**# Method to update vehicle attributes**

def update\_attributes(self, make=None, model=None, color=None, year=None, mileage=None):

**# Update attributes if new values are provided**

if make:

self.\_\_make = make

if model:

self.\_\_model = model

if color:

self.\_\_color = color

if year:

self.\_\_year = year

if mileage:

self.\_\_mileage = mileage

**# String representation of the Vehicle object**

def \_\_str\_\_(self):

return f"{self.\_\_make}, {self.\_\_model}, {self.\_\_color}, {self.\_\_year}, {self.\_\_mileage}"

**# Define the VehicleInventory class to manage a collection of vehicles**

class VehicleInventory:

def \_\_init\_\_(self):

**# Initialize an empty list to store vehicles**

self.\_\_vehicles = []

**# Method to add a new vehicle to the inventory**

def add\_vehicle(self, vehicle):

self.\_\_vehicles.append(vehicle)

**# Method to remove a vehicle from the inventory by its index**

def remove\_vehicle(self, index):

if 0 <= index < len(self.\_\_vehicles):

del self.\_\_vehicles[index]

**# Method to update a vehicle's attributes by its index**

def update\_vehicle(self, index, \*\*kwargs):

if 0 <= index < len(self.\_\_vehicles):

self.\_\_vehicles[index].update\_attributes(\*\*kwargs)

**# Method to save the vehicle inventory to a text file**

def save\_to\_file(self):

with open("vehicle\_inventory.txt", "w") as f:

for vehicle in self.\_\_vehicles:

f.write(str(vehicle) + '\n')

**# String representation of the VehicleInventory object**

def \_\_str\_\_(self):

return "\n".join(str(vehicle) for vehicle in self.\_\_vehicles)

**# Main program**

if \_\_name\_\_ == "\_\_main\_\_":

**# Create a new VehicleInventory object**

inventory = VehicleInventory()

while True:

**# Display the menu to the user**

print("1: Add a new vehicle")

print("2: Remove a vehicle")

print("3: Update vehicle attributes")

print("4: Display all vehicles")

print("5: Save to file and exit")

**# Get the user's choice**

choice = input("Enter your choice: ")

**# Perform the appropriate action based on the user's choice**

if choice == '1':

make = input("Enter make: ")

model = input("Enter model: ")

color = input("Enter color: ")

year = int(input("Enter year: "))

mileage = int(input("Enter mileage: "))

vehicle = Vehicle(make, model, color, year, mileage)

inventory.add\_vehicle(vehicle)

elif choice == '2':

print(inventory)

index = int(input("Enter index of vehicle to remove: "))

inventory.remove\_vehicle(index)

elif choice == '3':

print(inventory)

index = int(input("Enter index of vehicle to update: "))

make = input("Enter new make (leave empty to keep current): ")

model = input("Enter new model (leave empty to keep current): ")

color = input("Enter new color (leave empty to keep current): ")

year = input("Enter new year (leave empty to keep current): ")

mileage = input("Enter new mileage (leave empty to keep current): ")

kwargs = {}

if make:

kwargs['make'] = make

if model:

kwargs['model'] = model

if color:

kwargs['color'] = color

if year:

kwargs['year'] = int(year)

if mileage:

kwargs['mileage'] = int(mileage)

inventory.update\_vehicle(index, \*\*kwargs)

elif choice == '4':

print(inventory)

elif choice == '5':

**# Save to file and exit**

inventory.save\_to\_file()

break

**Program Interaction:**

A black screen with many small colored dots

Description automatically generated with medium confidence

A screen shot of a computer screen

Description automatically generated

**References**

*How to use variable-length arguments in a function in Python*. (n.d.). How to Use Variable-length Arguments in a Function in Python. https://www.tutorialspoint.com/How-to-use-variable-length-arguments-in-a-function-in-Python

Besbes, A. (2022, January 22). *How To Use Variable Number of Arguments in Python Functions*. Medium. https://towardsdatascience.com/how-to-use-variable-number-of-arguments-in-python-functions-d3a49a9b7db6

Python, R. (n.d.). *Python Classes: The Power of Object-Oriented Programming – Real Python*. Python Classes: The Power of Object-Oriented Programming – Real Python. https://realpython.com/python-classes/

*Python Classes and Objects - GeeksforGeeks*. (2019, October 15). GeeksforGeeks. https://www.geeksforgeeks.org/python-classes-and-objects/