main_code

December 6, 2024

[ ]:
```
class CarRental:
    def __init__(self, stock):
        """
        Constructor for CarRental class.
        :param stock: Total number of car available for rent.
        """
        self.stock = stock

    def display_cars(self):
        """
        Displays the number of cars available for rent.
        """
        print(f"We have currently {self.stock} cars available to rent.")
        return self.stock

    def rent_hourly(self, n, hours):
        """
        Rents cars on an hourly basis.
        :param n: Number of cars to rent.
        :param hours: Number of hours to rent the cars.
        :return: Rental time.
        """
        if n <= 0 or hours <= 0:
            print("Number of cars and hours should be positive!")
            return None
        elif n > self.stock:
            print(f"Sorry, we have currently only {self.stock} cars available␣
↪to rent.")
            return None
        else:
            self.stock -= n
            now = datetime.now()
            return now, hours

    def rent_daily(self, n, days):
        """
        Rents cars on daily basis.
```

```
        :param n: Number of cars to rent.
        :param days: Number of days to rent the cars.
        :return: Rental time.
        """
        if n <= 0 or days <= 0:
            print("Number of cars and days should be positive!")
            return None
        elif n > self.stock:
            print(f"Sorry, we have currently only {self.stock} cars avialable␣
↪to rent.")
            return None
        else:
            self.stock -= n
            now = datetime.now()
            return now, days

    def rent_weekly(self, n, weeks):
        """
        Rents cars on a weekly basis.
        :param n: Number of cars to rent.
        :param weeks: Number of weeks to rent the cars.
        :return: Rental time.
        """
        if n <= 0 or weeks <= 0:
            print("Number of cars and weeks should be positive!")
            return None
        elif n > self.stock:
            print(f"Sorry, we have currently only {self.stock} cars avialable␣
↪to rent.")
            return None
        else:
            self.stock -= n
            now = datetime.now()
            return now, weeks

    def return_car(self, request):
        """
        Returns a rented car.
        :param request: Tuple containing rental time, rental basis, rental␣
↪duration, and number of cars rented.
        :return: Bill amount.
        """
        rentalTime, rentalBasis, rentalDuration, numOfCars = request
        bill = 0

        if rentalTime and rentalBasis and rentalDuration and numOfCars:
            self.stock += numOfCars
```

```
            now = datetime.now()
            rentalPeriod = now - rentalTime

            if rentalBasis == 1: # hourly
                bill = rentalDuration * 5 * numOfCars
            elif rentalBasis == 2: # daily
                bill = rentalDuration * 20 * numOfCars
            elif rentalBasis == 3: # weekly
                bill = rentalDuration * 60 * numOfCars

            if numOfCars >= 2:
                print("You have an extra 10% discount")
                bill = 0.9

            print(f"Thank you for returning your car(s). Your total bill is␣
↪${bill:.2f}")
            return bill
        else:
            print("Are you sure you rented a car with us?")
            return None
```

[ ]:
```
class Customer:
    def __init__(self):
        """
        Constructor for Customer class.
        """
        self.cars = 0
        self.rentalBasis = 0
        self.rentalTime = None
        self.rentalDuration = 0

    def request_car(self):
        """
        Requests the car rental.
        :return: Number of cars to rent
        """
        cars = input("How many cars would you like to rent? ")
        try:
            cars = int(cars)
        except ValueError:
            print("That's not positive integer!")
            return -1

        if cars < 1:
            print("Invalid input. Number of cars should be greater than zero!")
            return -1
        else:
```

```
            self.cars = cars
            return self.cars

    def request_duration(self):
        """
        Requests the rental duration.
        :return: Rental duration.
        """
        duration = input("For how long would you like to rent the cars (in␣
↪hours/days/weeks)? ")
        try:
            duration = int(duration)
        except ValueError:
            print("That's not a positive integer!")
            return -1
        if duration < 1:
            print("Invalid input. Duration should be greater than zero!")
            return -1
        else:
            self.renatalDuration = duration
        return self.rentalDuration

    def return_car(self):
        """
        Returns the car rental.
        :return: A Tuple containing rental time, rental basis, rental duration,␣
↪and number of cars rented.
        """
        if self.rentalBasis and self.rentalTime and self.cars:
            return self.rentalTime, self.rentalBasis, self.rentalDuration, self.
↪cars
        else:
            return None, None, None, None
```

[ ]:
```
from car_Temp import CarRental, Customer

def main():
    rental_Shop = CarRental(100)
    Customer = Customer()

while True:
    print("""
    ====== Car Rental Shop ======
    1. Display available Car
    2. Request a car on hourly basis $5/hour
    3. Request a car on daily basis $20/day
    4. Request a car on weekly basis $60/week
```

```
    5. Return a car
    6. Exit
    """)

    choice = input("Enter your choice: ")

    try:
        choice = int(choice)
    except ValueError:
        print("Invalid input. Please enter a number between 1 and 6.")
        continue

    if choice == 1:
        rental_shop.display_cars()

    elif choice == 2:
        customer.rentalTime, customer.rentalDuration = rental_shop.
↪rent_hourly(customer.request_car(), customer.request_duration(), customer.
↪return_car())
        customer.rentalBasis = 1


    elif choice == 3:
        customer.rentalTime, customer.rentalDuration = rental_shop.
↪rental_daily(customer.request_car(), customer.request_duration(), customer.
↪return_car())
        customer.rentalBasis = 2


    elif choice == 4:
        customer.rentalTime, customer.rentalDuration = rental_shop.
↪rental_weekly(customer.request_car(), customer.request_duration(), customer.
↪return_car())
        customer.rentalBasis = 3


    elif choice == 5:
        request = customer.return_car()
        rental_shop.return_car(request)
        customer.rentalBasis, customer.rentalTime, customer.rentalDuration,␣
↪customer.car = 0, None, 0, 0


    elif choice == 6:
        print('Okay, Goodbye! Visit Again')
        break
```

```
    else:
        print("Invalid input. Please enter a number between 1 and 6.")
```

[1]:
```
if __name__ == "_main_":
    main()
```