

```

1  #Hlanhla Hlungwane
2  #30 April 2025
3  #Define a class for an Online Car Rental Platform
4  #Object Oriented Programming on Python
5  #Performed on Jupyter Notebook
6
7  import datetime # Importing the built-in datetime module
8
9  class CarRental:
10     """
11     A class to manage car rental operations.
12     """
13
14     def __init__(self, inventory):
15         """
16         Constructor to initialize the car rental system.
17         :inventory: Number of cars available for rent.
18         """
19         self.inventory = inventory # Total number of cars available
20         self.rental_time = None    # Stores the time when cars are rented
21         self.rental_mode = None    # Stores the rental mode (hourly, daily, weekly)
22         self.rented_cars = 0       # Stores the number of cars rented
23
24     def display_available_cars(self):
25         """
26         Displays the number of available cars in the inventory.
27         """
28         print(f"Available cars for rent: {self.inventory}")
29
30     def rent_car_hourly(self, num_of_cars):
31         """
32         Rents cars on an hourly basis.
33         :num_of_cars: Number of cars requested for rent.
34         """
35         if num_of_cars <= 0:
36             print("Number of cars must be more than zero.")
37             return None
38         elif num_of_cars > self.inventory:
39             print("Not enough cars available for hourly rental.")
40             return None
41         else:
42             self.rental_time = datetime.datetime.now() # Store the current time
43             self.rental_mode = "hourly"               # Set rental mode
44             self.rented_cars = num_of_cars             # Store the number of rented cars
45             self.inventory -= num_of_cars              # Update inventory
46             print(f"{num_of_cars} car(s) rented on an hourly basis at {self.rental_time}.")
47             return self.rental_time
48
49     def rent_car_daily(self, num_of_cars):
50         """
51         Rents cars on a daily basis.
52         :num_of_cars: Number of cars requested for rent.
53         """
54         if num_of_cars <= 0:
55             print("Number of cars must be greater than zero.")
56             return None
57         elif num_of_cars > self.inventory:
58             print("Not enough cars available for daily rental.")
59             return None
60         else:
61             self.rental_time = datetime.datetime.now() # Store the current time
62             self.rental_mode = "daily"                 # Set rental mode
63             self.rented_cars = num_of_cars             # Store the number of rented cars
64             self.inventory -= num_of_cars              # Update inventory
65             print(f"{num_of_cars} car(s) rented on a daily basis at {self.rental_time}.")
66             return self.rental_time

```

```

68 def rent_car_weekly(self, num_of_cars):
69     """
70     Rents cars on a weekly basis.
71     :num_of_cars: Number of cars requested for rent.
72     """
73     if num_of_cars <= 0:
74         print("Number of cars must be greater than zero.")
75         return None
76     elif num_of_cars > self.inventory:
77         print("Not enough cars available for weekly rental.")
78         return None
79     else:
80         self.rental_time = datetime.datetime.now() # Store the current time
81         self.rental_mode = "weekly" # Set rental mode
82         self.rented_cars = num_of_cars # Store the number of rented cars
83         self.inventory -= num_of_cars # Update inventory
84         print(f"SUCCESS!!! {num_of_cars} car(s) rented on a weekly basis at {self.rental_time}.")
85         return self.rental_time
86
87 def return_car(self):
88     """
89     Returns the rented cars, calculates the rental period, and generates the final bill.
90     """
91     if self.rental_time is None:
92         print("No cars have been rented.")
93         return None
94
95     # Calculate the rental period
96     return_time = datetime.datetime.now()
97     rental_period = return_time - self.rental_time
98
99     # Determine cost based on rental mode
100    if self.rental_mode == "hourly":
101        bill = rental_period.seconds / 3600 * 250 * self.rented_cars # R250 per hour per car
102    elif self.rental_mode == "daily":
103        bill = rental_period.days * 1200 * self.rented_cars # R1200 per day per car
104    elif self.rental_mode == "weekly":
105        bill = (rental_period.days / 7) * 4500 * self.rented_cars # R4500 per week per car
106    else:
107        print("Invalid Selection")
108        return None
109
110    # Update inventory
111    self.inventory += self.rented_cars
112    self.rental_time = None
113    self.rental_mode = None
114    self.rented_cars = 0
115
116    print(f"Cars returned successfully. Total bill: R{bill:.2f}")
117    return bill
118
119 class Customer:
120     """
121     A class to represent customers interacting with the car rental system.
122     """
123
124     def __init__(self):
125         """
126         Constructor to initialize customer details.
127         """
128         self.rented_cars = 0 # Tracks the number of cars rented by the customer
129
130     def request_car(self):
131         """
132         Allows the customer to request cars for rent.
133         :return: Number of cars requested.
134         """

```

```
135     try:
136         num_of_cars = int(input("How many cars do you want to rent? "))
137         if num_of_cars <= 0:
138             print("Number of cars must be more than zero.")
139             return None
140         else:
141             self.rented_cars = num_of_cars
142             return num_of_cars
143     except ValueError:
144         print("Invalid input. Please enter a positive number")
145         return None
146
147 def return_car(self):
148     """
149     Allows the customer to return rented cars.
150     :return: Confirmation that cars are being returned.
151     """
152     if self.rented_cars == 0:
153         print("You haven't rented any car")
154         return None
155     else:
156         print(f"You have returned {self.rented_cars} car(s).")
157         return self.rented_cars
```