lab2-solution

February 1, 2024

1 TASK 1:

[1]:

```
class AutoCleaner:

    # initializing the values
    def __init__(self, dimensions):
        self.length, self.width, self.height = dimensions

    # function to calculate area
    def calculate_area(self):
        return self.length * self.width

    # function to calculate volume
    def calculate_volume(self):
        return self.length * self.width * self.height

    # display the area and volume
    def display(self):
        print(f"Area of the room: {self.calculate_area()}")
        print(f"Volume of the room: {self.calculate_volume()}")
        print("Cleaning Starts...")
```

[2]:

```
length = float(input("Enter the length of the room: "))
width = float(input("Enter the width of the room: "))
height = float(input("Enter the height of the room: "))

dimensions = (length, width, height)

cleaner = AutoCleaner(dimensions)

cleaner.display()
```

```
Area of the room: 25.0
Volume of the room: 250.0
Cleaning Starts…
```

2 TASK 2:

```
[ ]: class Fruit:
         def __init__(self):
             # Initialize sets for winter and summer season fruits
             self.winter_fruits = set()
             self.summer_fruits = set()

         def add_fruit(self, season, fruit):
             # Add a fruit to the specified season set
             if season.lower() == 'winter':
                 self.winter_fruits.add(fruit)
             elif season.lower() == 'summer':
                 self.summer_fruits.add(fruit)
             else:
                 print("Invalid season. Use 'winter' or 'summer'.")

         def remove_fruit(self, season, fruit):
             # Remove a fruit from given season
             if season.lower() == 'winter':
                 self.winter_fruits.discard(fruit)
             elif season.lower() == 'summer':
                 self.summer_fruits.discard(fruit)
             else:
                 print("Invalid season. Use 'winter' or 'summer'.")

         def is_fruit_present(self, season, fruit):
             # Check if a fruit is present in the specified season set
             if season.lower() == 'winter':  # make it lower case if user input some␣
       ↪mix string
                 return fruit in self.winter_fruits
             elif season.lower() == 'summer':
                 return fruit in self.summer_fruits
             else:
                 print("Invalid season. Use 'winter' or 'summer'.")
                 return False

         def display_sets(self):
             # Display sets
             print("Winter Fruits:", self.winter_fruits)
             print("Summer Fruits:", self.summer_fruits)

         def perform_set_operations(self):
             # set operations
             intersection_result = self.winter_fruits.intersection(self.
       ↪summer_fruits)
             union_result = self.winter_fruits.union(self.summer_fruits)
```

```
        difference_result = self.winter_fruits.difference(self.summer_fruits)

        print("Intersection of Winter and Summer Fruits:", intersection_result)
        print("Union of Winter and Summer Fruits:", union_result)
        print("Difference (Winter - Summer) Fruits:", difference_result)
```

[26]:

```
fruit_instance = Fruit()

# added some fruits
fruit_instance.add_fruit("winter", "orange")
fruit_instance.add_fruit("summer", "banana")
fruit_instance.display_sets()

# user input to add fruits
season_input = input("Enter the season ('winter' or 'summer'): ")
fruit_input = input("Enter the fruit to add: ")
fruit_instance.add_fruit(season_input, fruit_input)

# Display sets
fruit_instance.display_sets()

# user input to check if a fruit is present
check_season_input = input("Enter the season to check ('winter' or 'summer'): ")
check_fruit_input = input("Enter the fruit to check: ")
print(f"Fruit Present: {fruit_instance.is_fruit_present(check_season_input, check_fruit_input)}")

# user input to remove a fruit
remove_season_input = input("Enter the season to remove from ('winter' or 'summer'): ")
remove_fruit_input = input("Enter the fruit to remove: ")
fruit_instance.remove_fruit(remove_season_input, remove_fruit_input)

# Display sets after removal
fruit_instance.display_sets()

# set operations
fruit_instance.perform_set_operations()
```

```
Winter Fruits: {'orange'}
Summer Fruits: {'banana'}
Enter the season ('winter' or 'summer'): winter
Enter the fruit to add: banana
Winter Fruits: {'orange', 'banana'}
Summer Fruits: {'banana'}
Enter the season to check ('winter' or 'summer'): winter
Enter the fruit to check: banana
```

```
Fruit Present: True
Enter the season to remove from ('winter' or 'summer'): wnter
Enter the fruit to remove: banana
Invalid season. Use 'winter' or 'summer'.
Winter Fruits: {'orange', 'banana'}
Summer Fruits: {'banana'}
Intersection of Winter and Summer Fruits: {'banana'}
Union of Winter and Summer Fruits: {'orange', 'banana'}
Difference (Winter - Summer) Fruits: {'orange'}
```

3 TASK 3:

[3]:

```python
class MalariaDetectionSystem:
    def __init__(self):
        # Dictionary of symptoms and their weights
        self.symptoms_weights = {
            'fever': 4,
            'chills': 3,
            'sweating': 2,
            'headache': 4,
            'nausea': 3,
            'muscle aches': 5,
            'fatigue': 3,
            'joint pain': 2
        }

    def get_user_input(self):
        #input for symptom
        user_symptoms = {}
        for symptom, weight in self.symptoms_weights.items():
            response = input(f"Do you have {symptom}? (yes/no): ")
            if response.lower() == 'yes':
                user_symptoms[symptom] = 1
            else:
                user_symptoms[symptom] = 0
        return user_symptoms

    def calculate_total_score(self, user_symptoms):
        total_score = 0
        for symptom, weight in self.symptoms_weights.items():
            total_score += user_symptoms[symptom] * weight
        return total_score

    def check_malaria_risk(self, total_score):
        # Check if user has higher risk of Malaria
        threshold = 15
        if total_score > threshold:
            print("\nBased on your symptoms, there is a possibility that you␣
↪may have Malaria.")
            print("It is important to consult with a healthcare professional␣
↪for further guidance.")
        else:
            print("\nBased on your symptoms, it is less likely that you have␣
↪Malaria.")
            print("However, if you have concerns, it is always a good idea to␣
↪consult with a healthcare professional.")

    def run_detection_system(self):
        # Run the Malaria Detection System
        print("**Welcome to the Malaria Detection System!**\nPlease answer to␣
↪following questions.\n")

        #input for symptoms
        user_symptoms = self.get_user_input()

        #total score
        total_score = self.calculate_total_score(user_symptoms)

        #Check Malaria risk
        self.check_malaria_risk(total_score)
```

[4]:

```
# Create an instance of the MalariaDetectionSystem class
malaria_system = MalariaDetectionSystem()

# Run the Malaria Detection System
malaria_system.run_detection_system()
```

```
**Welcome to the Malaria Detection System!**
Please answer to following questions.

Do you have fever? (yes/no):  yes
Do you have chills? (yes/no):  yes
Do you have sweating? (yes/no):  no
Do you have headache? (yes/no):  yes
Do you have nausea? (yes/no):  yes
Do you have muscle aches? (yes/no):  no
Do you have fatigue? (yes/no):  yes
Do you have joint pain? (yes/no):  yes

Based on your symptoms, there is a possibility that you may have Malaria.
It is important to consult with a healthcare professional for further guidance.
```

4 TASK 4

[22]:

```
import random
from datetime import datetime, timedelta

class CarReservationSystem:
    def __init__(self):
        self.users = {} # dictionary to store the user information
        # dictionary to store the cars information
        self.cars = {
            'sports': {'name': 'Ferrari'},
            'luxury': {'name': 'Mercedes-Benz S-Class'},
            'electric': {'name': 'tesla model S'},
            'micro': {'name': 'smart fortwo'}
        }


    def add_user(self):
        # function to add new user details
        name = input("Enter your name: ")
        cnic = input("Enter your CNIC: ")
        phone = input("Enter your phone number: ")

        return {'name': name, 'cnic': cnic, 'phone': phone}


    def check_user(self, cnic):
        # function to check if user already exist or not
        return cnic in self.users



    def displayCarInfo(self, selectedCar):
        carInfo = self.cars.get(selectedCar)
        if carInfo: # checking if the car type exist, then display the car info␣
 ↪else display a invalid message
            carName = carInfo['name']
            rentalTime = self.generate_random_time()

            print("\n**Congratulations on renting a car**\n")
            print(f"\nCar type: {selectedCar}")
            print(f"Car Name: {carName}")
            print(f"Rental Time: {rentalTime}")
        else:
            print("Invalid car type!")
    def generate_random_time(self):
        # function to generate a random time
        currentTime = datetime.now()
        randomDays = random.randint(1, 10)
        rentalTime = currentTime + timedelta(days=randomDays)
        return rentalTime.strftime("%Y-%m-%d %H:%M:%S") #specified a formate to␣
 ↪display date and time


    def main_menu(self):
        # made a simple interactive main menu for user
        print("Welcome to the car Rental Reservation System!")

        cnic = input("Enter your cnic to check if you are existing user: ")
        userExist = self.check_user(cnic)

        if userExist:
            print(f"\nHello {self.users[cnic]['name']}")
        else:
            print("\nPlease provide the following information: ")
            userInfo = self.add_user()
            self.users[cnic] = userInfo
            print("New user added!")

        print("\nCar types: ")
        for car in self.cars:
            print(f"{car}")

        selectedCar = input("\nEnter the car type you want to rent: ")
        self.displayCarInfo(selectedCar)
```

[24]:

```
carRental = CarReservationSystem()
carRental.main_menu()
```

```
Welcome to the car Rental Reservation System!
Enter your cnic to check if you are existing user:  3320199994444

Please provide the following information:
Enter your name:  waghib ahmad
Enter your CNIC:  3320199994444
Enter your phone number:  03133333322
New user added!

Car types:
sports
luxury
electric
```

```
micro

Enter the car type you want to rent:  sports

**Congratulations on renting a car**


Car type: sports
Car Name: Ferrari
Rental Time: 2024-02-11 21:29:28
```

```
[25]: carRental.main_menu()
```

```
Welcome to the car Rental Reservation System!
Enter your cnic to check if you are existing user:  3320199994444

Hello waghib ahmad

Car types:
sports
luxury
electric
micro

Enter the car type you want to rent:  luxury

**Congratulations on renting a car**


Car type: luxury
Car Name: Mercedes-Benz S-Class
Rental Time: 2024-02-06 21:29:50
```

```
[ ]:
```