SORULAR

1] Bir kütüphane yönetim sistemi sınıf tasarımı yapmanız gerekmektedir:

- **Kütüphane** (Library), sıfır veya daha fazla kitap (Book) ve üye (Member) içerebilir.
- Her kitap (Book), bir yazar (Author) ve bir ISBN numarası içerir.
- Bir üye (Member), ödünç aldığı kitapları (BorrowedBooks) takip edebilmelidir.
- Üyeler, aynı anda en fazla 5 kitap ödünç alabilir.
- Kütüphane olmadan kitap (Book) veya üye (Member) olamaz.

2] Bir **akıllı ev yönetim sistemi** tasarımı yapmanız gerekmektedir:

- Her bir ev (SmartHome), en az bir cihaz (Device) içermelidir.
- Cihazlar (Device), ışık (Light), klima (AC), ve güvenlik kamerası (SecurityCamera) gibi türlere ayrılır.
- Her cihaz (Device) açılıp kapatılabilir.
- Işıklar (Light), parlaklık seviyesi (Brightness) 0-100 arasında ayarlanabilir.
- Klima (AC), sıcaklık seviyesi (Temperature) 16-30 derece arasında ayarlanabilir.

3] Bir film arşiv sistemi sınıf tasarımı yapmanız gerekmektedir:

- Bir arşiv (MovieArchive), birden fazla film (Movie) içerebilir.
- Her film (Movie), bir isim (Title), tür (Genre), ve bir çıkış yılı (ReleaseYear) bilgisine sahip olmalıdır.
- Filmler türlerine göre filtrelenebilmelidir.
- Kullanıcılar (User), arşivden film ekleyip çıkarabilir ve izledikleri filmleri işaretleyebilir.
- Her bir kullanıcı (User), izlediği filmler listesini (WatchedMovies) görebilmelidir.

4] Akıllı Ev Sistemi sorusuna ait sınıfların kodlamasını gerçekleştiriniz.

- Device sınıfı, ortak özellikleri içermelidir (ör. açma/kapama).
- Işık (Light) ve klima (AC) sınıfları, kendi özelliklerini (ör. parlaklık ve sıcaklık ayarı) içermelidir.
- Kodlama, gerekli tüm metotları ve getter/setter'ları içermelidir.

Cevaplar

1) Kütüphane Yönetim Sistemi

```
class Book:
    def __init__(self, title, author, isbn):
        self.title = title
        self.author = author
        self.isbn = isbn

    def __str__(self):
        return f"Book: {self.title}, Author: {self.author}, ISBN: {self.isbn}"

class Member:
    def __init__(self, name):
        self.name = name
        self.borrowed_books = []

    def borrow_book(self, book):
        if len(self.borrowed_books) < 5:
            self.borrowed_books.append(book)
            print(f"{book.title} borrowed by {self.name}")
        else:
            print(f"{self.name} cannot borrow more than 5 books.")

    def return_book(self, book):
        if book in self.borrowed_books:
            self.borrowed_books.remove(book)
```

```
            print(f"{book.title} returned by {self.name}")
        else:
            print(f"{self.name} does not have {book.title}.")

    def __str__(self):
        return f"Member: {self.name}, Borrowed Books: {[book.title for book in self.borrowed_books]}"

class Library:
    def __init__(self):
        self.books = []
        self.members = []

    def add_book(self, book):
        self.books.append(book)

    def add_member(self, member):
        self.members.append(member)

    def __str__(self):
        return f"Library has {len(self.books)} books and {len(self.members)} members."

Örenkleme
library = Library()
book1 = Book("1984", "George Orwell", "123456")
book2 = Book("To Kill a Mockingbird", "Harper Lee", "654321")
library.add_book(book1)
library.add_book(book2)

member1 = Member("Alice")
library.add_member(member1)

member1.borrow_book(book1)
print(member1)
```
2) Akıllı Ev Yönetim Sistemi
```
class Device:
    def __init__(self, name):
        self.name = name
        self.is_on = False

    def turn_on(self):
        self.is_on = True
        print(f"{self.name} is now ON.")

    def turn_off(self):
        self.is_on = False
        print(f"{self.name} is now OFF.")

class Light(Device):
    def __init__(self, name, brightness=0):
        super().__init__(name)
        self.brightness = brightness
```

```
    def set_brightness(self, level):
        if 0 <= level <= 100:
            self.brightness = level
            print(f"{self.name} brightness set to {level}.")
        else:
            print("Brightness must be between 0 and 100.")

class AC(Device):
    def __init__(self, name, temperature=24):
        super().__init__(name)
        self.temperature = temperature

    def set_temperature(self, temp):
        if 16 <= temp <= 30:
            self.temperature = temp
            print(f"{self.name} temperature set to {temp}°C.")
        else:
            print("Temperature must be between 16°C and 30°C.")

class SmartHome:
    def __init__(self):
        self.devices = []

    def add_device(self, device):
        self.devices.append(device)

    def __str__(self):
        return f"SmartHome has {len(self.devices)} devices."

örnekleme
home = SmartHome()
light = Light("Living Room Light")
ac = AC("Bedroom AC")
home.add_device(light)
home.add_device(ac)

light.turn_on()
light.set_brightness(75)
ac.turn_on()
ac.set_temperature(22)
print(home)

3) Film Arşiv Sistemi
class Movie:
    def __init__(self, title, genre, release_year):
        self.title = title
        self.genre = genre
        self.release_year = release_year

    def __str__(self):
        return f"Movie: {self.title}, Genre: {self.genre}, Year: {self.release_year}"

class User:
```

```
    def __init__(self, name):
        self.name = name
        self.watched_movies = []

    def add_watched_movie(self, movie):
        if movie not in self.watched_movies:
            self.watched_movies.append(movie)
            print(f"{movie.title} marked as watched by {self.name}.")
        else:
            print(f"{self.name} has already watched {movie.title}.")

    def __str__(self):
        return f"User: {self.name}, Watched Movies: {[movie.title for movie in self.watched_movies]}"

class MovieArchive:
    def __init__(self):
        self.movies = []

    def add_movie(self, movie):
        self.movies.append(movie)
        print(f"Movie {movie.title} added to the archive.")

    def filter_by_genre(self, genre):
        return [movie for movie in self.movies if movie.genre == genre]

    def __str__(self):
        return f"Archive contains {len(self.movies)} movies."

örnekleme
archive = MovieArchive()
movie1 = Movie("Inception", "Sci-Fi", 2010)
movie2 = Movie("Titanic", "Romance", 1997)
archive.add_movie(movie1)
archive.add_movie(movie2)

user1 = User("Bob")
user1.add_watched_movie(movie1)
print(user1)
print(archive.filter_by_genre("Sci-Fi"))
```
---

4) Akıllı Ev Sistemi Sınıflarının Kodlaması
```
class Device:
    def __init__(self, name):
        self.name = name
        self.is_on = False

    def turn_on(self):
        self.is_on = True
        print(f"{self.name} turned ON.")

    def turn_off(self):
```

```
        self.is_on = False
        print(f"{self.name} turned OFF.")


class Light(Device):
    def __init__(self, name, brightness=0):
        super().__init__(name)
        self.brightness = brightness

    def set_brightness(self, level):
        if 0 <= level <= 100:
            self.brightness = level
            print(f"{self.name} brightness set to {level}.")
        else:
            print("Brightness must be between 0 and 100.")

class AC(Device):
    def __init__(self, name, temperature=24):
        super().__init__(name)
        self.temperature = temperature

    def set_temperature(self, temp):
        if 16 <= temp <= 30:
            self.temperature = temp
            print(f"{self.name} temperature set to {temp}°C.")
        else:
            print("Temperature must be between 16°C and 30°C.")

örnekleme
light = Light("Bedroom Light")
ac = AC("Living Room AC")

light.turn_on()
light.set_brightness(85)
ac.turn_on()
ac.set_temperature(20)
```