Kütüphane sistemi

Kaynak kod:

```
using System;
using System.Collections.Generic;
using System.Linq;

class Program
{
  class Kitap
  {
    public string Baslik { get; set; }
    public string Yazar { get; set; }
    public string Yayinci { get; set; }
    public bool Mevcut { get; set; } = true;
    public string Kiralayan { get; set; } = null;
    public int Gun { get; set; } = 0;

    public Kitap(string baslik, string yazar, string yayinci)
    {
      Baslik = baslik;
      Yazar = yazar;
      Yayinci = yayinci;
    }
  }

  static void Main(string[] args)
  {
    List<Kitap> kitaplar = new List<Kitap>();
```

```
while (true)
{
    Console.WriteLine("\nKütüphane Yönetim Sistemi");
    Console.WriteLine("1. Kitap Ekle");
    Console.WriteLine("2. Kitap Kirala");
    Console.WriteLine("3. Kitap İade Et");
    Console.WriteLine("4. Kitap Ara");
    Console.WriteLine("5. Raporlar");
    Console.WriteLine("0. İşlemi Bitir");
    Console.Write("Seçiminizi yapın: ");
    string choice = Console.ReadLine();

    if (choice == "0")
    {
        Console.WriteLine("Çıkış yapılıyor...");
        break;
    }

    switch (choice)
    {
        case "1":
            KitapEkle(kitaplar);
            break;
        case "2":
            Kirala(kitaplar);
            break;
        case "3":
```

```
            İade(kitaplar);
            break;
        case "4":
            KitapArama(kitaplar);
            break;
        case "5":
            Raporla(kitaplar);
            break;
        default:
        Console.WriteLine("Lütfen tekrar deneyin.");
            break;
      }
   }
}

static void KitapEkle(List<Kitap> kitaplar)
{
   Console.Write("Kitap adi: ");
   string baslik = Console.ReadLine();
   Console.Write("Yazar adi: ");
   string yazar = Console.ReadLine();
   Console.Write("Yayinci adi: ");
   string yayinci = Console.ReadLine();

   kitaplar.Add(new Kitap(baslik, yazar, yayinci));
   Console.WriteLine($"\"{baslik}\" adlı kitap eklendi.");
}
```

```
  static void Kirala(List<Kitap> kitaplar)
  {
    Console.Write("Kiralanacak kitap adi: ");
    string baslik = Console.ReadLine();
    Kitap kitap = kitaplar.FirstOrDefault(b => b.Baslik.Equals(baslik,
StringComparison.OrdinalIgnoreCase) && b.Mevcut);

    if (kitap != null)
    {
      Console.Write("Kiralayan: ");
      string kiralayan = Console.ReadLine();
      Console.Write("Kaç gün kiralanacak: ");
      int gun = int.Parse(Console.ReadLine());

      kitap.Mevcut = false;
      kitap.Kiralayan = kiralayan;
      kitap.Gun = gun;
      Console.WriteLine($"\"{kitap.Baslik}\" adli kitap {gun} günlüğüne kiralandi.
Ödenecek toplam tutar: {gun * 5}.");
    }
    else
    {
      Console.WriteLine($"\"{baslik}\" adli kitap şu anda uygun değil veya mevcut
değil.");
    }
  }

  static void İade(List<Kitap> kitaplar)
  {
```

```
    Console.Write("İade edilecek kitap adi: ");
    string baslik = Console.ReadLine();
    Kitap kitap = kitaplar.FirstOrDefault(b => b.Baslik.Equals(baslik,
StringComparison.OrdinalIgnoreCase) && !b.Mevcut);

    if (kitap != null)
    {
      Console.WriteLine($"\"{kitap.Baslik}\" adli kitap başariyla iade edildi. Ödenmesi
gereken toplam ücret: {kitap.Gun * 5}.");
      kitap.Mevcut = true;
      kitap.Kiralayan = null;
      kitap.Gun = 0;
    }
    else
    {
      Console.WriteLine($"\"{baslik}\" adli kitap iade edilemez çünkü kirada değil.");
    }
  }

  static void KitapArama(List<Kitap> kitaplar)
  {
    Console.Write("Aranacak kitap adi: ");
    string baslik = Console.ReadLine();
    List<Kitap> bulunanlar = kitaplar.Where(b => b.Baslik.Contains(baslik,
StringComparison.OrdinalIgnoreCase)).ToList();

    if (bulunanlar.Any())
    {
      Console.WriteLine("Bulunan kitaplar:");
```

```
        foreach (var kitap in bulunanlar)
        {
            Console.WriteLine($"- {kitap.Baslik} (Yazar: {kitap.Yazar}, Yayıncı: {kitap.Yayinci}, Durum: {(kitap.Mevcut ? "Mevcut" : $"Kirada, Kiracı: {kitap.Kiralayan}, Geri Getirme Süresi: {kitap.Gun} gün" )})");
        }
    }
    else
    {
        Console.WriteLine("Böyle bir kitap bulunamadi.");
    }
}

static void Raporla(List<Kitap> Kitaplar)
{
    Console.WriteLine("Raporlama Seçenekleri:");
    Console.WriteLine("1. Bir yazarin tüm kitaplarini ara");
    Console.WriteLine("2. Bir yayinin tüm kitaplarini ara");
    Console.WriteLine("3. Hangi kitaplarin kimde olduğunu listele");
    Console.WriteLine("4. Tüm kitaplari listele");
    Console.Write("Seçiminizi yapin: ");
    string rapor = Console.ReadLine();

    switch (rapor)
    {
        case "1":
            Console.Write("Yazar adi: ");
            string yazar = Console.ReadLine();
```

```
            var yazarkitap = Kitaplar.Where(b => b.Yazar.Equals(yazar,
StringComparison.OrdinalIgnoreCase)).ToList();
            if (yazarkitap.Any())
            {
               Console.WriteLine($"{yazar} adli yazarin kitaplari:");
               foreach (var kitap in yazarkitap)
               {
                  Console.WriteLine($"- {kitap.Baslik}");
               }
            }
            else
            {
               Console.WriteLine($"{yazar} adli 0yazara ait kitap bulunamadi.");
            }
            break;

         case "2":
            Console.Write("Yayinci adi: ");
            string yayinci = Console.ReadLine();
            var yayincikitap = Kitaplar.Where(b => b.Yayinci.Equals(yayinci,
StringComparison.OrdinalIgnoreCase)).ToList();
            if (yayincikitap.Any())
            {
               Console.WriteLine($"{yayinci} yayincisinin kitaplari:");
               foreach (var kitap in yayincikitap)
               {
                  Console.WriteLine($"- {kitap.Baslik}");
               }
            }
```

```
            else
            {
                Console.WriteLine($"{yayinci} adli yayinciya ait kitap bulunamadi.");
            }
            break;

        case "3":
            var kiralananlar = Kitaplar.Where(b => !b.Mevcut).ToList();
            if (kiralananlar.Any())
            {
                Console.WriteLine("Kirada olan kitaplar:");
                foreach (var kitap in kiralananlar)
                {
                    Console.WriteLine($"- {kitap.Baslik}, Kiralayan: {kitap.Kiralayan}, Geri
Getirme Süresi: {kitap.Gun} gün");
                }
            }
            else
            {
                Console.WriteLine("Kirada olan kitap yok.");
            }
            break;

        case "4":
            if (Kitaplar.Any())
            {
                Console.WriteLine("Tüm kitaplar:");
                foreach (var kitap in Kitaplar)
```

```
            {
                Console.WriteLine($"- {kitap.Baslik} (Yazar: {kitap.Yazar}, Yayinci: {kitap.Yayinci}, Durum: {(kitap.Mevcut ? "Mevcut" : $"Kirada, Kiraci: {kitap.Kiralayan}, Geri Getirme Süresi: {kitap.Gun} gün")})");
            }
        }
        else
        {
            Console.WriteLine("Kütüphanede hiçbir kitap yok.");
        }
        break;

        default:
        Console.WriteLine("Geçersiz seçim. Lütfen tekrar deneyin.");
        break;
    }
  }
}
```

Basit bir kütüphane sistemi ilk başta yapılacak işlem seçiliyor ve seçilen işleme göre kitap ekleniyor yada kitap kiralama ve iade işlemleri yapılıyor extra olarak kitapların durumu ve kitap varmı kontrol etmek için arama seçenekleri de var.

Emir yılmaz 221120241006