**FIRAT ÜNİVERSİTESİ**

**MÜHENDİSLİK FAKÜLTESİ**

**YAZILIM MÜHENDİSLİĞİ BÖLÜMÜ**

NESNE TABANLI PROGRAMLAMA DERSİ

1.PROJE RAPORU

* PROJE ADI: KÜTÜPHANE YÖNETİM SİSTEMİ

* RAPOR HAZIRLAMA TARİHİ: 14.11.2024

**HAZIRLAYAN:**

MEHMET AÇIKGÖZ

**İÇİNDEKİLER**

GİRİŞ 3

PROJE AMACI 3

PROJE GELİŞTİRME SÜRECİ 3

KAYNAK KODLAR 5

ÇIKTILAR – SONUÇ VE DEPERLENDİRME 34

DEĞERLENDİRME 54

# GİRİŞ

Günümüzde, bilgiye erişim çabasına verilen zaman git gide azalıyor. Bu da bazı bilgilere hemen ulaşma isteğine sebep oluyor. Kütüphaneler gibi birden fazla bilgi barındıran kuruluşlarda yönetim süreçleri, kitapların takibi, üyelerin yönetimi, ödünç alma ve iade işlemleri gibi birçok karmaşık işlevi içeren bilgi sistemlerinin kullanılması çoğalıyor. Bildiğiniz üzere bu tür işlemleri etkin ve verimli bir şekilde yönetebilmek, kütüphane personelinin ve kullanıcılarının karşılaştığı en büyük zorluklardan biridir.

Proje kapsamında oluşturulmasını istenen yazılımda, kitaplar ve üyelerle ilgili işlemleri etkili bir şekilde yönetebilmek için sınıflar ve bu sınıflara ait metotlar tasarlanmıştır. Bu sınıflar, kütüphane içerisindeki işlemlerin sistematik bir şekilde düzenlenmesini ve takip edilmesini kolaylaştırır.

Bu rapor projenin amacını, geliştirme sürecini, projede kullanılan kaynak kodları, çıktılar ve sonuçları en son olarak ise projenin değerlendirilmesini içermektedir.

# PROJE AMACI

Bu oluşturmamız istene proje, kütüphane yönetim süreçlerini dijital ortamda kolaylaştırmayı amaçlayan bir **Kütüphane Yönetim Sistemi** tasarımını konu almaktadır. Proje, kütüphanedeki kitapların ve üyelerin yönetimini sağlayan, ödünç alma ve iade işlemlerini gerçekleştiren, ayrıca kitap ve üye bilgileri üzerinde sorgulama yapabilen bir yazılımdır. Bu tür yazılımsal projeler, kütüphane yada kütüphane gibi karmaşık bilgi içeren kurumlarda işleyişin daha hızlı, güvenilir ve kullanıcı dostu hale getirilmesine katkı sağlar.

Bu sayede ortada oluşan karmaşıklığı minimum seviye indirgeyerek istenen bilgi arayışında kolaylık ve bilgiye ulaşmayı hızlandırma gibi kolaylıklar sağlar.

# PROJE GELİŞTİRME SÜRECİ

Projenin amacında verilen ihtiyaç gereğiyle oluşturulan yazılımsal süreç aşamaları:

* 1. **İhtiyaç Analizi**
  2. **Sistem Tasarımı**
  3. **Kodlama**
  4. **Test Aşaması**

1. **İhtiyaç Analizi:**

İlk aşamada, projenin hedefleri ve fonksiyonel gereksinimler belirlenmiştir. Kütüphane Yönetim Sistemi’nin temel işlevselliği, kullanıcıların kütüphanedeki kitapları yönetmesini, kitap ödünç alma ve iade etme işlemlerini gerçekleştirmesini ve üye bilgilerini takip etmesini sağlamaktır.

Belirlenen temel ihtiyaçlar şunlardır:

* Kitap Yönetimi: Kitap ekleme, silme, arama, ödünç alma, iade etme, mevcut kitap durumunu kontrol etme.
* Üye Yönetimi: Üye ekleme, silme, arama, bilgilerini güncelleme ve ödünç geçmişini görüntüleme.
* Sorgulama ve Raporlama: Kitaplar ve üyeler hakkında detaylı sorgulama ve raporlama.

Bu gereksinimler doğrultusunda, kullanılacak sınıflar ve işlevsel metotlar tasarlanmıştır. Bunlar Kitap, Uye, Kitaplik, Uyeler, OduncKitap, Main gibi sınıfların her biri belirli işlevlere sahiptir.

1. **Sistem Tasarımı:**

Sistem tasarımı, Kütüphane Yönetim Sistemi’nin nasıl çalışacağını ve hangi bileşenlerden oluşacağını belirlemek için yapılmıştır. Bu aşamada, nesne tabanlı programlamaya dayalı olarak sistemin sınıf yapıları oluşturulmuştur.

* **Ana Sınıflar:**
  + Kitap: Kitaplarla ilgili tüm bilgiler ve işlemler (başlık, yazar, ISBN, ödünç durumu) bu sınıfta yönetilir.
  + Uye: Üye bilgilerini tutar ve üyenin ödünç aldığı kitapları takip eder.
  + Kitaplik:Kitaplar üzerinde yapılan (kitap ekleme, arama, ödünç) işlemlerini yönetir.
  + Uyeler:Üye (üye ekleme, silme, arama, listeleme) işlemlerini yönetir.
  + OduncKitap:Ödünç alınan kitaplar için kitap, oduncAlmaTarihi, iadeTarihi gibi bilgileri tutar.

Ek olarak sitemde kullanıcının etkileşimde bulunacağı bir menü yapısı bulunmaktadır. Ana menüde, kullanıcılar “Üye Yönetimi”, “Kitap Yönetimi” ve “Çıkış” seçeneklerinden birini seçebilir. Bu menüler üzerinden ilgili sınıflara ait işlevler çağrılır.

Sistemde kullanıcı dostu bir arayüz sunarak, kullanıcının yapmak istediği işlemleri kolayca gerçekleştirmesini ve karışık içinde kalmasını önlemeyi sağlar.

1. **Kodlama (Geliştirme) Aşaması:**

Kodlama aşamasında, tasarlanan sınıflar ve metotlar, Java programlama dili kullanılarak geliştirilmiştir. Sistemde kullanılan işlevsel bölümlerin kodları sırasıyla:

* **Kitap Sınıfı**

Kitap sınıfı, kitaplarla ilgili temel bilgileri ve işlemleri yöneten, kitap başlığı, yazarı, ISBN numarası ve ödünç durumu gibi bilgileri içeren sınıftır.

class Kitap {  
 private String baslik; // Kitabın başlığı  
 private String yazar; // Kitabın yazarı  
 private String isbn; // Kitabın ISBN numarası  
 private boolean oduncDurumu; /\* Kitap ödünçte mi? Durumu belirten

boolean değişkeni \*/  
  
 /\* Constructor: Kitap sınıfının örneğini (objesini) oluşturduğumuzda bu

metot çalışır \*/

public Kitap(String baslik, String yazar, String isbn) {

this.baslik = baslik; /\* Kitabın başlığını parametre olarak alıp

sınıfın 'baslik' değişkenine atıyoruz \*/

this.yazar = yazar; /\* Kitabın yazarını parametre olarak alıp

sınıfın 'yazar' değişkenine atıyoruz \*/

this.isbn = isbn; /\* Kitabın ISBN numarasını parametre olarak

alıp sınıfın 'isbn' değişkenine atıyoruz \*/

this.oduncDurumu = false; /\* Başlangıçta kitap ödünçte değil, yani

'false' olarak ayarlanır \*/

}  
  
 /\* Getter metodları: Sınıfın özel değişkenlerine dışarıdan erişim

sağlamak için kullanılan metotlar \*/

public String getBaslik() {

return baslik; /\* 'baslik' değişkeninin değerini döndürür, yani

kitabın başlığını verir \*/

}  
  
 public String getYazar() {

return yazar; /\* 'yazar' değişkeninin değerini döndürür, yani

kitabın yazarını verir \*/

}  
  
 public String getIsbn() {

return isbn; /\* 'isbn' değişkeninin değerini döndürür, yani

kitabın ISBN numarasını verir \*/

}  
  
 public boolean isOduncDurumu() {

return oduncDurumu; /\* 'oduncDurumu' değişkeninin değerini

döndürür, yani kitabın ödünç alınıp

alınmadığını kontrol eder \*/

}  
  
 /\* Setter metodları: Sınıfın özel değişkenlerinin değerlerini dışarıdan

değiştirmek için kullanılan metotlar \*/  
  
 public void setOduncDurumu(boolean oduncDurumu) {

this.oduncDurumu = oduncDurumu; /\* 'oduncDurumu' değişkenine yeni

bir değer atar \*/

/\* Bu metot sayesinde, kitabın ödünç durumu (ödünç alındı mı)

güncellenebilir \*/

}

}

**Kitap Sınıfı Kod Açıklaması:**

* + 1. **Değişkenler (Fields):**

private String baslik; // Kitabın başlığı  
 private String yazar; // Kitabın yazarı  
 private String isbn; // Kitabın ISBN numarası  
 private boolean oduncDurumu; // Kitap ödünçte mi? Durumu belirten

boolean değişkeni

Bu satırlar, **kitap** sınıfında yer alan değişkenler (fields) ve sınıfın içinde tutlan özelliklerdir. Bu değişkenler:

* baslik: Kitabın adını tutar.
* yazar: Kitabın yazarının adını tutar.
* isbn: Kitabın benzersiz ISBN numarasını tutar.
* oduncDurumu: Kitabın ödünç alınıp alınmadığını belirten bir boolean değeri. Eğer kitap ödünç alındıysa true, alınmadıysa false olur.
  + 1. **Constructor (Yapıcı Metot):**

public Kitap(String baslik, String yazar, String isbn) {

this.baslik = baslik; /\* Kitabın başlığını parametre olarak alıp

sınıfın 'baslik' değişkenine atıyoruz \*/

this.yazar = yazar; /\* Kitabın yazarını parametre olarak alıp

sınıfın 'yazar' değişkenine atıyoruz \*/  
 this.isbn = isbn; /\* Kitabın ISBN numarasını parametre olarak

alıp sınıfın 'isbn' değişkenine atıyoruz \*/  
 this.oduncDurumu = false; /\* Başlangıçta kitap ödünçte değil, yani

'false' olarak ayarlanır \*/  
}

* Kitap sınıfının constructor olup, kitap nesnesi oluşturulurken kitap başlığı yazarı ve ISBN numarası gibi parametreler alınır.
* this.baslik, this.yazar, ve this.isbn gibi ifadeler, sınıfın içindeki ilgili değişkenlere parametre değerlerini atar. this.oduncDurumu = false ise kitabın ödünç durumu başlangıçta false olacak şekilde ayarlanır.
  + 1. **Getter Metodları**:

public String getBaslik() {  
 return baslik; /\* 'baslik' değişkeninin değerini döndürür, yani

kitabın başlığını verir \*/  
}

* Bu metodun amacı, kitabın ödünç durumunu değiştirmektir.
* setOduncDurumu metodu bir **boolean** parametre alır ve sınıf içindeki oduncDurumu değişkenini bu yeni değerle günceller. Örneğin, kitap ödünç alındığında bu metot çağrılarak oduncDurumu değeri true yapılabilir.
  + 1. **Kodun Genel Amacı:**

 Kitap sınıfı, her kitabı temsil eden bir nesne olarak tasarlanmıştır. Kitabın başlığı, yazarı, ISBN numarası ve ödünç durumu gibi bilgileri saklar.

 Bu sınıfın **getter** metodları sayesinde, başka sınıflardan bu bilgilere erişilebilir.

 **Setter** metoduyla, ödünç alınan bir kitabın durumu güncellenebilir.

 Kitaplar, Kitaplik sınıfı gibi diğer sınıflar tarafından yönetilecek ve ödünç alma işlemleri gerçekleştirilecektir.

* **Kitaplik Sınıfı**

Kitaplik sınıfı, kitapları saklayan ve yönetmeyi kolaylaştıran bir sınıftır.

// Gerekli kütüphaneyi içe aktarıyoruz  
 import java.util.ArrayList; // ArrayList sınıfını kullanabilmek için

gerekli  
  
 // Kitaplik adında bir sınıf oluşturuyoruz  
 class Kitaplik {  
  
 // Kitapları saklamak için bir ArrayList tanımlıyoruz  
 private ArrayList<Kitap> kitapListesi = new ArrayList<>();  
  
 // \*\*\* Kitap Ekleme Fonksiyonu \*\*\*

// Yeni bir kitap eklemek için kullanılır  
 public void kitapEkle(Kitap kitap) {

kitapListesi.add(kitap); // Verilen kitabı listeye ekler  
 }  
  
 // \*\*\* Kitap Silme Fonksiyonu \*\*\*

// ISBN numarasına göre kitap siler  
 public boolean kitapSil(String isbn) {

// Listedeki her bir kitabı kontrol eder  
 for (Kitap kitap : kitapListesi) {

// Eğer ISBN numarası eşleşiyorsa  
 if (kitap.getIsbn().equals(isbn)) {

kitapListesi.remove(kitap); // Kitabı listeden kaldır

return true; // Silme başarılıysa true döndür  
 }  
 }

return false; // Kitap bulunamazsa false döndür  
 }  
  
 // \*\*\* Kitap Ödünç Alma Fonksiyonu \*\*\*

// Kitabın ödünç alınmasını sağlar  
 public boolean kitapOduncAl(String isbn) {

for (Kitap kitap : kitapListesi) {  
 // Eğer ISBN numarası eşleşiyor ve kitap ödünç alınmamışsa  
 if (kitap.getIsbn().equals(isbn) && !kitap.isOduncDurumu()) {

kitap.setOduncDurumu(true); // Kitap ödünç olarak

işaretlenir  
 return true; // Başarıyla ödünç alındıysa true döndür  
 }  
 }  
 return false; // Kitap bulunamazsa veya ödünç alınmışsa false

döndür  
 }  
  
 // \*\*\* Kitap İade Etme Fonksiyonu \*\*\*

// Ödünç alınan kitabın iade edilmesini sağlar  
 public boolean kitapIadeEt(String isbn) {

for (Kitap kitap : kitapListesi) {  
 // Eğer ISBN eşleşiyor ve kitap ödünç alınmışsa  
 if (kitap.getIsbn().equals(isbn) && kitap.isOduncDurumu()) {

kitap.setOduncDurumu(false); // Kitap iade edildi olarak

işaretlenir  
 return true; // Başarıyla iade edildiyse true döndür  
 }  
 }  
 return false; // Kitap bulunamazsa veya ödünç alınmamışsa false

döndür  
 }  
  
 // \*\*\* Kitap Arama Fonksiyonu \*\*\*

// Belirtilen ISBN numarasına göre kitap arar  
 public Kitap kitapAra(String isbn) {

for (Kitap kitap : kitapListesi) {  
 // Eğer ISBN numarası eşleşiyorsa kitabı döndür  
 if (kitap.getIsbn().equals(isbn)) {

return kitap;  
 }  
 }  
 return null; // Kitap bulunamazsa null döndür  
 }  
  
 // \*\*\* Kitapları Listeleme Fonksiyonu \*\*\*

// Kütüphanedeki tüm kitapları listeler  
 public void kitapListesiGoster() {  
 // Eğer liste boşsa bir mesaj göster  
 if (kitapListesi.isEmpty()) {

System.*out*.println("Kütüphanede kitap bulunmamaktadır.");

} else {  
 // Kitapları listele

System.*out*.println("\nKitap Listesi:");  
 for (Kitap kitap : kitapListesi) {  
 // Her kitabın başlık, yazar ve ISBN bilgilerini yazdır

System.*out*.println("Başlık: " + kitap.getBaslik() + ", Yazar: " + kitap.getYazar() + ", ISBN: " + kitap.getIsbn());  
 }  
 }  
 }  
  
 // \*\*\* Mevcut Kitap Durumu Fonksiyonu \*\*\*

// Kitabın kütüphanede mevcut olup olmadığını kontrol eder  
 public void mevcutKitapDurumu(String isbn) {

Kitap kitap = kitapAra(isbn); // Kitabı ISBN ile arar

if (kitap != null) { // Kitap bulunursa

// Kitap ödünç alınmışsa veya kütüphanedeyse mesaj göster  
 if (kitap.isOduncDurumu()) {

System.*out*.println("Kitap ödünç alınmış.");

} else {

System.*out*.println("Kitap kütüphanede mevcut.");

}  
 } else {

System.*out*.println("Kitap bulunamadı."); // Kitap bulunamazsa

mesaj göster  
 }  
 }  
 }

### ****Kitaplık Sınıfının Tanımı****

Kitaplık sınıfı, kütüphanedeki kitapları saklamak ve yönetmek için kullanılan bir sınıftır. Bu sınıfın içinde yer alan **ArrayList** veri yapısı, kitapların dinamik bir şekilde saklanmasını sağlar. Kitapların eklendiği, silindiği, ödünç alındığı ve iade edildiği işlemler, kullanıcının ihtiyaçlarına göre yönetilir. Kitaplık sınıfı, şu temel özellikleri ve metodları içerir:

* **Kitap Listesi**: Kitapları saklamak için kullanılan bir **ArrayList<Kitap>** nesnesi.
* **Kitap Ekleme**: Yeni bir kitap eklemek için kullanılan bir metot.
* **Kitap Silme**: Belirli bir ISBN numarasına göre kitap silme işlemi.
* **Kitap Ödünç Alma**: Kullanıcıların kitapları ödünç alabilmesi için bir fonksiyon.
* **Kitap İade Etme**: Kullanıcıların ödünç aldıkları kitapları iade etmeleri için bir fonksiyon.
* **Kitap Arama**: ISBN numarasına göre kitap arama fonksiyonu.
* **Kitapları Listeleme**: Kütüphanedeki tüm kitapların başlık, yazar ve ISBN bilgileriyle listeleme fonksiyonu.
* **Kitap Durumu Kontrolü**: Kitapların ödünç alınıp alınmadığını kontrol eden fonksiyon.

**Kitaplik Sınıfı Kod Açıklaması:**

1. **Kitap Ekleme Fonksiyonu**

public void kitapEkle(Kitap kitap) {  
 kitapListesi.add(kitap); // Verilen kitabı listeye ekler  
}

 **Amacı**: Bu metot, dışarıdan gelen bir Kitap nesnesini kitaplar listesine ekler. Kitap başarıyla eklenmişse, kullanıcıya bir mesaj gösterilir.

 **İşleyişi**:

* kitapListesi.add(kitap) ifadesiyle, parametre olarak gelen Kitap nesnesi **kitapListesi** ArrayList'ine eklenir.
* Kitap başarıyla eklenirse, eklenen kitabın başlığı yazdırılır.

1. **Kitap Silme Fonksiyonu**

public boolean kitapSil(String isbn) {  
 for (Kitap kitap : kitapListesi) {  
 if (kitap.getIsbn().equals(isbn)) {  
 kitapListesi.remove(kitap); // Kitabı listeden kaldır  
 System.*out*.println("Kitap başarıyla silindi: " + kitap.getBaslik());  
 return true; // Silme başarılıysa true döndür  
 }  
 }  
 System.*out*.println("Kitap bulunamadı.");  
 return false; // Kitap bulunamazsa false döndür  
}

 **Amacı**: Bu metot, belirtilen ISBN numarasına göre bir kitabı kitap listesinde arar ve bulduğunda siler.

 **İşleyişi**:

* Kitap listesinde her bir kitabı kontrol ederiz.
* Eğer kitap, belirtilen ISBN numarasına sahipse, bu kitap listeden çıkarılır (kitapListesi.remove(kitap)).
* Silme işlemi başarılıysa true, aksi takdirde false döndürülür.

1. **Kitap Ödünç Alma Fonksiyonu**

public boolean kitapOduncAl(String isbn) {  
 for (Kitap kitap : kitapListesi) {  
 if (kitap.getIsbn().equals(isbn) && !kitap.isOduncDurumu()) {  
 kitap.setOduncDurumu(true); // Kitap ödünç olarak işaretlenir  
 System.*out*.println("Kitap başarıyla ödünç alındı: " + kitap.getBaslik());  
 return true; // Başarıyla ödünç alındıysa true döndür  
 }  
 }  
 System.*out*.println("Kitap ödünç alınamıyor. Ya kitap ödünçte ya da bulunamadı.");  
 return false; // Kitap ödünç alınamazsa false döndür  
}

 **Amacı**: Bu metot, belirtilen ISBN numarasına sahip kitabı ödünç alır.

 **İşleyişi**:

* Eğer kitap bulunursa ve ödünç alınmamışsa, kitap ödünç alınmış olarak işaretlenir (kitap.setOduncDurumu(true)).
* Eğer kitap ödünç alınmışsa veya bulunamazsa, işlem gerçekleştirilmez ve kullanıcıya bir hata mesajı gösterilir.

1. **Kitap İade Etme Fonksiyonu**

public boolean kitapIadeEt(String isbn) {  
 for (Kitap kitap : kitapListesi) {  
 if (kitap.getIsbn().equals(isbn) && kitap.isOduncDurumu()) {  
 kitap.setOduncDurumu(false); // Kitap iade edildi olarak işaretlenir  
 System.*out*.println("Kitap başarıyla iade edildi: " + kitap.getBaslik());  
 return true; // Başarıyla iade edildiyse true döndür  
 }  
 }  
 System.*out*.println("Kitap iade edilemiyor. Ya kitap ödünçte değil ya da bulunamadı.");  
 return false; // Kitap iade edilemezse false döndür  
}

 **Amacı**: Bu metot, ödünç alınmış bir kitabı iade eder.

 **İşleyişi**:

* Eğer kitap ödünç alınmışsa, ödünç durumu false olarak değiştirilir.
* Kitap iade edilemezse, kullanıcıya uygun bir mesaj gösterilir.

1. **Kitap Arama Fonksiyonu**

public Kitap kitapAra(String isbn) {  
 for (Kitap kitap : kitapListesi) {  
 if (kitap.getIsbn().equals(isbn)) {  
 return kitap;  
 }  
 }  
 return null; // Kitap bulunamazsa null döndür  
}

 **Amacı**: Bu metot, belirtilen ISBN numarasına göre kitap arar ve bulursa geri döndürür.

 **İşleyişi**:

* Kitaplar arasında ISBN numarasına göre arama yapılır.
* Kitap bulunursa döndürülür, bulunamazsa null döner.

1. **Kitapları Listeleme Fonksiyonu**

public void kitapListesiGoster() {  
 if (kitapListesi.isEmpty()) {  
 System.*out*.println("Kütüphanede kitap bulunmamaktadır.");  
 } else {  
 System.*out*.println("\nKitap Listesi:");  
 for (Kitap kitap : kitapListesi) {  
 System.*out*.println("Başlık: " + kitap.getBaslik() + ", Yazar: " + kitap.getYazar() + ", ISBN: " + kitap.getIsbn());  
 }  
 }  
}

 **Amacı**: Bu metot, kütüphanedeki tüm kitapları listeleyerek kullanıcıya gösterir.

 **İşleyişi**:

* Eğer kitap listesi boşsa, kullanıcıya "Kitap bulunmamaktadır." mesajı gösterilir.
* Kitap listesi boş değilse, tüm kitaplar başlık, yazar ve ISBN bilgileriyle sırasıyla listelenir.

1. **Mevcut Kitap Durumu Fonksiyonu**

public void mevcutKitapDurumu(String isbn) {  
 Kitap kitap = kitapAra(isbn);  
 if (kitap != null) {  
 if (kitap.isOduncDurumu()) {  
 System.*out*.println("Kitap ödünç alınmış.");  
 } else {  
 System.*out*.println("Kitap kütüphanede mevcut.");  
 }  
 } else {  
 System.*out*.println("Kitap bulunamadı.");  
 }  
}

 **Amacı**: Bu metot, belirtilen ISBN numarasına sahip kitabın ödünç alınıp alınmadığını kontrol eder ve buna göre kullanıcıya bilgi verir.

 **İşleyişi**:

* Kitap, ISBN ile aranır.
* Eğer kitap bulunursa, ödünç durumu kontrol edilir ve uygun mesaj gösterilir.

1. **Kodun Genel Amacı:**

Kitaplık sınıfı, kütüphane yönetimi için temel işlevselliği sağlayan bir sınıftır. Kitapları ekleme, silme, ödünç alma, iade etme, arama ve listeleme işlemleri, kullanıcıların kitaplarla etkileşimde bulunmasını sağlar. Kitaplık sınıfı, kütüphane sisteminin temel yapı taşı olarak, kullanıcıların kitaplarla ilgili işlemleri kolayca gerçekleştirmelerini sağlar.

* **Uye Sınıfı**

Uye sınıfı, kütüphane yönetim sistemi içerisinde her bir üyenin bilgilerini saklamak ve üyelerin kitaplarla ilgili işlemlerini yönetmek amacıyla tasarlanmıştır. Üye, kütüphanede kitap ödünç alabilir, iade edebilir ve ödünç geçmişini görüntüleyebilir. Bu sınıf, kullanıcı bilgilerini (isim, üye numarası) ve ödünç alınan kitapların bilgilerini yönetir.

Aşağıda Uye sınıfının detaylı açıklaması, kullanılan metodlar ve bu metodların işlevleri ele alınacaktır.

import java.time.LocalDate; // LocalDate sınıfını kullanarak tarihlerle

çalışabilmek için import edilir.  
 import java.util.ArrayList; // ArrayList sınıfını kullanarak dinamik liste

oluşturabilmek için import edilir.  
  
 class Uye { // Uye sınıfı, bir kütüphane üyesinin özelliklerini ve

davranışlarını temsil eder.  
  
 private String isim; // Üyenin ismi.

private int uyeNumarasi; // Üyenin benzersiz numarası.

private ArrayList<OduncKitap> oduncAlinanKitaplar; // Üyenin

ödünç aldığı kitaplar listesi.  
  
 // Constructor (Yapıcı Metot): Uye sınıfının nesnesi

oluşturulurken üye ismi ve numarası belirlenir.  
 public Uye(String isim, int uyeNumarasi) {

this.isim = isim; // Üye ismini parametreden alır.

this.uyeNumarasi = uyeNumarasi; // Üye numarasını

parametreden alır.

this.oduncAlinanKitaplar = new ArrayList<>(); // Ödünç

alınan kitaplar listesi başlatılır.  
 }  
  
 // Getter (Alıcı) Metotlar: Üyenin ismini ve numarasını almak için kullanılır.  
 public String getIsim() {

return isim; // Üyenin ismini döndürür.  
 }

public int getUyeNumarasi() {

return uyeNumarasi; // Üyenin numarasını döndürür.  
 }  
  
 // Kitap ödünç alma metodu: Üye bir kitap ödünç alır.  
 public void kitapOduncAl(Kitap kitap) {

LocalDate oduncAlmaTarihi = LocalDate.now(); // Geçerli

tarihi alır.  
 OduncKitap oduncKitap = new OduncKitap(kitap, oduncAlmaTarihi); // Kitap ve ödünç alma tarihi ile OduncKitap nesnesi

oluşturulur.

oduncAlinanKitaplar.add(oduncKitap); // Bu kitap ödünç

alınan kitaplar listesine eklenir.  
 }  
  
 // Kitap iade etme metodu: Üye ödünç aldığı bir kitabı iade

eder.  
 public void kitapIadeEt(String isbn, LocalDate iadeTarihi) {  
 // OduncKitap listesindeki her bir kitaba bakılır.  
 for (OduncKitap oduncKitap : oduncAlinanKitaplar) {

// Kitabın ISBN numarasına göre eşleşme yapılır.  
 if (oduncKitap.getKitap().getIsbn().equals(isbn)) {

oduncKitap.setIadeTarihi(iadeTarihi); // İade

tarihi güncellenir.  
 System.*out*.println("Kitap '" + oduncKitap.getKitap().getBaslik() + "' iade edildi."); // İade işlemi

hakkında kullanıcı bilgilendirilir.

return; // Fonksiyon sonlandırılır.  
 }  
 }  
 System.*out*.println("Kitap bulunamadı!"); // Eğer kitap

bulunamazsa hata mesajı gösterilir.  
 }  
  
 // Ödünç geçmişini gösterme metodu: Üyenin ödünç aldığı

kitapların geçmişini listeler.  
 public void oduncGeçmisiGoster() {  
 // Eğer üye ödünç kitap almamışsa, bilgilendirme mesajı

gösterilir.  
 if (oduncAlinanKitaplar.isEmpty()) {

System.*out*.println(isim + " adlı üyenin ödünç geçmişi bulunmamaktadır.");

} else {  
 // Aksi takdirde, üyenin ödünç aldığı kitaplar yazdırılır.  
 System.*out*.println(isim + " adlı üyenin ödünç aldığı kitaplar:");

for (OduncKitap oduncKitap : oduncAlinanKitaplar) {  
 // Kitap başlığı, yazarı, ödünç alınma tarihi ve

iade tarihi bilgileri gösterilir.

System.*out*.println("Başlık: " + oduncKitap.getKitap().getBaslik() +  
 ", Yazar: " + oduncKitap.getKitap().getYazar() +  
 ", Ödünç Alınma Tarihi: " + oduncKitap.getOduncAlmaTarihi() +  
 ", İade Tarihi: " + (oduncKitap.getIadeTarihi() != null ? oduncKitap.getIadeTarihi() : "Henüz iade edilmedi"));  
 }  
 }  
 }  
  
 public void setIsim(String isim) {

this.isim = isim; // Üyenin ismini günceller.  
 }  
  
 // Bu metot şu an aktif değil, bir kitap silme işlemi

yapmamaktadır.  
 public boolean oduncKitapSil(String iadeIsbn) {

return false; // Şu an bu işlem yapılmıyor.  
 }  
 }

* 1. **Sınıfın Özellikleri (Fields)**

 **isim** (String): Üyenin adı.

 **uyeNumarasi** (int): Üyenin sistemdeki benzersiz numarası.

 **oduncAlinanKitaplar** (ArrayList<OduncKitap>): Bu liste, üyenin ödünç aldığı kitapların bilgilerini saklar. Her bir kitap, OduncKitap nesnesi olarak saklanır. Bu nesnede kitap bilgileri ile ödünç alınma tarihi ve iade tarihi yer alır.

**2. Yapıcı Metot (Constructor)**

public Uye(String isim, int uyeNumarasi) {  
 this.isim = isim;  
 this.uyeNumarasi = uyeNumarasi;  
 this.oduncAlinanKitaplar = new ArrayList<>();  
}

* **Amacı**: Bu metot, Uye sınıfının bir nesnesini oluştururken üye ismi, üye numarası ve ödünç alınan kitaplar listesini başlatır.
* **İşleyişi**:
  + this.isim = isim: Üye ismi parametre olarak alınıp, sınıfın isim değişkenine atanır.
  + this.uyeNumarasi = uyeNumarasi: Üye numarası parametre olarak alınır ve sınıfın uyeNumarasi değişkenine atanır.
  + this.oduncAlinanKitaplar = new ArrayList<>(): OduncKitap nesnelerini saklayacak olan liste, boş bir ArrayList ile başlatılır.

**3.Getter ve Setter Metodları**

Uye sınıfında, üyenin ismi ve üye numarası gibi bilgilere erişim sağlamak amacıyla getter metodları bulunur. Ayrıca, üye isminin değiştirilmesi için bir setter metodu da vardır.

* **getIsim()**: Bu metod, üyenin ismini döndürür.

public String getIsim() {  
 return isim;  
}

* **getUyeNumarasi()**: Bu metod, üyenin numarasını döndürür.

public int getUyeNumarasi() {  
 return uyeNumarasi;  
}

* **setIsim()**: Bu metod, üyenin ismini değiştirmeye yarar.

public void setIsim(String isim) {  
 this.isim = isim;  
}

**4. Kitap Ödünç Alma Metodu**

public void kitapOduncAl(Kitap kitap) {  
 LocalDate oduncAlmaTarihi = LocalDate.now(); // Geçerli tarihi al  
 OduncKitap oduncKitap = new OduncKitap(kitap, oduncAlmaTarihi); // Güncel tarihi geçerek OduncKitap nesnesini oluştur  
 oduncAlinanKitaplar.add(oduncKitap); // Kitap listesine ekle  
}

* **Amacı**: Bu metot, bir kitabın üye tarafından ödünç alınmasını sağlar.
* **İşleyişi**:
  + LocalDate.now() ile geçerli tarih alınır. Bu tarih, kitabın ödünç alındığı tarihi temsil eder.
  + OduncKitap nesnesi oluşturularak ödünç alınan kitap ve tarih bilgisi bu nesnede saklanır.
  + oduncAlinanKitaplar.add(oduncKitap) ile bu OduncKitap nesnesi üyenin ödünç aldığı kitaplar listesine eklenir.

### Kitap İade Etme Metodu

public void kitapIadeEt(String isbn, LocalDate iadeTarihi) {  
 for (OduncKitap oduncKitap : oduncAlinanKitaplar) {  
 if (oduncKitap.getKitap().getIsbn().equals(isbn)) {  
 oduncKitap.setIadeTarihi(iadeTarihi); // İade tarihi güncelleniyor  
 System.*out*.println("Kitap '" + oduncKitap.getKitap().getBaslik() + "' iade edildi.");  
 return;  
 }  
 }  
 System.*out*.println("Kitap bulunamadı!");  
}

 **Amacı**: Bu metot, bir kitabın üye tarafından iade edilmesini sağlar.

 **İşleyişi**:

* Kitaplar listesinde belirtilen ISBN numarasına sahip kitap aranır.
* Eğer kitap bulunursa, OduncKitap nesnesinin setIadeTarihi(iadeTarihi) metodu ile iade tarihi güncellenir.
* Kitap başarıyla iade edildiyse, kullanıcıya bir mesaj gösterilir.
* Eğer kitap bulunamazsa, kullanıcıya hata mesajı verilir.
  + 1. **Ödünç Geçmişini Gösterme Metodu**

public void oduncGeçmisiGoster() {  
 if (oduncAlinanKitaplar.isEmpty()) {  
 System.*out*.println(isim + " adlı üyenin ödünç geçmişi bulunmamaktadır.");  
 } else {  
 System.*out*.println(isim + " adlı üyenin ödünç aldığı kitaplar:");  
 for (OduncKitap oduncKitap : oduncAlinanKitaplar) {  
 System.*out*.println("Başlık: " + oduncKitap.getKitap().getBaslik() +  
 ", Yazar: " + oduncKitap.getKitap().getYazar() +  
 ", Ödünç Alınma Tarihi: " + oduncKitap.getOduncAlmaTarihi() +  
 ", İade Tarihi: " + (oduncKitap.getIadeTarihi() != null ? oduncKitap.getIadeTarihi() : "Henüz iade edilmedi"));  
 }  
 }  
}

 **Amacı**: Bu metot, üyenin geçmişte ödünç almış olduğu tüm kitapları ve bu kitapların ödünç alınma tarihlerini gösterir.

 **İşleyişi**:

* Eğer üye hiç kitap ödünç almamışsa, uygun bir mesaj gösterilir.
* Ödünç alınan kitaplar listelenir ve her bir kitabın başlığı, yazarı, ödünç alınma tarihi ve iade tarihi (eğer iade edilmişse) gösterilir.
  + 1. oduncKitapSil Metodu (Eksik)

public boolean oduncKitapSil(String iadeIsbn) {  
 return false;  
}

* **Amacı**: Bu metodun amacı, ödünç alınan bir kitabı silmektir. Ancak şu an için işlevsel değildir ve false döndürür.
* **Eksik İşlev**: Bu metodun çalışabilmesi için ödünç alınan kitaplar listesinde bir kitap silme işlemi gerçekleştirilmelidir. Ancak, mevcut sınıf tasarımında bu fonksiyonun nasıl işlevselleştirileceği belirtilmemiştir.
  + 1. **Kodun Genel Amacı:**

Uye sınıfı, kütüphane yönetim sisteminde her bir üyenin kitaplarla ilgili işlemlerini yönetmek için gerekli tüm fonksiyonları içerir. Üye bilgileri saklanır, kitaplar ödünç alınabilir, iade edilebilir ve ödünç geçmişi görüntülenebilir. Bu sınıf, OduncKitap sınıfı ile birlikte, kütüphanedeki kitapların ödünç alma ve iade işlemlerinin düzgün bir şekilde yönetilmesini sağlar.

* **Uyeler Sınıfı**

Uyeler sınıfı, kütüphane yönetim sisteminde bulunan tüm üyeleri yönetmek için tasarlanmış bir sınıftır. Bu sınıf, üyelerin eklenmesi, silinmesi, listelemesi ve araması gibi işlemleri gerçekleştiren metodlar içerir. Üye yönetimi, kütüphane sisteminin en önemli unsurlarından biridir çünkü üyelerin kitap ödünç alma ve iade etme işlemleri doğrudan onlarla ilişkilidir. Bu sınıf sayesinde, sistemdeki tüm üyeler üzerinde işlemler kolaylıkla yapılabilir.

import java.util.ArrayList; // Java'nın koleksiyon sınıflarından

ArrayList'i kullanabilmek için gerekli olan import işlemi yapılır.

class Uyeler { // Uyeler sınıfı, üye ekleme, silme, listeleme ve arama

işlemlerini yapacak metodları içerir.  
  
 private ArrayList<Uye> uyeListesi = new ArrayList<>(); //

ArrayList<Uye> türünde bir liste tanımlanır. Bu liste, üye nesnelerini

tutacaktır.  
  
 // Üye ekleme metodu

public void uyeEkle(Uye uye) {  
 uyeListesi.add(uye); // Parametre olarak alınan 'uye' nesnesi,

'uyeListesi' ArrayList'ine eklenir.  
 }  
  
 // Üye silme metodu

public boolean uyeSil(int uyeNumarasi) {  
 for (Uye uye : uyeListesi) { // uyeListesi'ndeki her bir 'Uye'

nesnesi üzerinde döngü başlatılır.

if (uye.getUyeNumarasi() == uyeNumarasi) { // Eğer 'Uye'

nesnesinin numarası, verilen numara ile eşleşiyorsa:  
 uyeListesi.remove(uye); // Eşleşen üye, listeden silinir.

return true; // Silme işlemi başarılı olduğunda 'true'

döndürülür.  
 }  
 }  
 return false; // Eğer verilen numaraya sahip bir üye bulunamazsa

'false' döndürülür.  
 }  
  
 // Üye listeleme metodu

public void uyeListele() {

if (uyeListesi.isEmpty()) { // Eğer 'uyeListesi' boşsa:  
 System.*out*.println("Listede üye bulunmamaktadır.");

// Kullanıcıya üye olmadığını belirten mesaj yazdırılır.

} else { // Liste boş değilse:

System.*out*.println("Üye Listesi:"); // Başlık yazdırılır.

for (Uye uye : uyeListesi) { // Her bir üye üzerinde döngü

yapılır:  
 System.*out*.println("İsim: " + uye.getIsim() + " - Üye Numarası: " + uye.getUyeNumarasi());  
 // Üyenin ismi ve numarası ekrana yazdırılır.  
 }  
 }  
 }  
  
 // Üye arama metodu

public Uye uyeAra(int uyeNumarasi) {

for (Uye uye : uyeListesi) { // Her bir üye üzerinde döngü

yapılır:  
 if (uye.getUyeNumarasi() == uyeNumarasi) { // Eğer üyenin

numarası, parametre olarak girilen numaraya eşitse:

return uye; // Eşleşen üye döndürülür.  
 }  
 }  
 return null; // Eğer üye bulunamazsa, 'null' döndürülür.  
 }  
}

**1. Uyeler Sınıfının Tanımı**

Uyeler sınıfı, kütüphaneye ait üyelerin yönetilmesini sağlamak amacıyla tasarlanmıştır. Bu sınıf, üyeleri saklamak için bir **ArrayList** kullanır. ArrayList, üyelerin listelenmesi, eklenmesi, silinmesi ve aranması işlemlerini etkili bir şekilde yönetir.

Uyeler sınıfının içeriğinde, aşağıdaki özellikler ve metodlar bulunmaktadır:

* **uyeListesi**: Bu, kütüphanedeki tüm üyeleri saklamak için kullanılan ArrayList<Uye> türünde bir koleksiyondur. Her Uye nesnesi, bir üyeyi temsil eder.
* **uyeEkle**: Yeni bir üye eklemeyi sağlar.
* **uyeSil**: Üye numarasına göre bir üyeyi siler.
* **uyeListele**: Tüm üyeleri listelemeyi sağlar.
* **uyeAra**: Üye numarasına göre bir üyeyi arar ve bulursa döndürür.

**2. Sınıfın Özellikleri (Fields)**

* **uyeListesi** (ArrayList<Uye>): Bu özellik, kütüphanedeki tüm üyeleri saklamak için bir liste oluşturur. Her Uye nesnesi bu listede bir eleman olarak bulunur. Bu liste sayesinde üyeler üzerinde ekleme, silme, listeleme ve arama işlemleri yapılabilir.

**3. Yapıcı Metot (Constructor)**

Uyeler sınıfı, parametre almayan bir yapıcı metoda sahiptir. Yapıcı metot, üye listesi (uyeListesi) için bir **ArrayList** nesnesi oluşturur. Bu liste, üye ekleme, silme ve arama işlemleri için kullanılacaktır. Yapıcı metot, sınıfın bir nesnesi oluşturulduğunda otomatik olarak çalışır.

public Uyeler() {  
 uyeListesi = new ArrayList<>(); // Yeni bir ArrayList oluşturuluyor  
}

**İşleyişi**: Yapıcı metot, uyeListesi özelliğini başlatır, böylece üye ekleme, silme ve arama işlemleri yapılabilir.

**4. Üye Ekleme (uyeEkle) Metodu**

public void uyeEkle(Uye uye) {  
 uyeListesi.add(uye); // Yeni bir üye listeye ekleniyor  
}

 **Amacı**: Bu metodun amacı, sisteme yeni bir üye eklemektir.

 **İşleyişi**:

* Metod, parametre olarak bir Uye nesnesi alır.
* Alınan Uye nesnesi, uyeListesi adlı listeye eklenir.
* Bu işlemle, kütüphaneye yeni bir üye dahil edilmiş olur.

5. Üye Silme (uyeSil) Metodu

public boolean uyeSil(int uyeNumarasi) {  
 for (Uye uye : uyeListesi) {  
 if (uye.getUyeNumarasi() == uyeNumarasi) {  
 uyeListesi.remove(uye); // Üye listeden siliniyor  
 return true; // Silme işlemi başarılı  
 }  
 }  
 return false; // Üye bulunamadı  
}

 **Amacı**: Bu metodun amacı, belirli bir üye numarasına sahip üyeyi sistemden silmektir.

 **İşleyişi**:

* uyeSil metodu, parametre olarak bir üye numarası alır.
* uyeListesi içindeki tüm üyeler tek tek kontrol edilir.
* Eğer listede verilen üye numarasına sahip bir üye bulunursa, bu üye listeden silinir ve true döndürülür.
* Eğer böyle bir üye bulunmazsa, false döndürülür.
* remove() metodu, listeye eklenen üyeleri çıkaran bir metoddur.

**6. Üye Listeleme (uyeListele) Metodu**

public void uyeListele() {  
 if (uyeListesi.isEmpty()) {  
 System.*out*.println("Listede üye bulunmamaktadır.");  
 } else {  
 System.*out*.println("Üye Listesi:");  
 for (Uye uye : uyeListesi) {  
 System.*out*.println("İsim: " + uye.getIsim() + " - Üye Numarası: " + uye.getUyeNumarasi());  
 }  
 }  
}

 **Amacı**: Bu metodun amacı, sistemdeki tüm üyeleri listelemektir.

 **İşleyişi**:

* İlk olarak, uyeListesi kontrol edilir. Eğer liste boşsa, "Listede üye bulunmamaktadır." mesajı gösterilir.
* Eğer liste boş değilse, tüm üyelerin ismi ve üye numarası ekrana yazdırılır.
* Her bir üye için uye.getIsim() ve uye.getUyeNumarasi() metodları çağrılarak üyenin bilgileri alınır ve ekrana yazdırılır.

**7. Üye Arama (uyeAra) Metodu**

public Uye uyeAra(int uyeNumarasi) {  
 for (Uye uye : uyeListesi) {  
 if (uye.getUyeNumarasi() == uyeNumarasi) {  
 return uye; // Aranan üye bulundu ve döndürüldü  
 }  
 }  
 return null; // Üye bulunamadı  
}

 **Amacı**: Bu metodun amacı, belirli bir üye numarasına sahip üyeyi aramaktır.

 **İşleyişi**:

* uyeAra metodu, parametre olarak bir üye numarası alır.
* uyeListesi içindeki her bir üye kontrol edilir.
* Eğer verilen numaraya sahip bir üye bulunursa, bu üye döndürülür.
* Eğer böyle bir üye bulunmazsa, null döndürülür.
* Bu metod, üye numarasına göre arama yapılmasını sağlar.

8. **Kodun Genel Amacı:**

Uyeler sınıfı, kütüphanedeki tüm üyeler üzerinde yapılması gereken temel işlemleri kapsar: üye ekleme, silme, listeleme ve arama. Bu sınıf, kütüphanenin üyelerini etkili bir şekilde yönetmek için gerekli olan tüm fonksiyonları sağlar. ArrayList kullanarak üyeleri saklamak, üyelerin hızlı bir şekilde eklenmesi, silinmesi ve aranması işlemlerini sağlar. Bu sınıf, kütüphane yönetim sisteminin üyelerle ilgili tüm işlevlerini yerine getiren önemli bir bileşendir.

* **OduncKitap Sınıfı**

OduncKitap sınıfı, bir kitabın ödünç alındığı tarih ve iade edildiği tarihi takip etmek için kullanılan bir sınıftır. Bu sınıf, kütüphane sistemlerinde kitap ödünç alma ve iade etme işlemlerini takip etmenin yanı sıra, ödünç alınan kitapların durumunu izlemek için gereklidir. OduncKitap sınıfı, her bir ödünç kitap işlemi için gerekli bilgileri saklar ve bu bilgileri sağlamak için getter ve setter metodları içerir.

import java.time.LocalDate; // Java'nın LocalDate sınıfını import eder. Bu

sınıf tarihleri temsil etmek için kullanılır.  
  
 class OduncKitap { // 'OduncKitap' sınıfı başlatılır, bu sınıf ödünç

alınan kitapları temsil eder.  
  
 private Kitap kitap; // 'Kitap' türünde bir değişken tanımlanır, ödünç

alınan kitabı temsil eder.

private LocalDate oduncAlmaTarihi; // Kitap ödünç alındığı tarih,

'LocalDate' türünde bir değişken.

private LocalDate iadeTarihi; // Kitap iade edildiği tarih, 'LocalDate'

türünde bir değişken.  
  
 // Constructor: OduncKitap nesnesi oluşturulurken kitap ve ödünç alma

tarihi parametre olarak alınır.

public OduncKitap(Kitap kitap, LocalDate oduncAlmaTarihi) {

this.kitap = kitap; // Constructor'da verilen kitap parametresi ile 'kitap' değişkeni atanır.

this.oduncAlmaTarihi = oduncAlmaTarihi; // 'oduncAlmaTarihi'

parametresi ile ödünç alma tarihi atanır.

this.iadeTarihi = null; // Başlangıçta 'iadeTarihi' null olarak

atanır çünkü kitap henüz iade edilmemiştir.  
 }  
  
 // Getter metodu: 'Kitap' türündeki kitabı döndürür.  
 public Kitap getKitap() {

return kitap; // Ödünç alınan kitabı geri döndürür.  
 }  
  
 // Getter metodu: Ödünç alınan kitabın tarihi olan 'oduncAlmaTarihi'ni

döndürür.  
 public LocalDate getOduncAlmaTarihi() {

return oduncAlmaTarihi; // Kitap ödünç alındığı tarihi geri

döndürür.  
 }

// Getter metodu: Kitap iade edilirse, 'iadeTarihi'ni döndürür.  
 public LocalDate getIadeTarihi() {

return iadeTarihi; // Kitap iade edilirse iade tarihi döndürülür,

yoksa null döner.  
 }  
  
 // Setter metodu: Kitap iade edildiğinde 'iadeTarihi'ni ayarlamak için

kullanılır.  
 public void setIadeTarihi(LocalDate iadeTarihi) {  
 this.iadeTarihi = iadeTarihi; // Kitap iade edildiğinde iade

tarihini belirler.  
 }  
}

**1. OduncKitap Sınıfının Tanımı**

OduncKitap sınıfı, kütüphaneden ödünç alınan bir kitabın takibini yapmak amacıyla tasarlanmış bir sınıftır. Bu sınıf, bir ödünç kitap işlemiyle ilgili iki temel bilgiyi saklar:

* **Ödünç alınan kitap**: Bu özellik, ödünç alınan kitabı temsil eden Kitap nesnesini tutar.
* **Ödünç alma tarihi**: Bu özellik, kitabın ödünç alındığı tarihi saklar.
* **İade tarihi**: Kitap geri iade edildiğinde, bu tarih tutulur.

Sınıfın amacı, ödünç alınan kitapların takibini yapabilmek ve ödünç kitap işlemleri ile ilgili verileri düzenli bir şekilde saklamaktır.

**2. Sınıfın Özellikleri (Fields)**

 **kitap** (Kitap): Bu özellik, ödünç alınan kitabı temsil eden Kitap nesnesini tutar. Kitap nesnesi, ödünç alınan kitabın başlık, yazar, ISBN numarası gibi temel bilgilerini içerir.

 **oduncAlmaTarihi** (LocalDate): Bu özellik, kitabın ödünç alındığı tarihi saklar. Bu tarih, ödünç alma işlemi gerçekleştirildiğinde belirlenir ve LocalDate.now( ) kullanılarak atanabilir.

 **iadeTarihi** (LocalDate): Bu özellik, kitabın iade edilme tarihini tutar. Başlangıçta null olarak ayarlanır, çünkü kitap henüz iade edilmemiştir. Kitap iade edildiğinde bu tarih belirlenir.

**3. Yapıcı Metot (Constructor)**

public OduncKitap(Kitap kitap, LocalDate oduncAlmaTarihi) {  
 this.kitap = kitap; // Kitap, parametre olarak alınır ve 'kitap' özelliğine atanır  
 this.oduncAlmaTarihi = oduncAlmaTarihi; // Ödünç alma tarihi parametre olarak alınır ve 'oduncAlmaTarihi' özelliğine atanır  
 this.iadeTarihi = null; // Başlangıçta iade tarihi null, çünkü kitap henüz iade edilmemiştir  
}

 **Amacı**: Yapıcı metot, OduncKitap nesnesi oluşturulurken ödünç alınan kitabı ve ödünç alma tarihini belirlemek için kullanılır.

 **İşleyişi**:

* Parametre olarak bir Kitap nesnesi ve bir LocalDate (ödünç alma tarihi) alınır.
* this.kitap ile ödünç alınan kitap nesnesi, sınıfın kitap özelliğine atanır.
* this.oduncAlmaTarihi ile ödünç alma tarihi sınıfın oduncAlmaTarihi özelliğine atanır.
* Başlangıçta iadeTarihi null olarak ayarlanır, çünkü kitap henüz iade edilmemiştir.

**4. Getter Metodları**

public Kitap getKitap() {  
 return kitap; // Ödünç alınan kitabı döndürür  
}

**Amacı**: Bu metod, kitap özelliğine erişmek için kullanılır. Ödünç alınan kitabın bilgilerini elde etmek için çağrılır.

public LocalDate getOduncAlmaTarihi() {  
 return oduncAlmaTarihi; // Kitap ödünç alınma tarihini döndürür  
}

**Amacı**: Bu metod, kitabın ödünç alınma tarihini almak için kullanılır. Kitap ödünç alındığında hangi tarihte alındığını öğrenmek için çağrılır.

public LocalDate getIadeTarihi() {  
 return iadeTarihi; // Kitap iade edilirse iade tarihini döndürür  
}

**Amacı**: Bu metod, kitabın iade edilip edilmediğini ve eğer edilmişse, hangi tarihte iade edildiğini öğrenmek için kullanılır. Kitap henüz iade edilmemişse, null döndürülecektir.

**5. Setter Metodu**

public void setIadeTarihi(LocalDate iadeTarihi) {  
 this.iadeTarihi = iadeTarihi; // Kitap iade edildiğinde tarihi ayarlar  
}

 **Amacı**: Kitap iade edildikten sonra, iade tarihini güncellemek için kullanılır.

 **İşleyişi**:

* Parametre olarak bir LocalDate (iade tarihi) alınır.
* this.iadeTarihi özelliği, verilen iade tarihiyle güncellenir.

6. **Kodun Genel Amacı:**

OduncKitap sınıfı, ödünç kitap işlemlerinin doğru bir şekilde izlenebilmesi için gerekli verileri saklar. Her bir ödünç alınan kitap için şu bilgiler sağlanır:

* **Kitap bilgisi**: Ödünç alınan kitabın başlık, yazar, ISBN numarası gibi temel bilgileri.
* **Ödünç alma tarihi**: Kitabın hangi tarihte ödünç alındığını gösteren tarih.
* **İade tarihi**: Kitabın geri verildiği tarih (kitap iade edilmemişse null olur).

Bu bilgiler, kütüphane yönetim sistemlerinde ödünç alma ve iade etme işlemlerinin takip edilmesi için kritik öneme sahiptir. Kullanıcılar, bir kitabı ödünç aldıklarında ve geri verdiklerinde, bu bilgiler güncellenerek sisteme kaydedilir.

OduncKitap sınıfı, ödünç kitap işlemlerini takip etmek için oldukça önemli bir sınıftır. Her ödünç kitap işlemi, bir OduncKitap nesnesi olarak sistemde kaydedilir ve ödünç alma tarihi ile iade tarihi gibi bilgilerle takip edilir. Bu sınıf, kütüphane yönetim sisteminde kitapların takibini ve kullanıcıların ödünç kitap işlemlerini yönetmek için kullanılır.

* **Main Sınıfı**

Main sınıfı, programın kullanıcı arayüzünü sağlamak amacıyla temel işlevlerin tümünü yönetir. Bu sınıf, kütüphane yönetim sisteminin başlangıç noktasıdır ve kullanıcıların kitaplar ve üyeler üzerinde gerçekleştirebileceği işlemleri sağlamak için çeşitli menüler ve etkileşimler sunar. Kullanıcı, bu sınıf aracılığıyla üye yönetimi ve kitap yönetimi işlemlerini gerçekleştirebilir.

**Ana sınıf ve değişkenler:**

import java.time.LocalDate;  
 import java.util.Scanner;  
  
 public class Main {  
 static Scanner *scanner* = new Scanner(System.*in*); // Kullanıcıdan giriş almak için Scanner nesnesi oluşturuluyor  
 static Kitaplik *kitaplik* = new Kitaplik(); // Kitaplık nesnesi

oluşturuluyor  
 static Uyeler *uyeler* = new Uyeler(); // Üyeler nesnesi oluşturuluyor

 Scanner metodu, kullanıcıdan veri almak için kullanılır.

 Kitaplik ve Uyeler, kitaplar ve üyelerle ilgili yönetimi sağlayan sınıfların nesneleridir.

**Ana metod (main):**

public static void main(String[] args) {  
 // Başlangıçta birkaç kitap ve üye ekleyelim  
 kitaplik.kitapEkle(new Kitap("Savaş ve Barış", "Lev Tolstoy", "24001"));  
 kitaplik.kitapEkle(new Kitap("İçimizdeki Şeytan", "Sabahattin Ali", "24002"));  
 kitaplik.kitapEkle(new Kitap("Yıldıza Dokunmak", "Sezin Karameşe", "24003"));  
 kitaplik.kitapEkle(new Kitap("Rezonans Kanunu", "Pierre Franckh", "24004"));  
 uyeler.uyeEkle(new Uye("Mehmet Açıkgöz", 1));  
 uyeler.uyeEkle(new Uye("Eda Kayıran", 2));  
 uyeler.uyeEkle(new Uye("Ayşe Kırgın", 3));  
 uyeler.uyeEkle(new Uye("Sümeyye Mart", 4));

 Program başladığında başlangıç olarak birkaç kitap ve üye ekleniyor.

 Kitap ve Uye nesneleri yaratılır ve kitaplık ve üyeler listesine eklenir.,

**Ana menü:**

while (true) {  
 System.out.println("Kullanıcı Yönetim Sistemine Hoşgeldiniz");  
 System.out.println("1. Üye Yönetimi");  
 System.out.println("2. Kitap Yönetimi");  
 System.out.println("3. Çıkış");  
 System.out.print("Bir seçenek girin: ");  
 int secim = scanner.nextInt();  
 scanner.nextLine(); // Buffer temizleme  
 switch (secim) {  
 case 1:  
 uyeYonetimi();  
 break;  
 case 2:  
 kitapYonetimi();  
 break;  
 case 3:  
 System.out.println("Çıkılıyor...");  
 return;  
 default:  
 System.out.println("Geçersiz seçenek. Tekrar deneyin.");  
 }  
 }  
 }

 Ana menü sürekli çalışır ve kullanıcıdan bir seçim alır.

 Seçime göre "Üye Yönetimi" veya "Kitap Yönetimi" menülerine yönlendirir.

 case 3 ile programdan çıkılır.

**Üye Yönetimi Menüsü:**

public static void uyeYonetimi() {  
 while (true) {  
 System.*out*.println("\nÜye Yönetimi");  
 System.*out*.println("1. Üye Ekle");  
 System.*out*.println("2. Üye Silme");  
 System.*out*.println("3. Üye Listele");  
 System.*out*.println("4. Üye Ara");  
 System.*out*.println("5. Üye Bilgilerini Güncelle");  
 System.*out*.println("6. Üye Ödünç Geçmişini Görüntüle");  
 System.*out*.println("7. Geri");  
 System.*out*.print("Bir seçenek girin: ");  
 int secim = scanner.nextInt();  
 scanner.nextLine(); // Buffer temizleme  
 switch (secim) {  
 case 1:  
 // Üye Ekleme  
 System.*out*.print("Üye İsmi: ");  
 String isim = scanner.nextLine();  
 System.*out*.print("Üye Numarası: ");  
 int uyeNumarasi = scanner.nextInt();  
 uyeler.uyeEkle(new Uye(isim, uyeNumarasi));  
 System.*out*.println("Üye başarıyla eklendi.");  
 break;  
 case 2:  
 // Üye Silme  
 System.*out*.print("Silmek istediğiniz üye numarasını girin: ");  
 int silinecekNumara = scanner.nextInt();  
 if (uyeler.uyeSil(silinecekNumara)) {  
 System.*out*.println("Üye başarıyla silindi.");  
 } else {  
 System.*out*.println("Üye bulunamadı.");  
 }  
 break;  
 case 3:  
 uyeler.uyeListele();  
 break;  
 case 4:  
 System.*out*.print("Aramak istediğiniz üye numarasını girin: ");  
 int aramaNumarasi = scanner.nextInt();  
 Uye uye = uyeler.uyeAra(aramaNumarasi);  
 if (uye != null) {  
 System.*out*.println("Üye bulundu: " + uye.getIsim());  
 } else {  
 System.*out*.println("Üye bulunamadı.");  
 }  
 break;  
 case 5:  
 // Üye bilgilerini güncelleme  
 System.*out*.print("Güncellemek istediğiniz üye numarasını girin: ");  
 uyeNumarasi = scanner.nextInt();  
 scanner.nextLine(); // Buffer temizleme  
 Uye uyeToUpdate = uyeler.uyeAra(uyeNumarasi);  
 if (uyeToUpdate != null) {  
 System.*out*.print("Yeni isim girin: ");  
 String yeniIsim = scanner.nextLine();  
 uyeToUpdate.setIsim(yeniIsim);  
 System.*out*.println("Üye ismi başarıyla güncellendi.");  
 } else {  
 System.*out*.println("Üye bulunamadı.");  
 }  
 break;  
 case 6:  
 // Ödünç geçmişi görüntüleme  
 System.*out*.print("Ödünç geçmişini görmek istediğiniz üye numarasını girin: ");  
 int numara = scanner.nextInt();  
 Uye uye1 = uyeler.uyeAra(numara);  
 if (uye1 != null) {  
 uye1.oduncGeçmisiGoster();  
 } else {  
 System.*out*.println("Üye bulunamadı.");  
 }  
 break;  
 case 7:  
 return;  
 default:  
 System.*out*.println("Geçersiz seçenek.");  
 }  
 }  
}

 Üye yönetim işlemleri için bir menü sağlar.

 Üye ekleme, silme, listeleme, arama, güncelleme ve ödünç geçmişini görüntüleme gibi işlemler yapılabilir.

**Kitap Yönetimi Menüsü:**

public static void kitapYonetimi() {  
 while (true) {  
 System.*out*.println("\nKitap Yönetimi");  
 System.*out*.println("1. Kitap Ekle");  
 System.*out*.println("2. Kitap Ödünç Al");  
 System.*out*.println("3. Kitap İade Et");  
 System.*out*.println("4. Kitap Arama");  
 System.*out*.println("5. Kitapları Listele");  
 System.*out*.println("6. Kitap Silme");  
 System.*out*.println("7. Mevcut Kitap Kontrolü");  
 System.*out*.println("8. Geri");  
 System.*out*.print("Bir seçenek girin: ");  
 int secim = scanner.nextInt();  
 scanner.nextLine(); // Buffer temizleme  
 switch (secim) {  
 case 1:  
 // Kitap Ekle  
 System.*out*.print("Kitap Başlığı: ");  
 String baslik = scanner.nextLine();  
 System.*out*.print("Yazar: ");  
 String yazar = scanner.nextLine();  
 System.*out*.print("ISBN: ");  
 String isbn = scanner.nextLine();  
 kitaplik.kitapEkle(new Kitap(baslik, yazar, isbn));  
 System.*out*.println("Kitap başarıyla eklendi.");  
 break;  
 case 2:  
 // Kitap Ödünç Al  
 System.*out*.print("Üye numaranızı girin: ");  
 int uyeNumarasi = scanner.nextInt();  
 Uye uye = uyeler.uyeAra(uyeNumarasi);  
 if (uye == null) {  
 System.*out*.println("Üye bulunamadı.");  
 break;  
 }  
 scanner.nextLine(); // Buffer temizleme  
 System.*out*.print("Ödünç almak istediğiniz kitabın ISBN'sini girin: ");  
 String oduncIsbn = scanner.nextLine();  
 Kitap kitap = kitaplik.kitapAra(oduncIsbn);  
 if (kitap != null && !kitap.isOduncDurumu()) {  
 if (kitaplik.kitapOduncAl(oduncIsbn)) {  
 uye.kitapOduncAl(kitap); // Kitap ödünç alındığında üye bilgilerine eklenir  
 System.*out*.println("Kitap ödünç alındı.");  
 } else {  
 System.*out*.println("Kitap şu an ödünç verilemiyor.");  
 }  
 } else {  
 System.*out*.println("Kitap bulunamadı veya zaten ödünçte.");  
 }  
 break;  
 case 3:  
 // Kitap İade Et  
 System.*out*.print("İade etmek istediğiniz kitabın ISBN'sini girin: ");  
 String iadeIsbn = scanner.nextLine();  
 Kitap kitap1 = kitaplik.kitapAra(iadeIsbn);  
 if (kitap1 != null) {  
 kitaplik.kitapIadeEt(kitap1);  
 System.*out*.println("Kitap başarıyla iade edildi.");  
 } else {  
 System.*out*.println("Kitap bulunamadı.");  
 }  
 break;  
 case 4:  
 // Kitap Arama  
 System.*out*.print("Aramak istediğiniz kitabın ISBN'sini girin: ");  
 String aramaIsbn = scanner.nextLine();  
 Kitap bulunanKitap = kitaplik.kitapAra(aramaIsbn);  
 if (bulunanKitap != null) {  
 System.*out*.println("Kitap bulundu: " + bulunanKitap);  
 } else {  
 System.*out*.println("Kitap bulunamadı.");  
 }  
 break;  
 case 5:  
 // Kitap Listeleme  
 kitaplik.kitapListele();  
 break;  
 case 6:  
 // Kitap Silme  
 System.*out*.print("Silmek istediğiniz kitabın ISBN'sini girin: ");  
 String silmeIsbn = scanner.nextLine();  
 kitaplik.kitapSil(silmeIsbn);  
 break;  
 case 7:  
 // Mevcut Kitap Kontrolü  
 System.*out*.print("Kitap ISBN'sini girin: ");  
 String kontrolIsbn = scanner.nextLine();  
 if (kitaplik.kitapAra(kontrolIsbn) != null) {  
 System.*out*.println("Kitap mevcut.");  
 } else {  
 System.*out*.println("Kitap bulunamadı.");  
 }  
 break;  
 case 8:  
 return;  
 default:  
 System.*out*.println("Geçersiz seçenek.");  
 }  
 }  
 }  
}

 Kitap yönetimi için seçenekler sunar.

 Kitap ekleme, ödünç alma, iade etme, arama, listeleme, silme ve mevcut kitap kontrolü gibi işlemler yapılabilir.

**1. Main Sınıfının Yapısı**

Main sınıfı, kütüphaneye ait kitaplar ve üyelerle etkileşimi yönetmek için bazı yardımcı sınıflara (Kitaplik, Uyeler, Kitap, Uye) başvurur. Bu sınıf, kullanıcıdan gelen girişlere göre belirli işlemleri yapabilmek için aşağıdaki bileşenleri içerir:

* **Scanner nesnesi**: Kullanıcının komutlarını almak için kullanılan Scanner nesnesi, program boyunca kullanıcıdan veri almak için kullanılır.
* **Kitaplik ve Uyeler nesneleri**: Kitap ve üye işlemleri için Kitaplik ve Uyeler sınıflarından nesneler oluşturulmuştur. Bu nesneler üzerinden kitap ve üye ekleme, silme, listeleme gibi işlemler yapılır.

**2. Ana Metot: main(String[] args)**

main metodu, programın giriş noktasıdır. Program başladığında ilk olarak bazı kitaplar ve üyeler eklenir. Ardından, kullanıcıdan gelen komutlara göre farklı menüler açılır. Bu metodun temel işlevleri şunlardır:

#### **Başlangıç Verisi**

* **Kitaplar**: Program başladığında birkaç kitap sisteme eklenir:
  + "Savaş ve Barış", "İçimizdeki Şeytan", "Yıldıza Dokunmak", "Rezonans Kanunu" kitapları, her birinin başlık, yazar ve ISBN bilgileriyle sisteme eklenir.
* **Üyeler**: Aynı şekilde birkaç üye sisteme eklenir:
  + "Mehmet Açıkgöz", "Eda Kayıran", "Ayşe Kırgın", "Sümeyye Mart" gibi üyeler, her biri için üye numarası atanarak sisteme eklenir.

#### **Kullanıcı Menü Seçenekleri**

Kullanıcıya şu ana menü seçenekleri sunulur:

1. **Üye Yönetimi**
2. **Kitap Yönetimi**
3. **Çıkış**

Kullanıcı bu seçeneklerden birini seçerek, üye yönetimi veya kitap yönetimi ile ilgili işlemleri gerçekleştirebilir.

**3. Üye Yönetimi Menüsü: uyeYonetimi()**

uyeYonetimi() metodu, üye yönetimi ile ilgili işlemleri gerçekleştirmek için kullanılan bir menüyü kontrol eder. Bu menüde kullanıcı, aşağıdaki işlemleri yapabilir:

#### **1. Üye Ekleme**

* Kullanıcıdan üye ismi ve numarası alınarak yeni bir üye eklenir.
* uyeler.uyeEkle(new Uye(isim, uyeNumarasi)) ile üye, Uyeler sınıfındaki uyeEkle metoduna eklenir.

#### **2. Üye Silme**

* Kullanıcıdan bir üye numarası alınarak, Uyeler sınıfındaki uyeSil(int uyeNumarasi) metodu ile ilgili üye listeden silinir.

#### **3. Üye Listeleme**

* Sistemdeki tüm üyeler, uyeler.uyeListele() metodu ile listelenir.

#### **4. Üye Ara**

* Kullanıcı, üye numarası girerek bir üye arar ve uyeler.uyeAra(int uyeNumarasi) metodu ile üye bulunur.

#### **5. Üye Bilgilerini Güncelleme**

* Kullanıcı, bir üyenin bilgilerini güncellemek için üye numarasını girer ve istenen yeni ismi sisteme kaydeder.

#### **6. Üye Ödünç Geçmişini Görüntüleme**

* Kullanıcı, ödünç geçmişini görmek istediği üyenin numarasını girer ve Uye sınıfındaki oduncGeçmisiGoster() metodu ile üyenin ödünç geçmişi görüntülenir.

#### **7. Geri**

* Bu seçenek, kullanıcıyı ana menüye geri döndürür.

**4. Kitap Yönetimi Menüsü: kitapYonetimi()**

kitapYonetimi() metodu, kitap yönetimi ile ilgili işlemleri gerçekleştirmek için kullanılan bir menüyü kontrol eder. Kullanıcı, kitaplarla ilgili şu işlemleri yapabilir:

#### **1. Kitap Ekleme**

* Kullanıcı, kitap başlığını, yazarını ve ISBN numarasını girerek yeni bir kitap ekler. kitaplik.kitapEkle(new Kitap(baslik, yazar, isbn)) metodu ile kitap, Kitaplik sınıfına eklenir.

#### **2. Kitap Ödünç Alma**

* Kullanıcı, bir kitabı ödünç almak için üye numarasını girer. Ardından, kitaplik.kitapOduncAl(isbn) metodu ile kitabın ödünç durumu kontrol edilir ve kitap ödünç alınır. Ödünç alınan kitap, üyenin ödünç listesine eklenir.

#### **3. Kitap İade Etme**

* Kullanıcı, iade etmek istediği kitabın ISBN numarasını ve üye numarasını girer. Uye sınıfındaki kitapIadeEt(isbn, LocalDate.now()) metodu ile kitap iade edilir.

#### **4. Kitap Arama**

* Kullanıcı, ISBN numarasını girerek aradığı kitabı bulur ve kitap bilgileri ekrana yazdırılır.

#### **5. Kitap Listeleme**

* Sistemdeki tüm kitaplar, kitaplik.kitapListesiGoster() metodu ile listelenir.

#### **6. Kitap Silme**

* Kullanıcı, silmek istediği kitabın ISBN numarasını girer. kitaplik.kitapSil(isbn) metodu ile ilgili kitap sistemden silinir.

#### **7. Mevcut Kitap Kontrolü**

* Kullanıcı, kitapların mevcut durumu hakkında bilgi almak için ISBN numarasını girer. kitaplik.mevcutKitapDurumu(isbn) metodu ile kitabın ödünç durumu sorgulanır.

#### **8. Geri**

* Bu seçenek, kullanıcıyı ana menüye geri döndürür.

**5. Program Akışı ve Kullanıcı Etkileşimi**

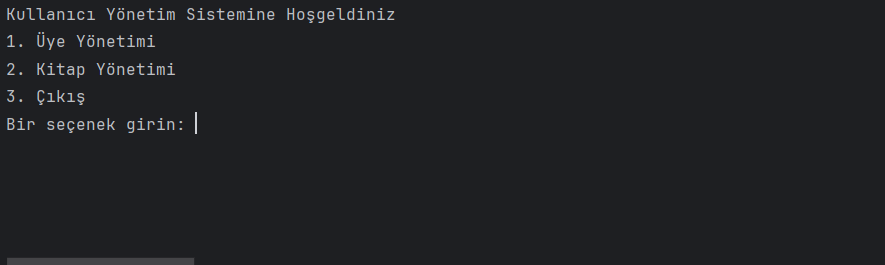
Programda, her bir menü seçeneği kullanıcıdan alınan girdilere göre işlem yapar. Kullanıcı doğru bir seçim yapana kadar menü tekrar gösterilir. Eğer kullanıcı yanlış bir seçim yaparsa, program kullanıcıyı uyarır ve yeniden seçim yapmasını ister.

**6. Kodun Genel Amacı:**

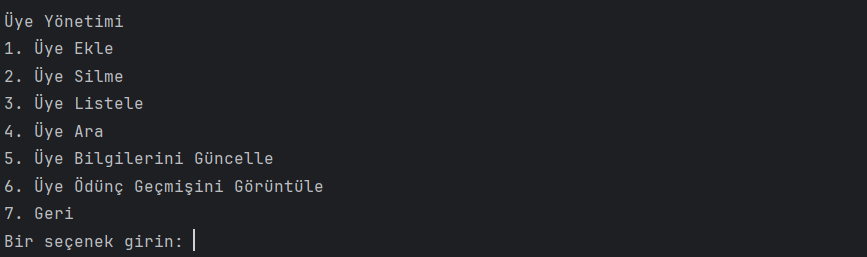
Main sınıfı, kütüphane yönetim sisteminin tüm işlemlerini koordine eden ana sınıf olup, kullanıcının üyelik ve kitap işlemleriyle etkileşime girmesini sağlar. Bu sınıf, programın temel işlevselliğini sağlamak için Kitaplik, Uyeler, Kitap, Uye, ve diğer yardımcı sınıfları etkili bir şekilde kullanarak kütüphane işlemlerini yönetir. Programın esnek yapısı sayesinde kullanıcılar, kitap ekleme, silme, ödünç alma, iade etme gibi işlemleri kolaylıkla gerçekleştirebilir.

**4. Test Aşaması:**

1-Kod çalıştırıldığında ilk çıktı:

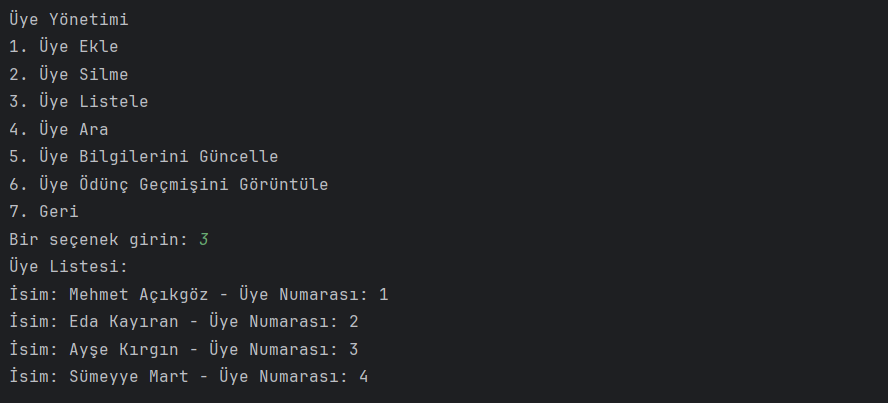


2-Kullanıcı 1 yazdığında çıktı:

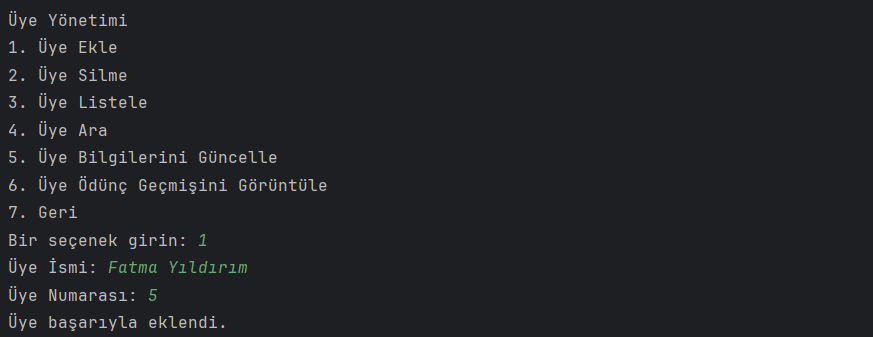


Şimdi sırasıyla üye listeleme, üye ekleme, eklediğimiz üyeyi silme, olmayan üyeyi silmeye çalışma, herhangi bir üye arama, listede olmayan üyeyi aramaya çalışma, tekrar hatalı bir üye ekleme, hatalı üyenin bilgilerini güncelleme, olmayan bir üyenin bilgisini güncellemeye çalışma, üye ödünç geçmişlerini kontrol etme.

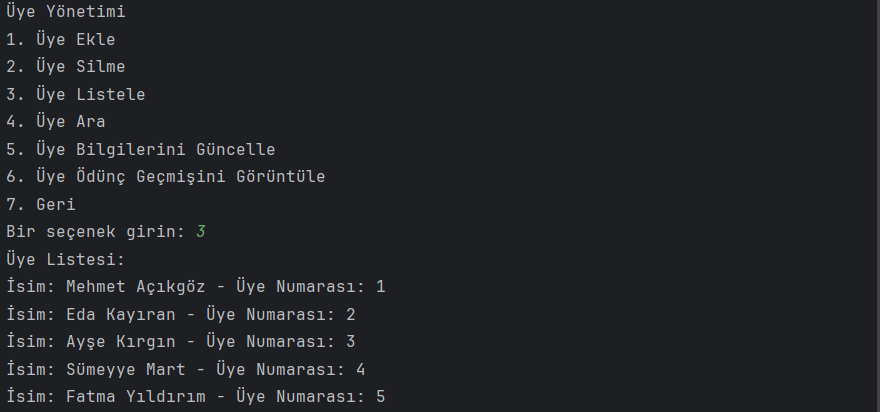
3- Üye listeleme:



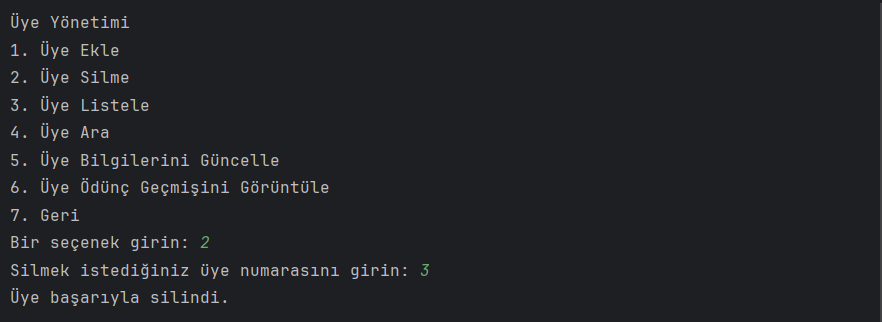
4- Üye ekleme:



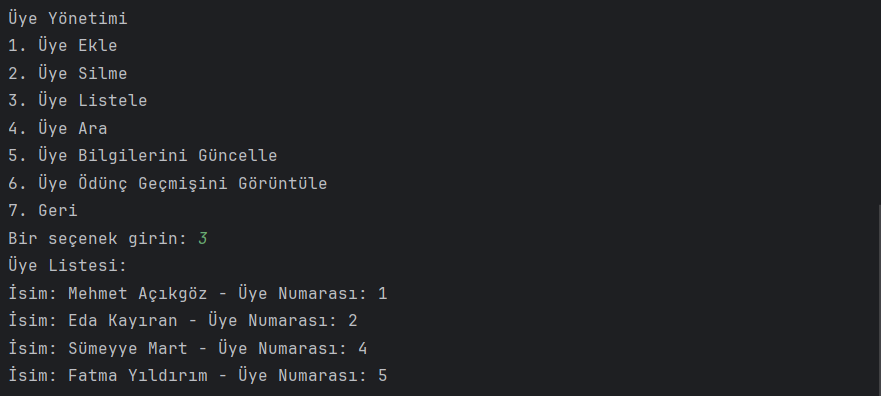
5- Eklenilen üyeyi listede kontrol etme:



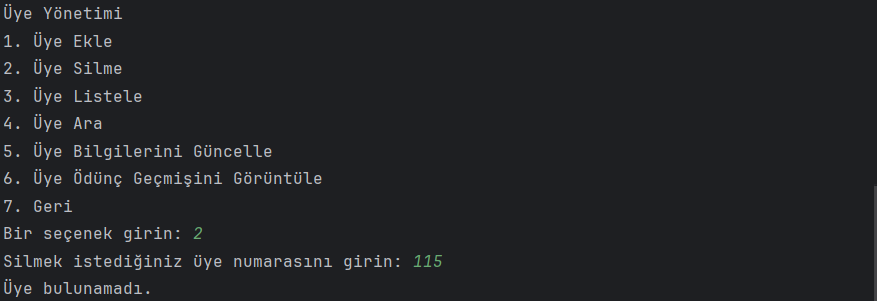
6- Eklenilen üyeyi yada istenilen üyeyi silme:



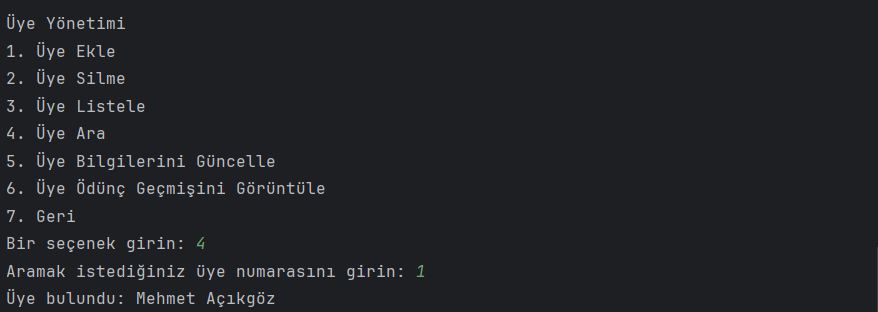
7- Silinen üyenin listeden silindiğini kontrol etme:



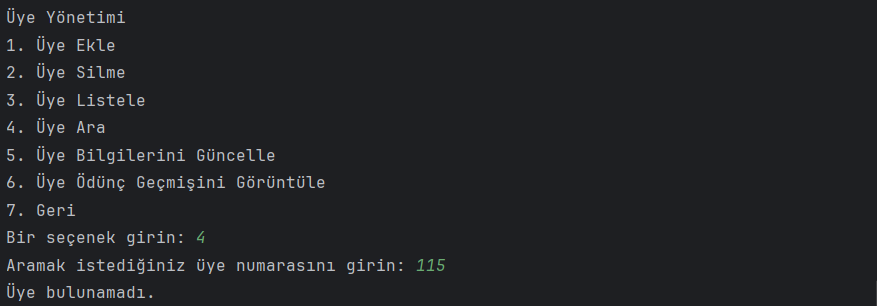
8- Olmayan üyeyi silmeye çalışma:



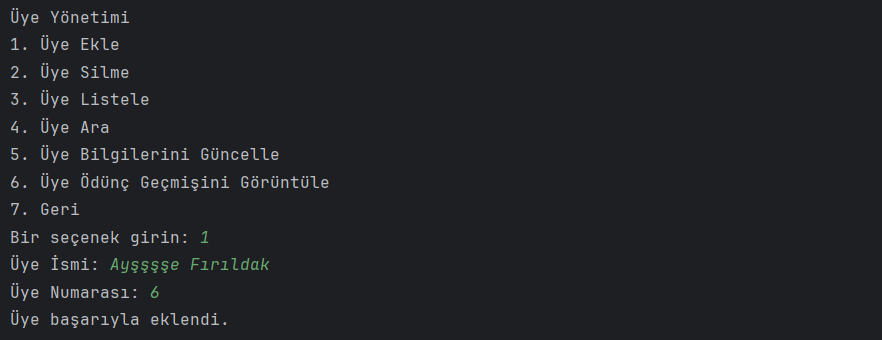
9- Herhangi bir üye arama:



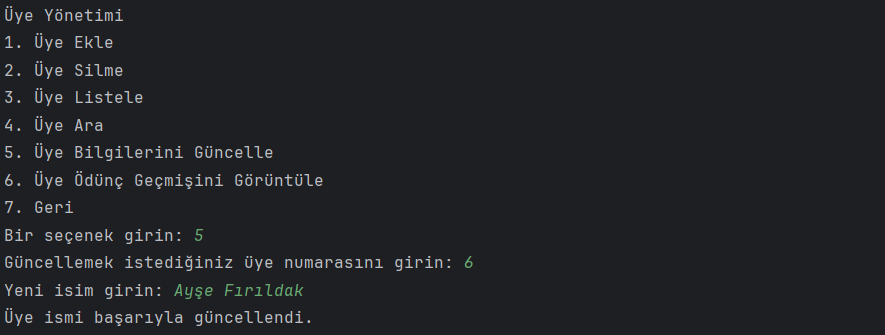
10- Listede olmayan üyeyi aramaya çalışma:



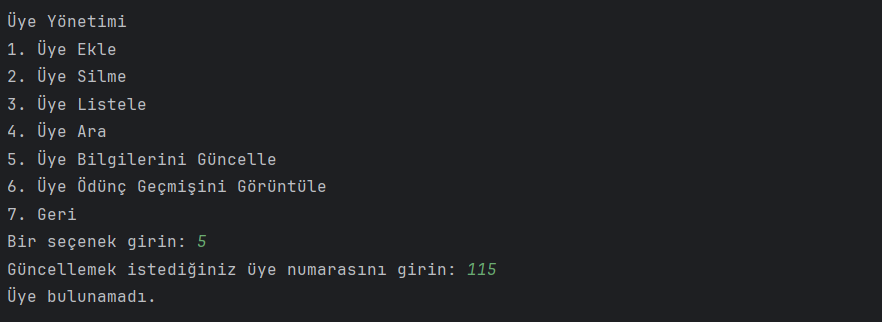
11- İçinde sonradan güncellenmesi gereken hata barındıran bir üye ekleme:



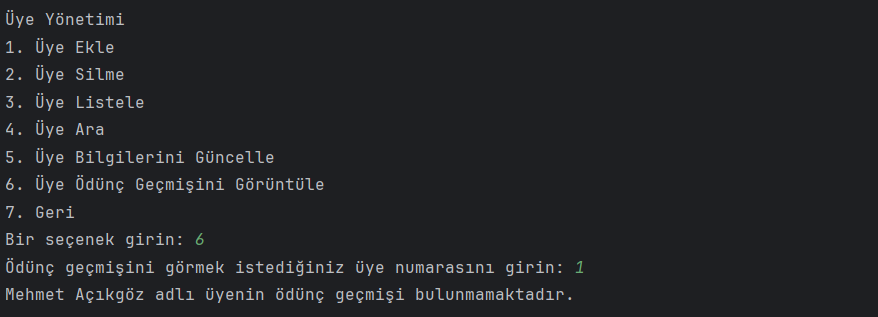
12- Hata bulunduran üyenin üye bilgisini güncelleme:



13- Olmayan bir üyenin bilgisini güncellemeye çalışma:

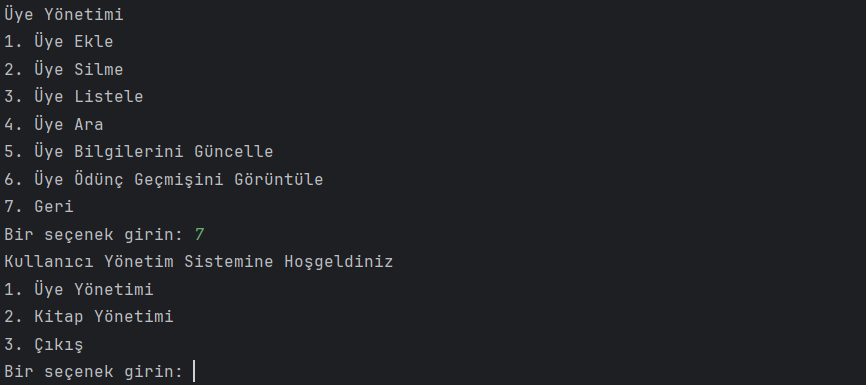


14- Üye ödünç geçmişini kontrol etme:

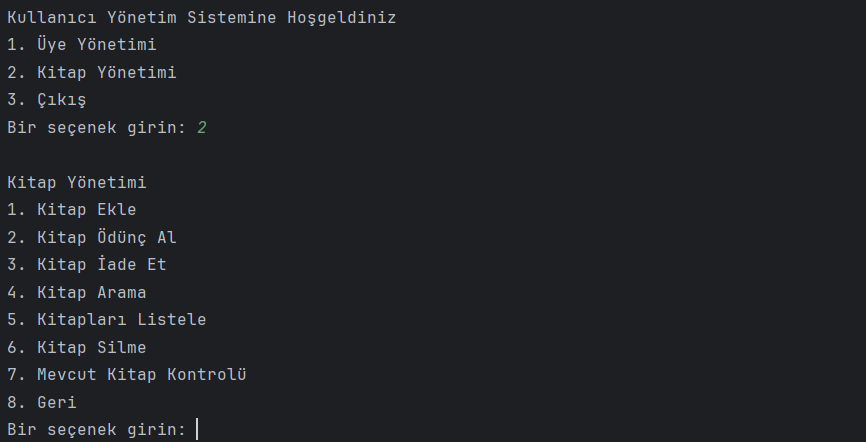


(üye ödünç geçmişleri yok çünkü henüz ödünç alma talebinde bulunmadı)

15- Ana menüye dönme:

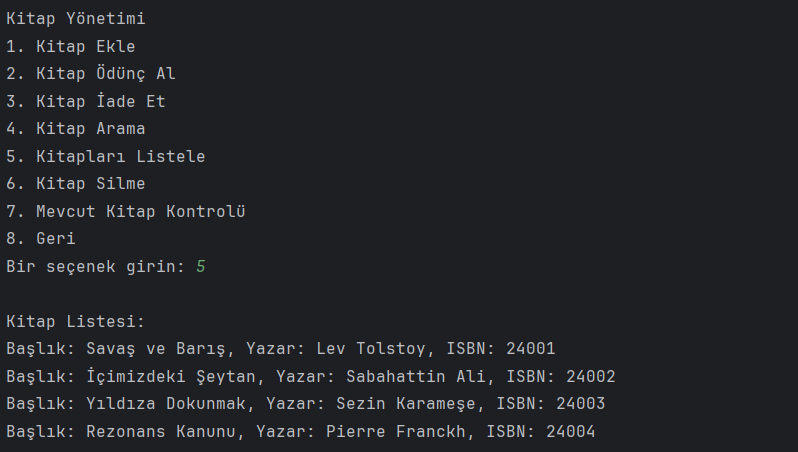


16- Kullanıcı 2 yazdığında çıktı:

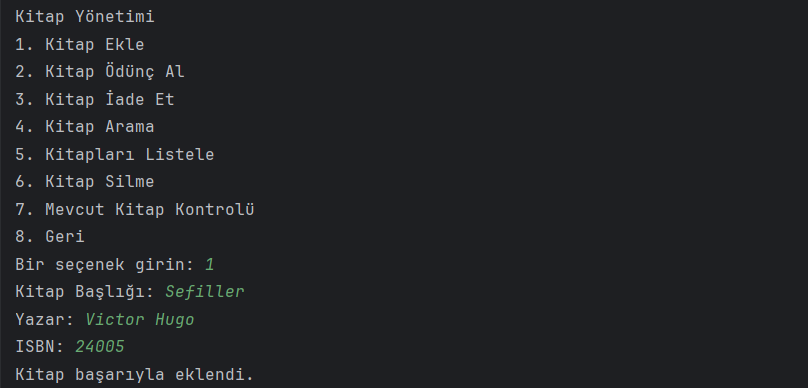


Şimdi sırasıyla kitapları listeleme, kitap ekleme, eklediğimiz kitabı silme, olmayan kitabı silmeye çalışma, kitap arama, olmayan kitabı arama, mevcut kitap kontrolü, mevcut olmayan kitap kontrolü yapmaya çalışma, bir üyeye kitap ödünç alma, olmayan bir üyeye kitap ödünç almaya çalışma, bir üyeye olmayan kitabı ödünç almaya çalışma, kitap listeleme, ödünç verilen kitabın mevcut kitap kontrolünü yapma, kitap iade etme, üye ödünç geçmişine eklenip eklenmediğini kontrol etmek için ilk ödünç alınan kitaptan farklı bir kitap ödünç alma, üye ödünç geçmişini kontrol etme.

17- Kitap listeleme:



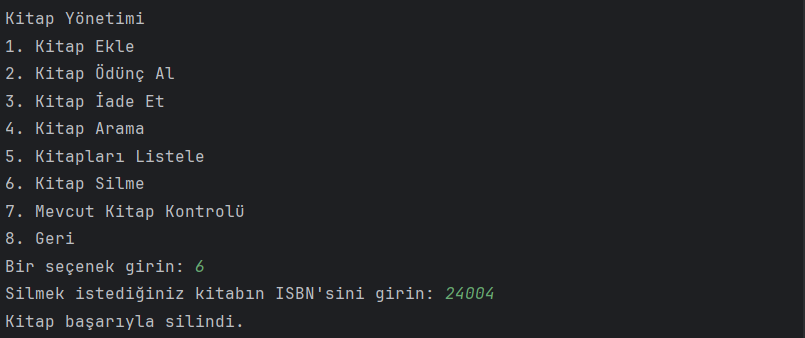
18-Kitap ekleme:



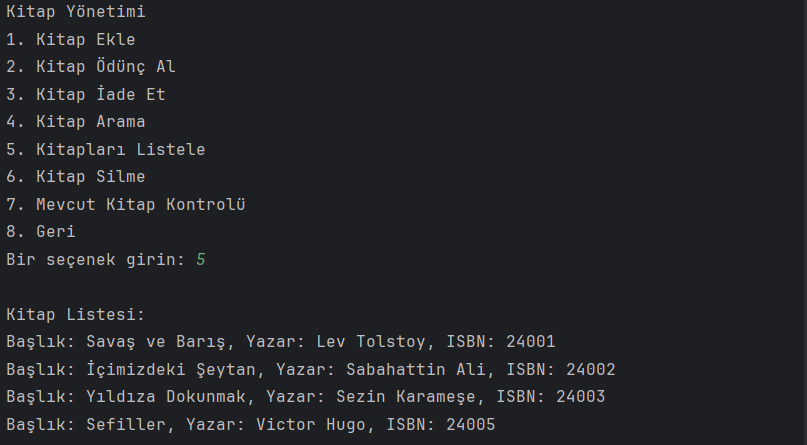
19- Kitap listeye eklendi mi diye kontrol etme:



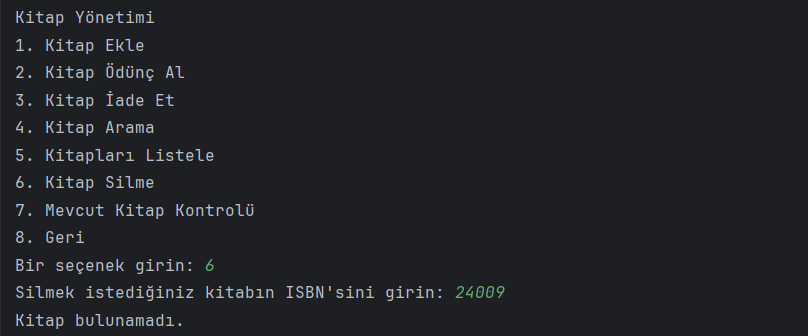
20- Eklenilen veya istediğimiz kitabı silme:



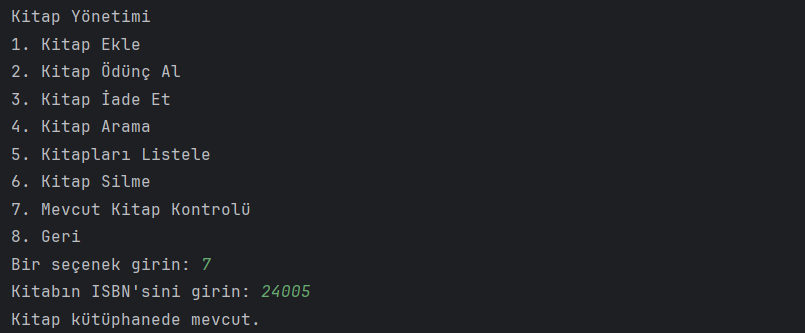
21- Kitap listeden silindi mi diye kontrol etme:



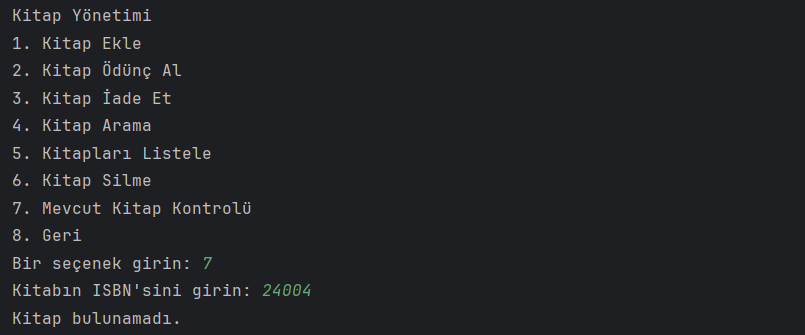
22- Olmayan kitabı silmeye çalışmak:



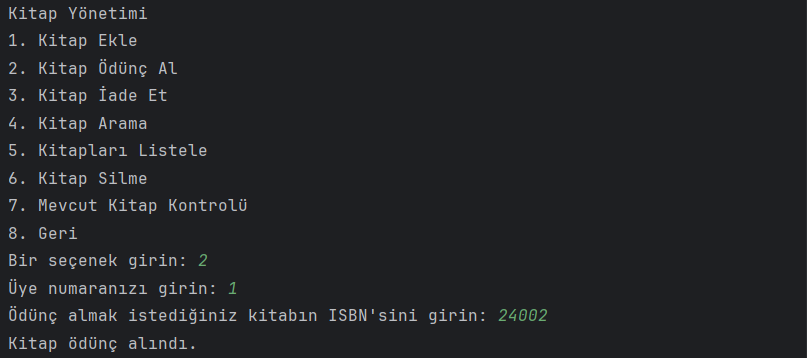
23- Mevcut kitap kontrolü yapma:



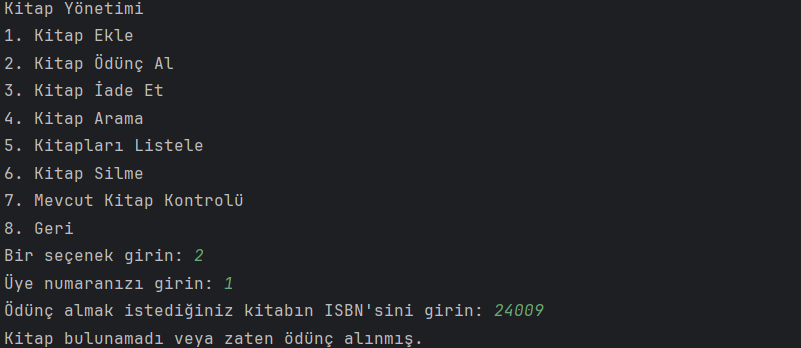
24- Olmayan bir kitabın mevcut kitap kontrolünü yapmaya çalışma:



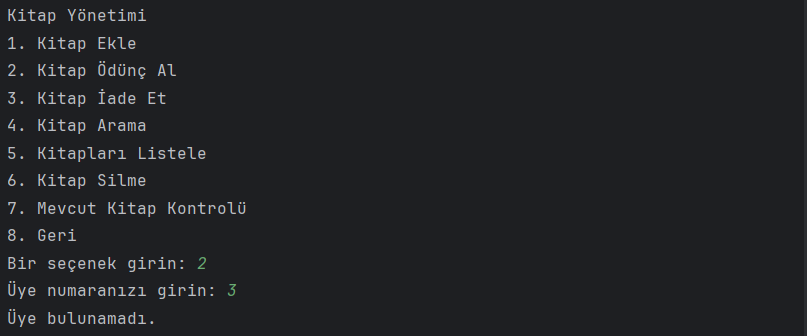
25- Bir üyeye kitap ödünç alma işlemini yaptırtma:



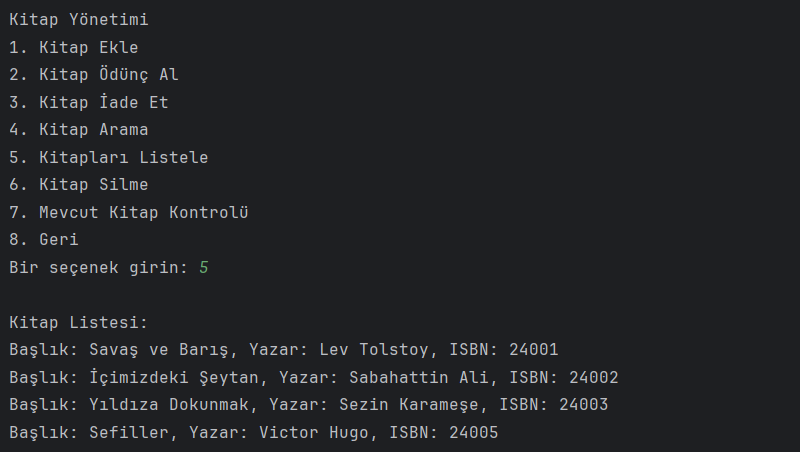
26- Bir üyeye olmayan kitabı ödünç aldırtmaya çalışma:



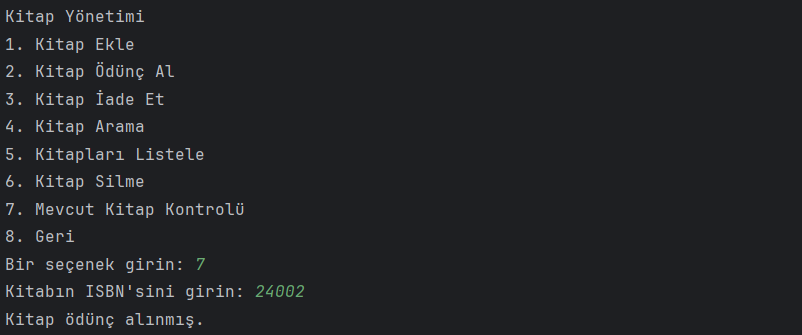
27- Olmayan üyeye kitap ödünç aldırtmaya çalışma:



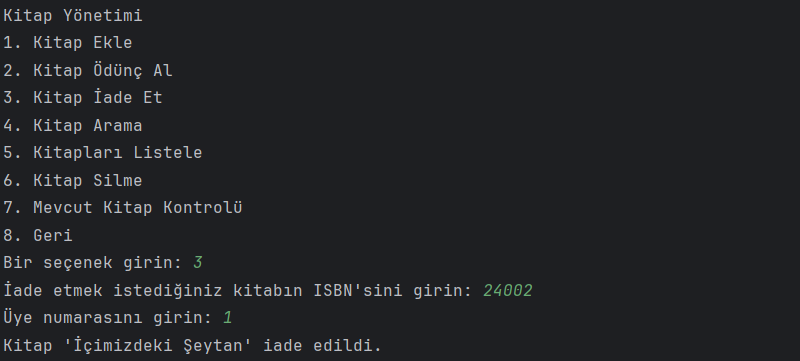
28- İade alındı diye kitaplıktan silinmediğini görmek için kitap listeleme:



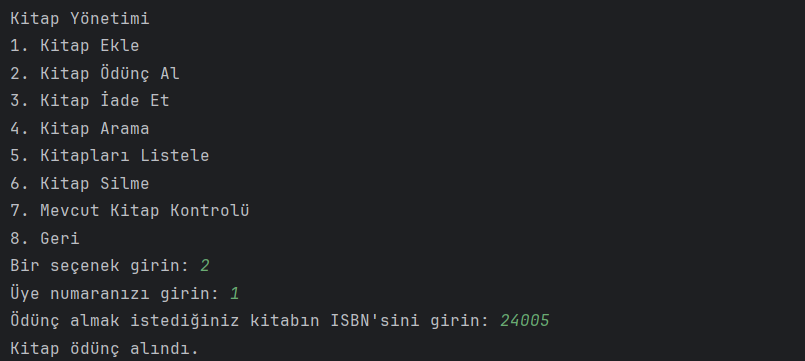
29- Ödünç verilen kitabın mevcut kitap kontrolünü yapma:



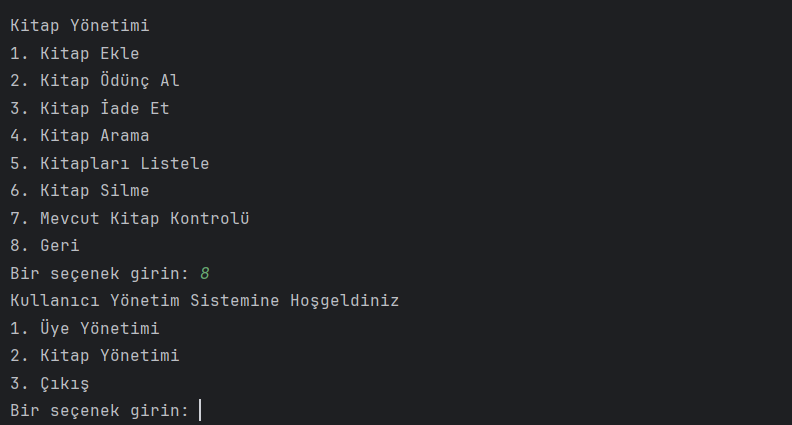
30- Ödünç alınan kitabı iade etme:



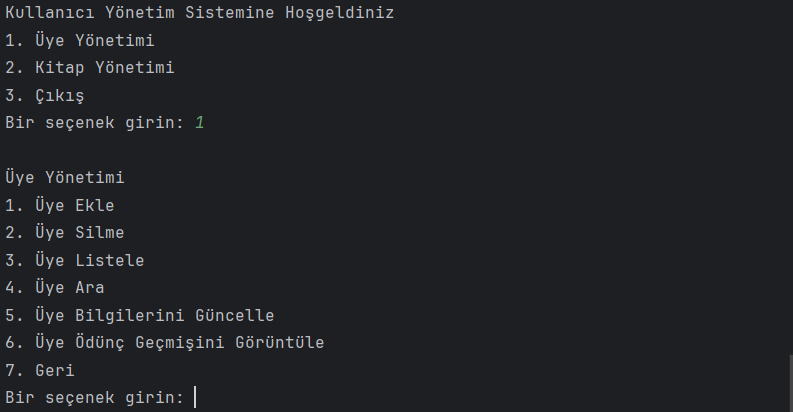
31- Aynı üyeye üye ödünç geçmişine eklenip eklenmediğini kontrol etmek için farklı bir kitap ödünç alma:



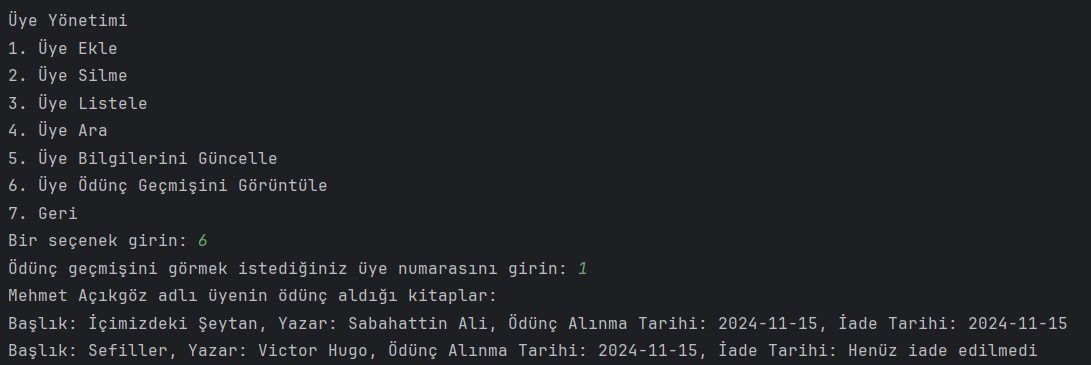
32- Ana menüye dönme:



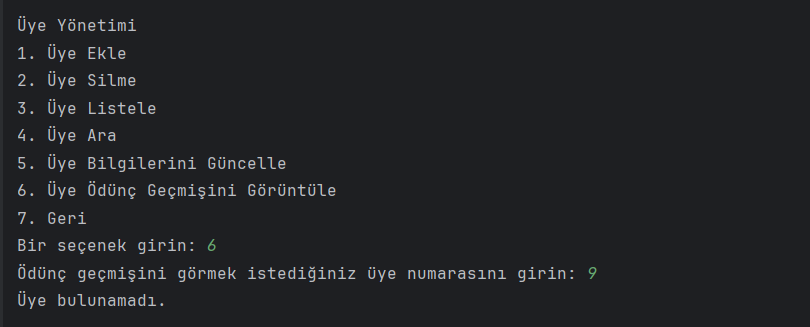
33- Üye yönetimi menüsüne girme:



34- Kitap ödünç alan üyenin üye ödünç geçmişini öğrenmeyi deneme:



35- Olmayan bir üyenin üye ödünç geçmişini öğrenmeyi deneme:



36- Ana menüye dönüp koddan çıkış yaparak kodun çalışmasını durduruyoruz:

### Değerlendirme:

**Sistemin Gereksinimleri Karşılama Derecesi:**

Proje, belirlenen gereksinimlerin hepsini başarıyla karşılamaktadır. Sistemde kitap ekleme, silme, listeleme, arama ve ödünç alma işlemleri sorunsuz bir şekilde gerçekleştirilebilmektedir. Ayrıca, üye yönetimi de tamamen işlevsel olup, üyelerin eklenmesi, silinmesi, bilgilerin güncellenmesi ve ödünç geçmişinin görüntülenmesi gibi temel işlevler doğru şekilde çalışmaktadır. Sistemde kitap ödünç alma ve iade etme süreçleri de düzgün bir şekilde entegre edilmiştir, böylece kullanıcılar kitapları ödünç alabilir ve iade edebilirler.

**Sistem Performansı:**

Sistem, az ve orta düzeyde kitap ve üye bilgileriyle çok iyi performans göstermektedir. İşlemler hızla gerçekleştirilmektedir, çünkü kitap ve üye bilgileri temel olarak liste yapılarıyla yönetilmektedir. Ancak, fazla sayıda bilgi (örneğin, 1000'lerce kitap ve üye) performansı bir miktar düşebilir.

# Olumlu Yönler:

1. **Başarıyla İşleyen Kitap ve Üye Yönetimi:**
   * Kitap ekleme, listeleme, arama ve silme işlemleri doğru şekilde çalışıyor.
   * Üye yönetimi, üyelerin bilgilerini düzenleme ve ödünç geçmişlerini görüntüleme işlemleri sorunsuz.
   * Kitap ödünç alma ve iade etme işlemleri doğru şekilde işliyor.
2. **Hızlı İşlem Süreleri:**
   * Küçük ve orta ölçekli veri setlerinde hızlı işlem süreleri.
   * Kitap ve üye işlemleri arasında gecikme olmadan gerçekleştirilmesi.
3. **Modüler Yapı:**
   * Sistemin her modülü (Kitap, Üye, Ödünç Kitap) ayrı ayrı yönetildiği için anlaşılabilir ve bakım yapılması kolay.

**Sonuç:**

Proje, temel kütüphane yönetim sisteminin gereksinimlerini karşılayan, kitap ve üye yönetimini, ödünç alma ve iade işlemlerini sorunsuz şekilde işleyen işlevsel ve kullanıcı dostu bir sistem olarak başarılı bir şekilde çalışmaktadır.