CASE-1-Başlangıç Seviye: Kütüphane Otomasyon Sistemi

Bir kütüphane programı, kitapların ödünç alınması ve iade edilmesi için bir otomasyon sistemi oluşturmanız isteniyor. Bu sistemi Java programlama dili ile OOP prensiplerine uygun bir şekilde geliştirmeniz gerekiyor.

Kütüphanemizde birden fazla türde kitaplar bulunmaktadır. Örneğin (Bilim, Roman, Tarih, v.b.) bu kitap türleri zaman içerisinde **genişletilebilir** yapıda olmalıdır. Kütüphanemizde **tek tip üye** bulunmaktadır. Kütüphane sistemine dahil olan herkes kütüphane stoğunda bulunan kitaplardan ödünç alabilir. (Ödünç işlemi için şartlar; sisteme dahil olmak ve kütüphane stoğundan yeterli sayıda kitap olması.)

Beklenen İşlevler:

- Bir üye kitap ödünç alabilmeli.
- Bir üye ödünç aldığı kitabı iade edebilmeli.
- Bir kitabın mevcut durumu güncellenebilmeli (ödünç alınabilir, ödünçte, mevcut değil).
- Bir üyenin ödünç aldığı kitapları görüntüleyebilmeli.
- Kütüphane durumunu (mevcut kitaplar, ödünç alınan kitaplar, üyeler vb.) görüntüleyebilmeli.

Başlangıç:

Öncelikle yeni bir java projesi oluşturalım ve adını Case01 _JavaLibraryManagement diyelim. Bu uygulama aşağıda gösterildiği gibi altı sınıftan oluşmaktadır.

- Kitap (Abstract Class)
- Durum (Enum)
- KitapBilim: Kitap (Sub Class)
- KitapRoman: Kitap (Sub Class)
- KitapTarih: Kitap (Sub Class)
- IUye : (Interface)
- Uye: IUye (Inheritance)
- Kutuphane (Class)

Temel Bileşenler:

1. **Kitap:** Kitapların temel özelliklerini (ISBN, başlık, yazar, yayın yılı vb.) içerir ve diğer kitap türleri için temel (base class) görevi görür.
2. **KitapBilim, KitapRoman, KitapTarih:** Kütüphanede bulunan kitap türlerimize göre BaseKitap'dan kalıtım alırak kütüphanedeki kitap türü kadar alt sınıf (sub class) oluşturulur. (KitapBilim, KitapRoman, v.b.) Bu sınıf mevcut durumunu (ödünç alınabilir, ödünçte, mevcut değil) tutan bir sınıf olmalıdır.
3. **Durum:** Kitap sınıfı içerisinde kullanmak üzere kitapların ödünç durumunu belirtmek için (OduncAlabilir, OduncVerildi, MevcutDegil, v.b.) bir enum tanımlayın.
5. **IUye*:*** Üyelerin ödünç kitap alma ve iade etme gibi işlemleri gerçekleştirmelerini sağlayan metotları tanımlayan bir interface. Bu interface, üyelerin kütüphane sistemiyle etkileşimini yöneten metotları içerir.
6. **Üye:** IUye interface'inden implent edilerek kütüphaneye üye olan kişilerin özelliklerini ve metotlarının uygulandığı bir sınıf.
6. **Kütüphane:** Kitapları ve üyeleri yöneten ana sınıf. Kitapların ödünç verilmesi, iade edilmesi ve mevcut durumunun güncellenmesi gibi işlemleri yapacak metotları içermeli.

Bu programı hazırlayanız. Buradaki yapılar dışında ekleme yapmak istediğiniz tüm yapıları ekleyebileceğinizi **unutmayınız**.

CASE-2-Ileri Seviye: Maaş Bordro Programı

Küçük bir şirketin maaş bordrolarını oluşturan bir uygulama oluşturalım. Şirketimize yazılacak olan maaş bordro program en az iki tipte (Yonetici ve Memur) personeline maaşlarını hesaplamak, kayıt altına almak ve raporlama işlemlerine sahip olmalıdır. Ayrıca projemize daha sonradan oluşabilecek yeni personel kadroları dahilinde genişletilebilir olmalıdır.

Beklenen İşlevler:

- Her personelin maaş hesabı saatlik ücret * çalışma saati bilgileri doğrultusunda hesaplanır.
- Yoneticinin saatlik ucreti 500 den kucuk olamaz ve her her yoneticiye maaş dışında bonus adlı ek bir ödeme alır.
- Memurların maaşı maksimum 180 saat den hesaplanır. 180 saati geçen her çalışma süresi normal saatlik ücretin 1.5 katı bedelle belirlenerek ek mesai ücreti olarak ana maaşa eklenir.
- Memurun saatlik ücreti varsayılan olarak 500 TL dir. Fakat memurun derecesine göre değişebilir olmalıdır. Dereceyi bir enum olarak oluşturabilirsiniz. (JUNIOR,MID,SENIOR)
- Memur ve Yonetici listesi bir json dosyası olarak verilecektir. Program maaş hesaplamaya .json dosyasından okuma yaparak sırasıyla personelin maaş bilgilerinin girişi yapılmasını isteyecektir.

Örnek .json dosyası:

```
[
 {
  "name": "Fatih",
  "surname": "Alkan",
  "role": "Yonetici"
 },
 {
  "name": "Beyazıt",
  "surname": "Dalgıç",
  "role": "Memur"
 }
]
```

- Hesaplanan maaş bilgileri her personelin adına açılan klasörün içersine maaş tarih bilgisiyle birlikte .json formatında kayıt edilecektir.

Örnek .json dosyası:

```
{
 "bordro": "SUBAT 2020",
 "personel": {
  "ismi": "Beyazıt",
  "calismaSaati": 200,
  "odemeDetaylari": {
   "anaOdeme": "₺9.000,00",
   "mesai": "₺1.000,00",
   "toplamOdeme": "₺10.000,00"
  }
 }
}
```

- Program sonunda maaş hesabı yapılan tüm personelin rapor görüntüsünün ekrana yazdırılmasını ve ayrıca 150 saat az çalışan personellerin bilgilerinin belirtilmesi gerekmektedir.

Başlangıç:

Öncelikle yeni bir class library projesi oluşturalım ve adını CSProjeDemo2 diyelim. Bu uygulama aşağıda gösterildiği gibi altı sınıftan oluşmaktadır.

- Personel (Abstract)
- Yonetici: Personel (Sub Class)
- Memur: Personel (Sub Class)
- DosyaOku
- MaasBordro
- Program

Temel Bileşenler:

1. **Personel:** Personel sınıfı, bir çalışan hakkında temel bilgileri içerir ve temel ücreti hesaplamak için bir metot sağlar. Diğer iki sınıfın türetileceği bir üst sınıf olarak hizmet eder
2. **Yönetici ve Memur:** Personel sınıfını devralır ve MaasHesapla() metodunu override eder.
3. **DosyaOku**: bir .json dosyasından okuyan ve .json dosyasındaki içeriğe dayalı olarak Personel nesnelerinin bir listesini oluşturan basit bir metot içerir.
4. **MaasBordro**: Şirketteki her çalışanın maaş bordrosunu oluşturur. Ayrıca ayda 10 saatten az çalışan personelin ayrıntılarının bir özetini de oluşturur.
5. **Program**: Son olarak, bir console uygulaması ile gelişrilen class library çalıştırdığımız projemiz.

Yukarıdaki temel anlatım doğrultusunda projeyi, yazmayı deneyiniz. Sizlere yukarıda olması gereken temel sınıflar için bir yol harıtası çıkarılmıştır. Projenize OOP presibleri doğrultusunda eklemeler yapabileceğiniz unutmayınız.