Gebze Technical University
Computer Engineering

CSE 443 - 2018 Fall

HOMEWORK 03 RAPOR

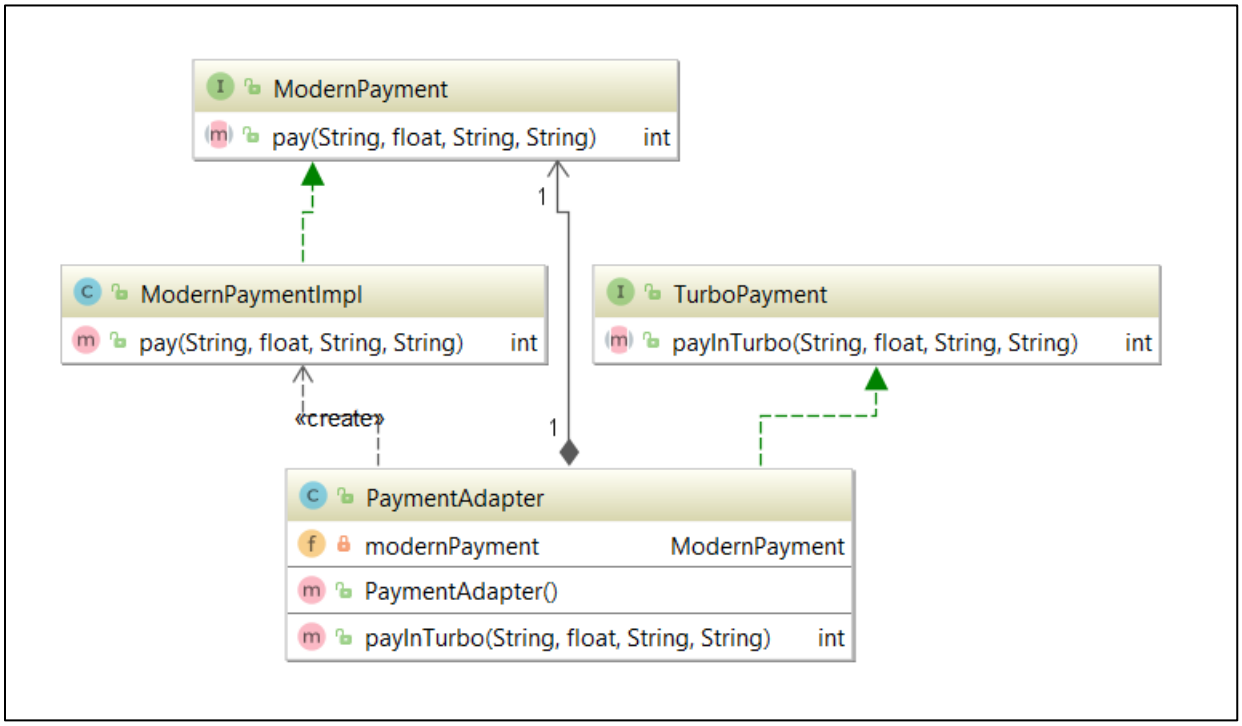
Elif Şeyma ARMAĞAN
151044042

Course Assistant:
Muhammed Ali DEDE

Part 1

Burada Adapter Pattern kullanılır. Turbo ve Modern payment interface'leri ile birlikte Adapter sınıfını da oluşturdum. Adapter sınıfında private olarak ModernPayment tutuluyor ve Adapter sınıfı TurboPayment'ı implement ediyor. payInTurbo metodunda ise işi modernPayment'ı çağırarak yapıyor.

Sınıf diyagramı:



Part 2

Burada Composite Pattern'ı uygulamak için öncelikle hem bireysel hem grup adreslerin atası olacak şekilde Email interface'ini oluşturdum. Tek metodu getAsStr(), yani email adresi ve ismini string şeklinde formatlayıp veren bir metod.

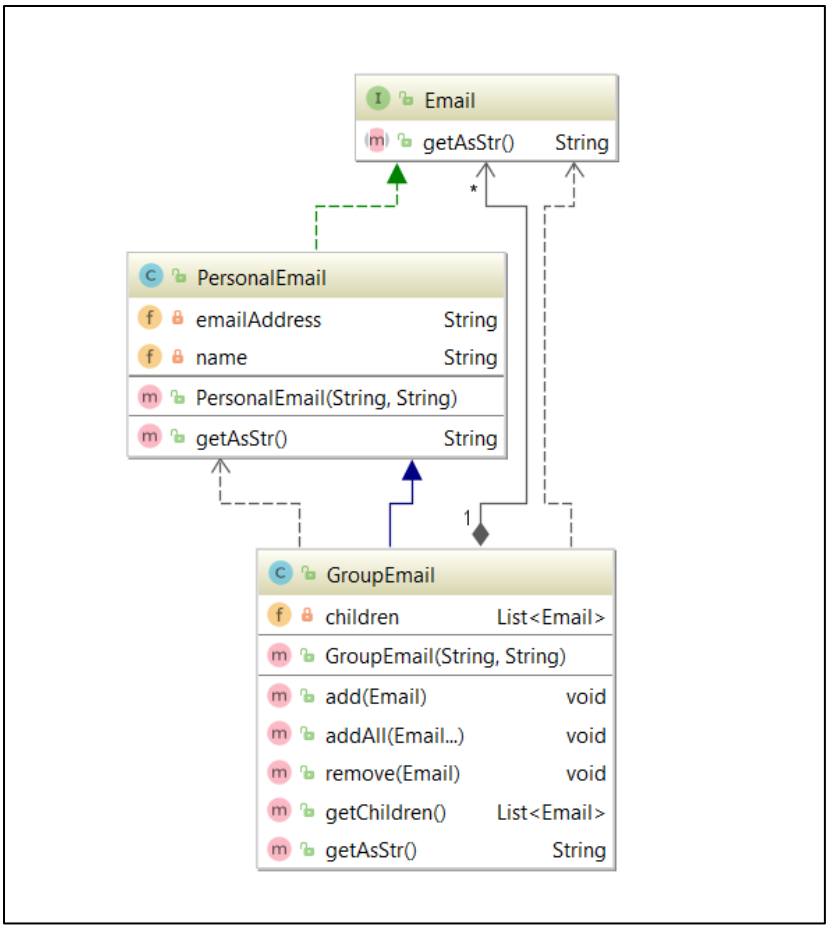
Bireysel email adresi bu interface'i implement ediyor ve içerisinde prive olarak adres ve ismi bulunduruyor. getAsStr() implementasyonu basitçe adres ve ismi birleştirip return ediyor.

Grup email adresini ise bireysel email adresinden türettim çünkü grup email adreslerinin de bir adresi ve ismi bulunuyor (altındaki diğer email adresleriyle birlikte). Bunları tekrar yazmaktansa bireyselden türetmek daha doğru geldi.

Grup email adresleri, bunların dışında bir de Email listesi bulunduruyor. Bu listeye ekleme ve çıkarma işlemleri için metodlar yazdım.

getAsStr() metodunu override edip, grup adresi ve ismiyle birlikte altındaki tüm child email adreslerinin getAsStr metodlarını çağırarak composite bir string oluşturdum.

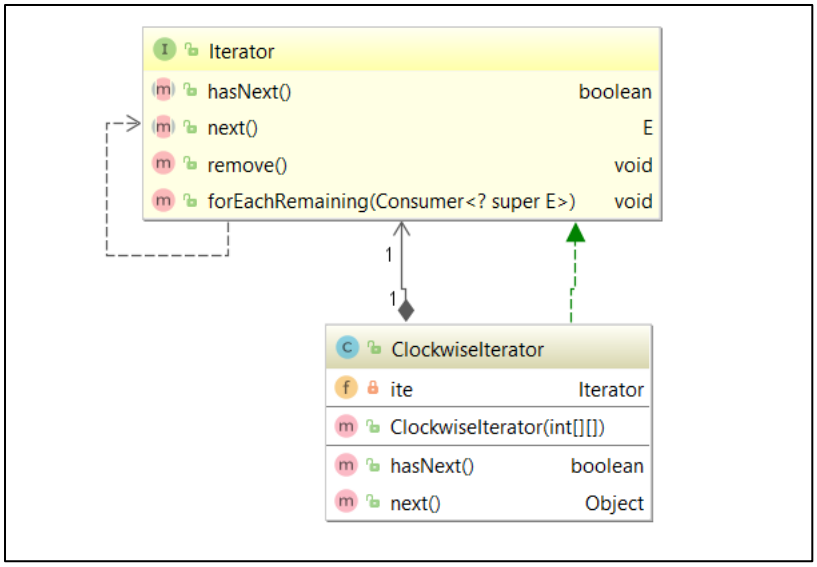
Sınıf diyagramı şu şekilde:



Part 3

Bu soruda, saat yönünde dönen iterator'u implement ettim. Java'nın Iterator interface'i implement eden sınıfım, içerisinde bir tane Integer iterator'u da bulunduruyor. Constuctor'ında iki boyutlu array'i alıyor ve içerisinde bir Integer listesine doğru sıralama ile ekliyor. Daha sonra, bu listenin iterator'unu alıp, sınıfımdaki getNext() ve hasNext() metodlarında bunu kullandım.

Sınıf diyagramı şu şekilde:



Part 4

Template Pattern için ortak metodların ve algoritmanın bulunduğu DiscreteTransformation sınıfını yazdım. Bu sınıf abstract, objesi oluşturulamıyor dolayısıyla her bir metod bu sınıfı extend edip, abstract metodu override edecek. Bu sınıfın içinde readFromFile, writeToFile, afterHook, transformation, getElapsedTime ve calculateTransformation metotları bulunuyor.

calculateTransformation, adımları içeriyor yani diğer gerekli metodların sırayla çağırmasını barındırıyor. readFromFile ve writeToFile dosyadan okuyup dosyaya yazıyor. İki metod için de dosya okuma işlemi aynı olduğundan (ikisinde de satır satır sayıları okuyor) implementasyonları DiscreteTransformation içerisinde yapıldı.

getElapsedTime metodu trasformation için geçen süreyi veriyor.

afterHook metodu, calculateTransformation içerisinde, transformation işlemi bittikten sonra çağırılıyor. Default implementasyonu boş, bir işlem yapmıyor. Ancak DiscreteFourierTransform sınıfı içerisinde bunu override edip, ekrana elapsedTime'ın basılmasını sağladım. Bu metodu override etmeyen sınıflar için bir işlem yapılmıyor, bir etki gösterilmiyor.

transformation metodu ise asıl işlemin yapıldığı metod ve DiscreteTransformation içerisinde abstract olarak tanımlandı. DiscreteFourierTransform ve DiscreteCosineTransform sınıfları bu metodu override edip kendi algoritmalarını gerçekleştiriyor.

Kullanırken sadece calculateTransformation metodu çağrılıyor, bu metod içerisinde dosya okuma, sonuç hesaplama ve dosyaya yazma adımları çağırılıyor.

Sınıf diyagramı: