السؤال الأول:

اذا كان balance برايفت سوف يكون من الصعب الوصول اليه مباشره بسبب انه برايفت الا بطرق معينة ()getBalance استعمال القيت للوصول الى الرصيد والتمكن من فرض العقوبة.

السؤال الثاني:

PredatoryCreditCard مخصصه تسوي العقوبة اذا فشل الشحن فاذا ماسوت أي عقوبة لفشل الشحنة يعني وظيفتها الي المفروض تسويهاا يعني كتابة الكود غير صحيحة تماما لان وظيفتها فرض عقوباتت على الشحن.

السؤال الثالث:

هذا الكود ينفذ سلسلة فيبوناتشي وتبدأ القيمه من 2,2 ونحسب حساب قيم السلسلة على زيادة القيمتين السابقتين الثنتين لين الوصول ال الرقم الثامن في السلسلة زي ٢، ٤، ٦، ١٠، ١٦، ٢٦، ٤٢.

السؤال الرابع:

Firstهيا القيمة الأولى و١٢٨ هوه الرقم الثابت بين القيم وعدد القيم التي يمكننا توليدها باستخدام التقدم الحسابي 1−263≈128×(n−1)+first يمكن استدعائها `nextValue()` حوالي مرة 1012×7.18{12}^10 .

السؤال الخامس:

لا يمكن لواجهتين (Interfaces) أن تمتد كل منهما من الأخرى بشكل متبادل، وذلك بسبب المشاكل المحتملة مثل **اللبس ان تكون واجهتان تمتدان من بعضهم البعض (ambiguity)** و **التعارضات .(conflicts)** تكرار الوراثه هذه العملية تُسمى **الوراثة الدائرية (Cyclic inheritance)**أو **الوراثة الحلقية**، وهي غير مدعومة في جافا.

السؤال السادس:

من العيوب التعقيد يصير من الصعب تتبع الفئات المورثه ويصير صعب نعدل ونضيف في الكود بسبب تعقد الفئات وبتكون في تعارضات كثير بين الفئاتت وبصير صعب إضافة فئات جديدة وحتى صعب نعدل بالفئات الموجودة واستهلاك الذاكرة بشكل كبيرر.

السؤال السابع:

العيوب المحتملة من حيث الكفاءة مثل زيادة التعلق بالفئة الأب، صعوبة التوسع، زيادة تعقيد الفئة الأب، والتأثير على الأداء. في بعض الحالات، قد تكون الحلول البديلة مثل الوراثة المتعددة أو استخدام الأنماط التصميمية الأخرى أكثر كفاءة ومرونة.

السؤال الثامن:

المخرج

Read it.

Box it.

Buy it.

Read it.

Box it.

Read it.

السؤال التاسع:

محلول بالمحاضره رسمه

السؤال العاشر:

لا يمكن تحويل كائن من النوع **Equestrian** إلى النوع **Racer** لأنه لا يوجد علاقة **فرعية** أو **عليا** بين **Equestrian** و **Racer**، يعني **Equestrian** لا تمتد من **Racer** ولا العكس. لذلك، التحويل بينهما مايصلح.

السؤال الحادي عشر:

السؤال الثاني عشر :

```
public void makePayment(double amount) { // make a payment
        if(amount<0)
           throw new IllegalArgumentException("Negative Amount is not
Allowed");
      balance -= amount;
      }
```