**Data Structure Lab2 - Object-Oriented Design**

1. **R-2.4 Assume that we change the CreditCard class (see Code Fragment 1.5) so that instance variable balance has private visibility. Why is the following implementation of the PredatoryCreditCard.charge method flawed?  
   الطريقة PredatoryCreditCard.charge تحتوي على خطأ لأنها قد تؤدي إلى حلقة لا نهائية. إذا فشلت عملية الشحن الأولية، يتم استدعاء الطريقة نفسها مرة أخرى مع إضافة غرامة 5، مما قد يؤدي إلى استمرار الاستدعاءات إلى ما لا نهاية.**
2. **R-2.5 Assume that we change the CreditCard class (see Code Fragment 1.5) so that instance variable balance has private visibility. Why is the following implementation of the PredatoryCreditCard.charge method flawed?  
   الطريقة تحتوي على خطأ لأنه لا يمكن فرض رسوم إضافية إذا كانت الرسوم ستؤدي إلى تجاوز الحد الائتماني. يجب التحقق مما إذا كانت الرسوم الإضافية ستتجاوز الحد قبل إضافتها.**
3. **R-2.6 Give a short fragment of Java code that uses the progression classes from Section 2.2.3 to find the eighth value of a Fibonacci progression that starts with 2 and 2 as its first two values.**

**java**

**CopyEdit**

**FibonacciProgression fibonacci = new FibonacciProgression(2, 2);**

**fibonacci.printProgression(8);**

1. **R-2.7 If we choose an increment of 128, how many calls to the nextValue method from the ArithmeticProgression class of Section 2.2.3 can we make before we cause a long-integer overflow?  
   يمكن استدعاء nextValue حوالي 7.18 × 10¹² مرة قبل أن يحدث تجاوز للقيم القصوى (long overflow).**
2. **R-2.8 Can two interfaces mutually extend each other? Why or why not?  
   لا يمكن لواجهتين أن ترثا من بعضهما البعض بشكل متبادل بسبب التكرار اللانهائي الذي قد يحدث (Cyclic inheritance). ومع ذلك، يمكن تحقيق الوظائف المطلوبة باستخدام الوراثة المتعددة.**
3. **R-2.9 What are some potential efficiency disadvantages of having very deep inheritance trees, that is, a large set of classes, A, B, C, and so on, such that B extends A, C extends B, D extends C, etc.?**
   * **يؤدي إلى زيادة زمن البحث عن الوظائف في الطبقات العليا.**
   * **يزيد من تعقيد فهم الكود وصيانته.**
   * **يجعل التعديلات أكثر صعوبة بسبب تأثيرها على الطبقات الفرعية.**
4. **R-2.10 What are some potential efficiency disadvantages of having very shallow inheritance trees, that is, a large set of classes, A, B, C, and so on, such that all of these classes extend a single class, Z?**
   * **تكرار الشيفرة في العديد من الفئات بدلاً من إعادة استخدامها عبر الوراثة.**
   * **يؤدي إلى ضعف تنظيم الفئات بسبب كثرة الوظائف في الفئة الأساسية.**
   * **يجعل تعديل الفئة الأساسية أكثر خطورة لأنه قد يؤثر على جميع الفئات المشتقة منها.**
5. **R-2.11 Consider the following code fragment, taken from some package: (الكود في السؤال)، What is the output from calling the main( ) method of the Maryland class?**

**mathematica**

**CopyEdit**

**Read it.**

**Box it.**

**Buy it.**

**Read it.**

**Buy it.**

**Read it.**

1. **R-2.12 Draw a class inheritance diagram for the following set of classes: (تفاصيل الفئات في السؤال).**

**bash**

**CopyEdit**

**Object**

**/ | \**

**Goat Pig Horse**

**| | / \**

**tail nose Racer Equestrian**

**| |**

**race() weight, isTrained**

1. **R-2.13 Consider the inheritance of classes from Exercise R-2.12, and let d be an object variable of type Horse. If d refers to an actual object of type Equestrian, can it be cast to the class Racer? Why or why not?  
   لا يمكن تحويل كائن Equestrian إلى Racer لأنهما ليسا في نفس سلسلة الوراثة المباشرة. كائن Equestrian مشتق من Horse، لكنه لا يمت بصلة إلى Racer، مما يؤدي إلى خطأ في التحويل.**
2. **R-2.14 Give an example of a Java code fragment that performs an array reference that is possibly out of bounds, and if it is out of bounds, the program catches that exception and prints the following error message: “Don’t try buffer overflow attacks in Java!”**

**java**

**CopyEdit**

**public static void main(String[] args) {**

**int[] x = {11, 12, 13, 14, 15};**

**System.out.println("input index to print negative number to exit");**

**Scanner input = new Scanner(System.in);**

**int y = input.nextInt();**

**while (y >= 0) {**

**try {**

**System.out.println(x[y]);**

**} catch (ArrayIndexOutOfBoundsException e) {**

**System.out.println("Don’t try buffer overflow attacks in Java!");**

**}**

**y = input.nextInt();**

**}**

**}**

**هذه الشيفرة تلتقط أي محاولة للوصول إلى عنصر خارج حدود المصفوفة وتعرض رسالة تحذيرية بدلاً من حدوث خطأ في التنفيذ.**

1. **R-2.15 If the parameter to the makePayment method of the CreditCard class (see Code Fragment 1.5) were a negative number, that would have the effect of raising the balance on the account. Revise the implementation so that it throws an IllegalArgumentException if a negative amount is sent as a parameter.**

**java**

**CopyEdit**

**public void makePayment(double amount) {**

**if (amount < 0)**

**throw new IllegalArgumentException("Negative Amount is not Allowed");**

**balance -= amount;**

**}**

**هذه الشيفرة تمنع إدخال قيم سالبة عند الدفع عن طريق توليد استثناء (IllegalArgumentException).**

**Top of Form**

**Bottom of Form**