Université de Nice Sophia - Antipolis **2014 – 2015**
Polytech'Nice Sophia - SI 3

NOM: .....

PRENOM: .....

GROUPE: .....

Programmation Orientée Objet

12 janvier 2015

Durée : 2h00

Vous apporterez un très grand soin à la présentation car elle interviendra dans la notation. Par exemple, les réponses très peu lisibles ou contenant du code non indenté ***seront considérées comme fausses.*** *Par ailleurs, la qualité du code proposé et la complexité des solutions interviendront dans la notation.*

Documents autorisés.

Question 1: *(2 pts)*

La classe ci-dessous...

```
public class CiDessous {
    private CiDessous() {
    }

    @Override
    public String toString() {
        // implementation adaptee
    }
}
```

(a). peut être sous-classée : vrai faux
Expliquez votre réponse.

(b). peut être instanciée : vrai faux
Expliquez votre réponse.

Question 2: *(2 pts)*

La classe ci-dessous...

```
public final class CiDessous {

    @Override
    public String toString() {
        // implementation adaptee
    }
}
```

(a). peut être sous-classée : vrai faux
Expliquez votre réponse.

(b). peut être instanciée : vrai faux
Expliquez votre réponse.

Question 3: *(2 pts)*

La classe ci-dessous...

```
public abstract class CiDessous {

    @Override
    public abstract String toString();
}
```

(a). peut être sous-classée : vrai faux
Expliquez votre réponse.

(b). peut être instanciée : vrai faux
Expliquez votre réponse.

Question 4: *(2 pts)*

Soit le code ci-dessous.

```
1  public class Person {
2      private String name;
3
4      public Person(String name) {
5          this.name = name;
6      }
7
8      @Override
9      public boolean equals(Object obj) {
10         if (this == obj) {
11             return true;
12         }
13         if (!(obj instanceof Person)) {
14             return false;
15         }
16         Person other = (Person) obj;
17         return name.equals(other.name);
18     }
19 }
```

(a). A quoi sert la ligne 8 ?

(b). A quoi servent les lignes 10–12 ?

(c). A quoi servent les lignes 13–15 ?

(d). A quoi servent les lignes 16–17 ?

Question 5: *(3 pts)*

Développeur A écrit :

```
/**
Cette methode fait quelquechose.
@return Resultat de quelquechose. Normalement >= 0 ; sinon < 0 en cas de probleme.
*/
public int toqsik() {
    ...
}
```

tandis que développeur B livre :

```
/**
Cette methode fait quelquechose.
@return Resultat de quelquechose. Toujours >= 0.
@throws QuaboomException en cas de probleme.
*/
public int toqsik() throws QuaboomException () {
    ...
}
```

(a). Donnez des exemples de l'utilisation de la méthode `toqsik` selon chacune des approches.

A

B

(b). Quel sont les avantages / inconvénients de chaque approche ?

A

B

Question 6: *(2 pts)*

Soit le code ci-dessous.

```
public interface CanSwim {
    void swim();
}

public abstract class ActionHero implements CanSwim {
    void fight() {
        // fighting code
    }
}

public class YourTeacher extends ActionHero {
    void swim() {
        // swimming code
    }
}

public class Main {
    public static void main(String[] args) {
        CanSwim tehTeach = new YourTeacher();
        tehTeach.swim(); // hey! the teacher is swimming!
    }
}
```

Ce code fonctionne comme il le devrait : vrai faux
Expliquez votre réponse.

Question 7: *(3 pts)*

Découvert sur un blog[1] :

> I have just started programming professionally and a few days ago my coach walked in to see how I was doing. He saw that I used the keyword `instanceof` and told me that that is not the right thing to do, because it's not truly object-oriented. Someone tried to explain it to me but I still don't get it. I love `instanceof`, it always works ! Why can't I use it ?

(a). Quels sages conseils donneriez-vous à cet appel au secours ?

1. En français : `instanceof` a toujours marché pour moi, alors pourquoi me le déconseille-t-on ?

(b). Un exemple appuierait votre réponse.

Question 8: *(2 pts)*

Soit le code ci-dessous.

```
public class Foo {
    public static void classMethod() {
        System.out.println("classMethod() in Foo");
    }

    public void instanceMethod() {
        System.out.println("instanceMethod() in Foo");
    }
}

public class Bar extends Foo {
    public static void classMethod() {
        System.out.println("classMethod() in Bar");
    }

    public void instanceMethod() {
        System.out.println("instanceMethod() in Bar");
    }
}

public class Main {
    public static void main(String[] args) {
        Foo f = new Bar();
        f.instanceMethod();
        f.classMethod();
    }
}
```

(a). Quel est le résultat de l'exécution de `Main` ?

(b). Expliquez ce résultat.

Question 9: *(2 pts)*

Soit la classe `Person`

```
public class Person {
    private int age;
    private String name;

    public Person(String name, int age) {
        this.name = name;
        this.age = age;
    }

    // other methods omitted
}
```

et le code qui l'utilise

```
    Person person = new Person(name, age);
    // code
```

Il peut y avoir un problème dans le code si `name = ""` ou si `age = -32`.

(a). Comment faire pour tenir compte d'éventuels problèmes ?

(b). Le code pour le faire...