IFT1020 - Session Été, Intra

Mohamed Lokbani

IFT1020 - INTRA

Inscrivez tout de suite votre nom et code permanent.

Nom: _____| Prénom(s): _____|

Signature: _____| Code perm: _____|

Date : 14 juin 2005

Durée: 2 heures (de 17h30 à 19h30) Local: 1360 du Pavillon A.-AISENSTADT.

Directives:

- Toute documentation permise.
- Calculatrice **non** permise.

- Répondre directement sur le questionnaire.
- Les réponses **doivent être brèves, précises et clairement présentées.**

1._____/15 (1.1, 1.2, 1.3, 1.4)

2._____/10 (2.1)

3._____/10 (3.1)

4._____/20 (4.1, 4.2)

5._____/25 (5.1, 5.2)

6._____/20 (6.1, 6.2, 6.3, 6.4)

Total:_____/100

Question 1 (15 points)

1.1 Que signifient les termes "une variable d'instance" et "une méthode d'instance"?

1.2 Expliquer rapidement, quels sont les problèmes posés suite à l'utilisation de l'héritage multiple? Comment a-t-on résolu ces problèmes en Java?

1.3 Une applet peut définir la méthode suivante:

```
public boolean keyDown(Event evt, int key);
```

Quelle est l'utilité d'une telle méthode? À quoi sert l'argument « key »? Pourquoi cette méthode est-elle déclarée « public »?

1.4 Un des gros problèmes dans le développement de logiciels est la réutilisation d'un travail déjà réalisé. Expliquer brièvement comment la programmation par objets peut aider à solutionner ce problème de la réutilisation.

Question 2 (10 points) **Tout en justifiant votre réponse**, que va afficher en sortie le programme suivant qui compile et s'exécute correctement?

```
class A{}
class B extends A {}
class C extends B {}
class D extends C {}
public class MaClasse {
    public static void main(String args []) {

        B b = new C();
        A a = b;
        if (a instanceof A) System.out.println("A");
        if (a instanceof B) System.out.println("B");
        if (a instanceof C) System.out.println("C");
        if (a instanceof D) System.out.println("D");
        if (a instanceof Object) System.out.println("O");
    }
}
```

Question 3 (10 points) **Tout en justifiant votre réponse**, quels sont les constructeurs qui doivent exister dans la classe Base pour que le fragment de code suivant puisse compiler correctement ?

```
public class Test extends Base {
   public Test(int j) {
   }
   public Test(int j, int k) {
     super(j, k);
   }
}
```

Question 4 (20 points)

4.1 Un Compteur est un objet défini par les propriétés suivantes:

- il possède une valeur entière, positive ou nulle, nulle à sa création; qui ne peut varier que par pas de 1 (incrémentation).
- Il possède aussi, deux méthodes incremente et getValeur, pour respectivement incrémenter la valeur entière et afficher cette dernière en sortie.

Écrire la définition de la classe Compteur (y compris la définition des méthodes).

4.2 Écrire la méthode Question, sans arguments et sans valeur de retour, qui permet de réaliser ce qui suit :

- Lancer aléatoirement 100 fois une pièce de monnaie.
- En supposant que nous avons une chance sur deux d'avoir une des deux faces possibles (pile ou face avec donc 50/50 de chance) ; pour chaque tirage on met à jour les compteurs pour le coté pile et le coté face de la pièce, on utilise pour cela la classe Compteur de **6.1**.
- Après les 100 tirages, on affiche en sortie les résultats obtenus.
- Vous pouvez supposer que nous avons déjà importé dans le programme tous les paquetages nécessaires. Que la méthode Question se trouve dans une classe qui fait partie du même paquetage que la classe Compte (Compte est donc accessible) etc.

Question 5 (20 points) Soit le programme suivant qui compile et s'exécute correctement.

```
import java.awt.*;
import java.awt.event.*;
import java.applet.*;

public class Question extends Applet implements ActionListener{
    Button btnGrid = new Button("GridLayout(2,2)");
    Button btnGrid2 = new Button("GridLayout(3,3)");
    Button btnFlow = new Button("Flow");
    public  void init(){
        setLayout(new BorderLayout());
        unemethode();
    }
    public void unemethode(){
        btnGrid.addActionListener(this);
        btnGrid2.addActionListener(this);
        btnFlow.addActionListener(this);

        add(btnGrid,"North");
        add(btnGrid2,"East");
        add(btnFlow,"South");
    }
    public void actionPerformed(ActionEvent evt) {
        String arg =evt.getActionCommand();
        if(arg.equals("GridLayout(2,2)")){
            setLayout(new GridLayout(2,2));
            validate(); // permet de forcer l'apparition des changements
        }
        if(arg.equals("GridLayout(3,3)")){
            System.out.println(arg);
            setLayout(new GridLayout(3,3));
            validate();
        }
        if(arg.equals("Flow")){
            System.out.println(arg);
            setLayout(new FlowLayout());
            validate();
        }
    }
}
```

5.1 **Tout justifiant votre réponse**, dessiner l'applet de départ

5.2 <u>Tout justifiant votre réponse</u>, dessiner l'applet quand on clique sur le bouton btnGrid

5.3 **Tout justifiant votre réponse**, dessiner l'applet quand on clique sur le bouton btnGrid2

5.4 **Tout justifiant votre réponse**, dessiner l'applet quand on clique sur le bouton btnFlow

Question 6 (25 points) Indiquer si les println provoquent une erreur à la compilation. S'il y a erreur, vous devez la commenter. Dans le cas contraire, tout en expliquant votre réponse, vous devez donner l'affichage en sortie. À noter, que si un appel à println provoque une erreur de compile, cette dernière ne va pas planter le reste du programme. Après avoir expliqué l'origine de l'erreur, vous pouvez la considérer comme en commentaire pour le reste du programme. Ceci va vous permettre de répondre aux println restants.

```
interface I1{
    int m1();
}
interface I2 extends I1 {
    int m2();
}
class B implements I1 {
    int i=7;
    public int m1(){
        return i;
    }
}
class C extends B implements I2 {
    int i = -7;
    public int m2() {
        return i;
    }
    public int m1() {
        return i;
    }
}
public class Class1 {
    public static void main(String [] a) {

        B x = new C();
        System.out.println(x.i); // -1-
```

☐ correcte ☐ incorrecte

Pourquoi (affichage en sortie)

```
        System.out.println(x.m1()); // -2-
```

☐ correcte ☐ incorrecte

Pourquoi (affichage en sortie)

```
        System.out.println(x.m2()); // -3-
```

☐ correcte ☐ incorrecte

Pourquoi (affichage en sortie)

```
        I1 y = (I1) x;
        System.out.println(y.i); // -4-
```

☐ correcte ☐ incorrecte

Pourquoi (affichage en sortie)

```
        System.out.println(y.m1()); // -5-
```

☐ correcte ☐ incorrecte

Pourquoi (affichage en sortie)

```
        System.out.println(y.m2()); // -6-
```

☐ correcte ☐ incorrecte

Pourquoi (affichage en sortie)

```
        C z = (C) x;
        System.out.println(z.i); // -7-
```

☐ correcte ☐ incorrecte

```
        System.out.println(z.m1()); // -8-
```

☐ correcte ☐ incorrecte

Pourquoi (affichage en sortie)

```
        System.out.println(z.m2()); // -9-
```

☐ correcte ☐ incorrecte

Pourquoi (affichage en sortie)

```
        I2 v = (I2) x;

        System.out.println(v.i); // -10-
```

☐ correcte ☐ incorrecte

Pourquoi (affichage en sortie)

```
        System.out.println(v.m1()); // -11-
```

☐ correcte ☐ incorrecte

Pourquoi (affichage en sortie)

```
        System.out.println(v.m2()); // -12-
```

☐ correcte ☐ incorrecte

Pourquoi (affichage en sortie)

```
    }
}
```