QUESTIONS DE COURS JAVA/POO Session 1- Année 2015 – 2016

**Question 1**

La définition de la classe Rectangle est la suivante:

public class Rectangle {

private double hauteur, largeur;

public double surface( ){

return hauteur\*largeur;

}

//...

}

Ecrire les instructions qui permettent de créer une instance r de la classe Rectangle puis d'en appeler la méthode surface()?

|  |
| --- |
| Rectangle r = new Rectangle( );  **//affecter des valeurs à hauteur et largeur**  double surf = r.surface( ); |

**Question 2**

La définition de la classe Point est la suivante:

class Point{

private double x;

private double y;

public void setX(double newX){

x = newX;

}

public void setY(double newY){

y = newY;

}

public double getX(){ return x;}

public double getY(){ return y;}

}

Qu'affichera le programme suivant (cocher la case correspondante)?

Point p1 = new Point();

Point p2 = new Point();

p2.setX(2.5);

p1.setY(4.0);

System.out.println(p1.getX() + ", " + p1.getY());

System.out.println(p2.getX() + ", " + p2.getY());

|  |  |
| --- | --- |
| 2.5, 0.0  4.0, 0.0 |  |
| 2.5, 4.0  2.5, 4.0 |  |
| 0.0, 4.0  2.5, 0.0 |  |
| 2.5, 0.0  0.0, 4.0 | **oui** |
| erreur de compilation |  |
| pas d'affichage |  |

**Question 3 :** Les affirmations ci-dessous sont-elles correctes?

|  |  |
| --- | --- |
| Les affirmations | Correcte : Vrai/Faux? |
| L'héritage définit une relation "A-UN/POSSEDE-UN" |  |
| Une sous-classe hérite du type de sa super-classe. | **VRAI** |
| La classe parente a accès à tous les membres de ses sous-classes. |  |
| La classe parente connait toutes ses sous-classes. |  |

**Question 4**

Soient 3 classes A, B, C qui respectent les contraintes suivantes:

* La classe `C` est une sous-classe de la classe `B` ainsi que de la classe `A`.
* La classe `B` est une sous-classe de la classe `A`.

|  |
| --- |
| class B extends A { ... }  class C extends B extends A{ ... } |

Expliquez pourquoi la définition n'est pas correcte.

|  |
| --- |
| *Les dépendances circulaires ne sont pas permises. (mais ils ne vont pas répondre ça…)* |

**Question 5**

Qu'est-ce que la composition? (dites si les affirmations suivantes sont vraies/fausses?)

|  |  |
| --- | --- |
| Affirmations | Correcte : Vrai/Faux? |
| une structure de données |  |
| une méthode pour créer un objet |  |
| un lien du type "A-UN/POSSEDE-UN" | **VRAI** |
| doter une classe d'un attribut faisant référence à une autre classe | **VRAI** |
| rien de tout cela |  |

**Question 6**

La classe Livre est définie comme suit :

class Livre {

private int numPages;

public void setNumPages(int numPages){

numPages = this.numPages; bon sens ?

}

public void afficher(int numPages) {

if (numPages > 0) {

System.out.println(this.numPages + " pages");

}

else {

System.out.println("sans page");

}

}

}

Qu'affichera le programme suivant (choisir parmi les propositions données **celle** qui est correcte)?

Livre livre = new Livre();

livre.setNumPages(150);

livre.afficher(120);

|  |  |
| --- | --- |
| proposition | correcte? (oui/non) |
| 0 pages | *OUI méchant !* |
| 120 pages |  |
| sans page |  |
| erreur de compilation |  |
| pas d'affichage |  |
| 150 pages |  |

**Question 7**

class ClassA {

protected int number;

public ClassA() {

number = 10;

}

public void print() {

System.out.println(getPrefix() + ": " + number);

}

protected String getPrefix(){

return "A";

}

}

class ClassB extends ClassA{

protected int number = 20;

protected String getPrefix(){

return "B";

}

}

class ClassC extends ClassB{

public ClassC() {

super();

}

protected String getPrefix(){

return "C";

}

}

Qu'affichera le programme suivant (cocher parmi les propositions données **celle** qui est correcte)?

ClassC c = new ClassC();

c.print();

ClassB b = new ClassB();

b.print();

ClassA ab = new ClassB();

ab.print();

|  |  |  |  |
| --- | --- | --- | --- |
| proposition | correcte? | proposition | correcte? |
| C: 10  B: 10  B: 10 | BONNE REPONSE | A: 10  A: 10  A: 10 |  |
| C: 20  B: 20  B: 20 |  | C: 10  B: 10  A: 10 |  |

**Question 8**

Laquelle de ces définitions est valide? Cocher la case correspondante.

|  |  |  |  |
| --- | --- | --- | --- |
|  | valide? |  | valide? |
| class A {  private int attr = 1;  }  class B extends A {  private int c;  public B(int c){  this.c = c;  }  private doSth(){  super();  }  } |  | class A {  private int attr = 1;  }  class B extends A {  private int c;  public B(int c){  this.c = c;  super(c);  }  } |  |
| class A {  private int attr = 1;  }  class B extends A {  private int c;  public B(int c){  this.c = c;  }  } | BONNE REPONSE | class A {  private int attr = 1;  }  class B extends A {  private A attrA;  public B(int c){  attrA = super();  this.c = c;  }  } |  |

**Question 9**

List liste = new ArrayList();

for(int i = 0; i < 10; i++) {

liste.add(i);

}

Combien d'objets sont créés dans cette séquence ? Justifiez votre réponse.

Réponse : 1 objet liste et 10 objets "Integer" soit 11 objets.

(L'autoboxing permet de transformer automatiquement une variable de type primitif en un objet du type du wrapper correspondant.)

**Question 10:** Qu'affichera le programme suivant (cocher parmi les propositions données **celle** qui est correcte)?

List<Object> al = **new** ArrayList<Object>();

al.add(12);

al.add("Une chaîne de caractères !");

al.add(12.20f);

al.add('d');

for(int i = 0; i < al.size(); i++) {

System.out.println("donnée à l'indice " + i + " = " + al.get(i));

}

|  |  |
| --- | --- |
| proposition | correcte? (oui/non) |
| donnée à l'indice 0 = 12.0  donnée à l'indice 1 = U  donnée à l'indice 2 = 12.2  donnée à l'indice 3 = d |  |
| donnée à l'indice 0 = 12  donnée à l'indice 1 = Une chaîne de caractères !  donnée à l'indice 2 = 12.2  donnée à l'indice 3 = d | *OUI* |
| une erreur de compilation |  |
| pas d'affichage |  |
| donnée à l'indice 0 = 12  donnée à l'indice 2 = 12.2 |  |

**Question 11:** dites si les affirmations suivantes sont correctes ou non?

|  |  |
| --- | --- |
| Affirmations | correcte : Vrai/Faux? |
| Une collection permet de stocker un nombre variable d'objets. | **VRAI** |
| LinkedList et ArrayList sont des classes qui héritent de la classe List. | **FAUX** |
| Les Collection se redimensionnent dynamiquement au fur et à mesure des ajouts/suppressions d'objets. | **VRAI** |
| Si vous insérez fréquemment des données en milieu de liste, choisissez de préférence une LinkedList. | **VRAI** |
| Dans une Map, la clé et la valeur doivent être uniques. | **FAUX** |
| Les ensembles sont les seules collections d’éléments qui n’admettent pas les doublons. | **FAUX** |

**Question 12**

Qu'est-ce qu'une variable de classe? Quelles sont ses caractéristiques?

|  |
| --- |
| voir cours |

**Question 13**

La classe Circle est définie comme suit :

class Circle {

private double radius;

|  |  |
| --- | --- |
| proposition | correcte? (oui/non) |
| 0.0, 0.0 | oui |
| 0.0, 10.0 | non |
| 10.0, 0.0 | non |
| 10.0, 10.0 | non |
| une erreur de compilation | non |

public void setRadius(double r){

radius = r;

}

public double getRadius(){

return radius;

}

}

Qu'affichera le programme suivant?

Circle c1 = new Circle( );

Circle c2 = new Circle( );

c1.setRadius(10.0);

c1 = c2;

c2.setRadius(c1.getRadius());

System.out.println(c1.getRadius() + ", " + c2.getRadius());

**Question 14**

Dans l'instructionString string = new String ("string");

|  |  |
| --- | --- |
| **proposition** | **vrai/faux** |
| String est le nom de la variable |  |
| String est le type de la variable | *vrai* |
| string est le nom de la variable | vrai |
| string est le type de la variable |  |
| string est le nom de l’objet créé |  |
| String est une classe | vrai |

**Question 15:** Dites si les affirmations suivantes sont vraies ou fausses.

|  |  |
| --- | --- |
| Affirmation | Vrai/Faux ? |
| Une classe peut implémenter plusieurs classes mais ne peut étendre qu’une seule interface. | *Faux* |
| Une classe peut implémenter plusieurs classes et peut étendre plusieurs interfaces. | *Faux* |
| Une classe peut implémenter plusieurs interfaces mais ne peut étendre qu’une seule classe. |  |
| Une classe doit implémenter une seule interface et étendre une seule classe. | *Faux* |

**Question 16**

Le programme ci-dessous affiche un bouton dans une fenêtre. Ce bouton devrait changer de couleur en fonction des événements souris. Lire soigneusement le code et répondre aux questions posées.

public static void main(String[] args) {

Fenetre fen = new Fenetre() ;

fen.setVisible(true) ;

}

class Fenetre extends JFrame{

private JPanel pan = new JPanel();

private JButton bouton= new Bouton("Mon bouton", Color.blue);

public Fenetre(){

this.setTitle("Animation");

this.setSize(300, 300);

this.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

this.setLocationRelativeTo(null);

pan.add(bouton);

this.addMouseListener((MouseListener) bouton );

this.setContentPane(pan);

this.setVisible(true);

}

}

class Bouton extends JButton implements MouseListener{

private Color color ;

public Bouton(String str, Color color){

super (str);

this.color= color;

}

public void paintComponent(Graphics g){

super.paintComponent(g);

this.setBackground(color);

}

public void mouseClicked(MouseEvent event) {

}

public void mouseEntered(MouseEvent event) {

// fond en jaune lors du survol

color= Color.yellow;

repaint();

}

public void mouseExited(MouseEvent event) {

// fond en vert lorsqu'on quitte le bouton

color = Color.green;

repaint();

}

public void mousePressed(MouseEvent event) {

// fond en rouge lors du clic gauche

color = Color.red;

repaint();

}

public void mouseReleased(MouseEvent event) {

// fond en bleu lorsqu'on relâche le clic

color = Color.blue;

repaint();

}

}

1. Pourquoi le résultat n’est pas celui attendu ?

|  |
| --- |
| aucun évènement n'est écouté. |

1. Quelles corrections sont nécessaires? A quel endroit du code?

|  |
| --- |
| addActionListener( ) dans le constructeur (plusieurs solutions). |

1. Pourquoi faut-il écrire à vide la méthode mouseClicked() ?

|  |
| --- |
| parce que toutes les méthodes de l'interface doivent être implémentées pour que la classe puisse être concrète. |