HEI
INGÉNIEURS POUR LE MONDE

yncréa

Année universitaire 2018 - 2019

Epreuve de : Conception Logicielle

Code module : ITI063

Groupe de Matière : HEI4 ITI

Date : Lundi 17/12/2018 de 08h00 à 10h00

Conditions : Sans documents et sans calculatrice (SDSC)

Nombre de pages du sujet : 4 pages

Remarque :

Conception Logicielle

Examen

Durée : 2h00

Session : Décembre 2018

Remarque:

- Document(s), calculatrice et téléphone interdits
- Ce document comprend 4 pages.
- Une attention particulière sera portée sur la lisibilité, la rédaction et la pertinence des réponses

- **Le barème est donné à titre indicatif**

Exercice 1 : (questions de cours) (6pts)

1- Veuillez expliquer le principe d'encapsulation ?

2- Quel est le rôle de la gestion des exceptions ?

3- Quelle est la différence entre une méthode d'instance et une méthode de classe ?

4- Quelle est la différence entre une classe et une interface ?

5- Un même référent peut-il désigner plusieurs objets ? Plusieurs référents peuvent-ils désigner un même et seul objet ?

6- Un objet peut-il faire référence à un autre ? si oui, comment ?

7- Quelle est la différence entre :

   a. Point p[][] = new Point[5][4] ;

   b. Point p = new Point(5,4) ;

Exercice 2 : Analyse de code (4pts)

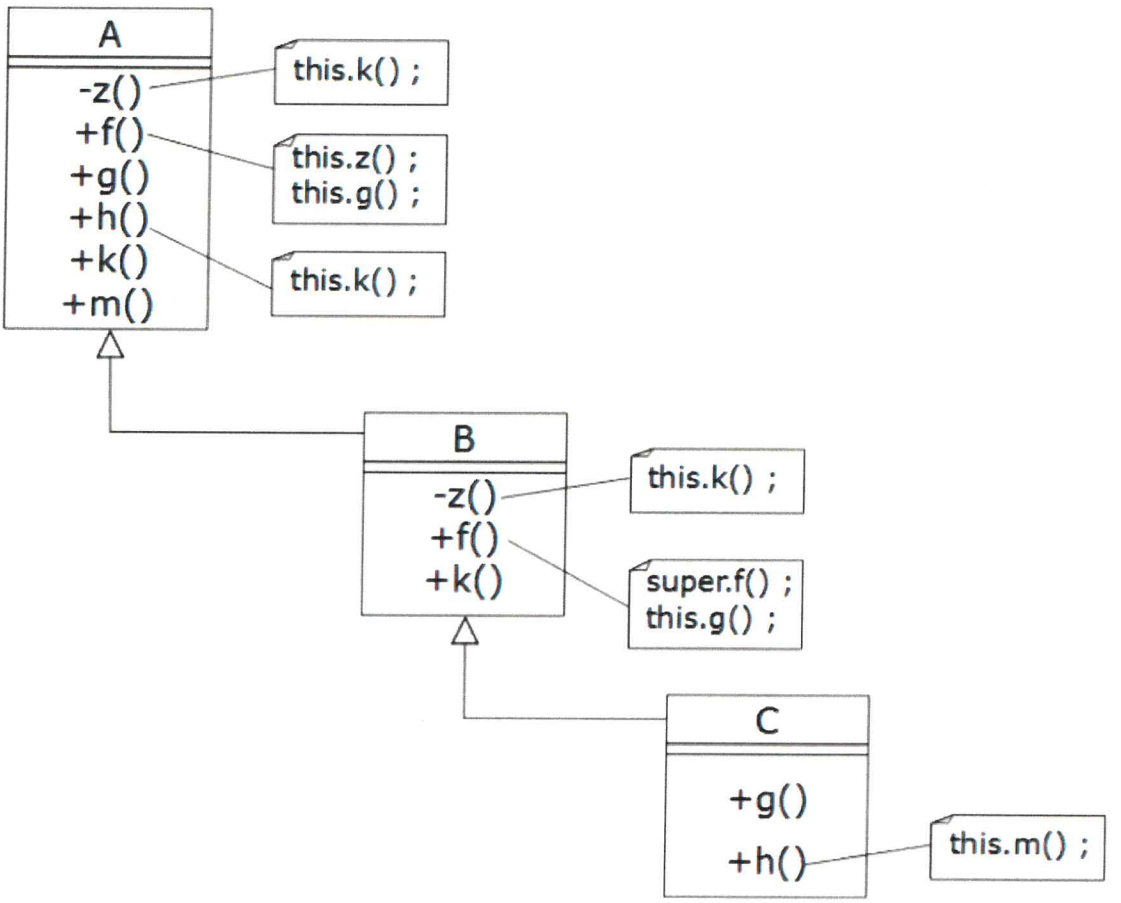
```
public class Parent {
      int x;
      Parent(int k) {x=k;}
      int ajoute(int a) { return x+a; }
      public void moi() { System.out.println(" x = "+ x); }
}

public class Enfant1 extends Parent {
      int y;
      Enfant1 (int k, int l) {
            super(k);
            y=l; }
     int ajoute(int a) { return x+2*a;}
}

public class Enfant2 extends Enfant1 {
      int z ;
      Enfant2 (int k, int l, int m) {
            super(k, l);
            z= m; }
      int ajoute(int a) { return x+3*a;}
      public void moi() {
            super.moi();
            System.out.println(" z = "+ z);
      }
}

public class Test{
      public static void main (String args[]) {
            int a =2;
            Parent p = new Parent(3);
            p.moi();
            System.out.println(" ajoute("+ a +") = "+ p.ajoute(a) );
```

```
            Enfant1 e1 = new Enfant1(3, 4);
            e1.moi();
            System.out.println(" ajoute("+ a +") = "+ e1.ajoute(a) );
            e1 = new Enfant2(3, 4, 5);
            e1.moi();
            System.out.println(" ajoute("+ a +") = "+ e1.ajoute(a) );
        }
}
```

1. **Ecrivez le résultat de l'exécution de la classe Test.**

Exercice 3 (5pts)

On donne le diagramme de classes suivant :



En plus des portions de code indiquées sur le diagramme, chacune des méthodes commence par l'instruction : « System.out.println("NomDeClasse.nomMéthode "); » ou NomDeClasse est évidemment remplacé par le nom de la classe dans laquelle le corps de la méthode est codée et nomMéthode par le nom de cette méthode [1].

1. Indiquez **précisément** ce qu'affiche le programme suivant (à chaque itération):

[1] L'exécution de « new A().k() » produit donc l'affichage « A.k »

```
public class Exo2ExamTracesExecustion {
	public static void main(String[] args) {
		ArrayList<A> listeA = new ArrayList<A>();
		listeA.add(new A());
		listeA.add(new B());
		listeA.add(new C());
		for(A a : listeA) {
			System.out.println("--- appel de h()---");
			a.h();
			System.out.println("--- appel de f()---");
			a.f();}
		}}
```

Exercice 4 : (5 pts)

Une interface *Compte* est donnée en annexe. Il existe différents types de comptes : des comptes chèques, des comptes ´épargne et des plans d'épargne. A chacun de ces types correspond une classe implémentant l'interface Compte.

Ces classes sont respectivement les classes **CompteCheque**, **CompteEpargne** et **PlanEpargne** du paquetage banque.compte. Les instances des classes CompteEpargne et PlanEpargne disposent d'informations supplémentaires telles qu'un taux d'intérêt, une date d'échéance, etc. On suppose ces classes définies.

Les instances de la classe CompteCheque sont quant à elles caractérisées par un attribut autorisationDecouvert dont la valeur peut être changée par le banquier si besoin. Cette valeur (négative) correspond au découvert autorisé sur ce compte. La méthode addEcriture de cette classe lève l'exception mentionnée dans la signature lorsque l'on essaie d'effectuer une écriture sur le compte qui amènerait `a un solde inférieur à la valeur de autorisationDecouvert. Cette écriture n'est dans ce cas pas ajoutée.

Donnez le code complet en java de la classe CompteCheque (un numéro est fourni en paramètre du constructeur).

Annexe : L'interface Compte

```
package banque.compte;
import java.util.*;
public interface Compte {
	// retourne le numéro du compte
	public String getNumeroCompte();
	// retourne le solde de ce compte
	public float getSolde();
	// retourne la liste des écritures pour le compte
	public List getEcritures();
	// ajoute une ecriture pour ce compte
	public void addEcriture(Ecriture e) throws UnsupportedOperationException;
	// retourne la liste des écritures du compte des <nbJours> jours
	// précédant <d> (inclus)
	public List ecrituresDepuis(util.date.Date d, int nbJours);  }
```