EPF – Lausanne *COURS DE MATHEMATIQUES SPECIALES*

9 janvier 2013

# Contrôle d'informatique no 2

Durée : 1 heure 45'

Nom : …… Groupe : ☐

Prénom : ……

| No | 1 | 2 | 3 |
|---|---|---|---|
| Total points | 55 points (40 + 15) | 34 points | 21 points |
| | | | |

***Remarque générale : toutes les questions qui suivent se réfèrent au langage de programmation Java (à partir du JDK 5.0) et les réponses doivent être rédigées à l'encre et d'une manière propre sur ces feuilles agrafées.***

**Sujet no 1.**

Le but de cet exercice est de réaliser une application autonome interactive qui aide l'utilisateur à traiter des problèmes liés à la gestion des comptes bancaires. Cette application doit correspondre à un projet doté d'un package appelé **cms_ctr2** et qui contient deux classes **publiques**, définies dans deux fichiers distincts et appelées **Compte** et, respectivement, **CP_Ctr2Exo1**. Par la suite, aux points **1.1** et **1.2**, on vous demande d'écrire le code complet de ces deux classes, en respectant les consignes précisées. Vous pouvez éventuellement répondre au point **1.2** en supposant le point **1.1** résolu correctement.

**1.1** Une instance de la classe **Compte** est un objet de type **Compte** qui regroupe (ou encapsule) des informations concernant un compte bancaire, à savoir le montant du compte et le mot de passe associé. De plus, la classe **Compte** est munie des méthodes permettant de manipuler les objets de type **Compte**.

Écrire le code complet de la classe **publique Compte** qui appartient au package **cms_ctr2** et qui définit :

- **a)** un champ (d'instance) nommé **montant** de type numérique réel, déclaré **privé** et sans valeur initiale explicite et qui sert à stocker la valeur du montant du compte ;
- **b)** un champ (d'instance) nommé **password** de type numérique entier, déclaré **privé** et sans valeur initiale explicite et qui sert à stocker le mot de passe qui donne accès au compte ;
- **c)** un champ nommé **nbComptes** de type entier, déclaré sans modificateur d'accès et avec la valeur initiale ***0*** et qui sert à compter les comptes bancaires créés ; par conséquent, la valeur de ce champ ne doit pas être stockée dans chaque instance de la classe **Compte** ;
- **d)** un constructeur **public** (surchargé) sans argument qui incrémente la valeur du champ **nbComptes** d'une unité ;
- **e)** un constructeur **public** (surchargé) avec deux arguments : le premier de type numérique réel et le deuxième de type numérique entier ; ce constructeur permet de créer un nouveau compte (en indiquant comme premier argument le montant initial et comme deuxième argument le mot de passe) et doit respecter les consignes suivantes :
  - **i.** il appelle le constructeur sans argument ;
  - **ii.** si la valeur de son premier argument est strictement positive, il stocke cette valeur dans le champ **montant** ;
  - **iii.** il stocke la valeur de son deuxième argument dans le champ **password** ;
- **f)** une méthode **publique d'instance** nommée **donnerMontant** qui a un argument de type numérique entier et retourne une valeur de type numérique réel ; cette méthode permet de consulter le montant d'un compte bancaire (en indiquant son mot de passe en tant qu'argument) et doit respecter les consignes suivantes :
  - **i.** si la valeur de son argument est égale au mot de passe correspondant à l'objet appelant, la méthode retourne la valeur du montant du compte correspondant à l'objet appelant ;
  - **ii.** autrement, la méthode affiche dans la fenêtre console le message **Mot de passe non valide !** et retourne la valeur **-1** ;

**g)** une méthode **publique d'instance** nommée **deposer** qui a un argument de type numérique réel et retourne une valeur de type logique ; cette méthode permet de déposer dans un compte bancaire une somme d'argent indiquée comme argument et doit respecter les consignes suivantes :

- **i.** si la valeur de son argument est négative ou nulle, la méthode affiche dans la fenêtre console le message **Dépôt non valide !** et retourne la valeur logique **faux** ;
- **ii.** autrement, la méthode ajoute la valeur de son argument à la valeur du champ **montant** de l'objet appelant et retourne la valeur logique **vrai** ;

**h)** une méthode **publique** et **statique** nommée **retirer** qui permet de retirer de l'argent d'un compte bancaire ; par conséquent, cette méthode retourne une valeur de type numérique réel et doit respecter les consignes suivantes :

- **i.** elle a trois arguments, à savoir : le premier de type **Compte** (qui correspond au compte bancaire), le deuxième de type numérique réel (qui correspond à la somme retirée) et le troisième de type numérique entier (qui correspond au mot de passe) ;
- **ii.** si la valeur du troisième argument n'est pas égale au mot de passe correspondant au compte indiqué par le premier argument, la méthode affiche le message **Mot de passe non valide !** et retourne la valeur **-1** ;
- **iii.** (autrement) si la valeur du deuxième argument est négative ou nulle, la méthode affiche le message **Retrait non valide !** et retourne la valeur **-2** ;
- **iv.** (autrement) si la valeur du deuxième argument est strictement plus grande que la valeur du **montant** correspondant au compte indiqué par le premier argument, la méthode affiche le message **Montant insuffisant !** et retourne la valeur **-3** ;
- **v.** (autrement) la méthode diminue le **montant** correspondant au compte indiqué par le premier argument avec la valeur de son deuxième argument et retourne la valeur de son deuxième argument.

***Remarque** : Commencez la rédaction de votre réponse à la page suivante, s'il vous plait.*

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

**1.2** En fonction des indications données comme commentaires, compléter ci-dessous le code de la classe principale **publique CP_Ctr2Exo1** qui appartient au package **cms_ctr2** et qui contient la méthode **main**. Cette classe utilise la classe **Compte** disponible dans le même package.

```
//déclarez le package
.....
.....
.....
//précisez l'en-tête de la classe principale
.....
.....
.....
{     //précisez l'en-tête de la méthode main
.....
.....
.....
      {     //créez un nouvel objet de type Compte qui correspond
            //à un compte bancaire, ayant le montant initial 1000 SFR
            //et le mot de passe 123456, et stockez son adresse dans
            //une variable locale déclarée à cet effet et nommée c
.....
.....
.....
            //déposez dans le compte la somme de 250 SFR
.....
.....
.....
            //retirez du compte la somme de 300 SFR
.....
.....
.....
            //affichez à l'écran le montant actuel du compte
.....
.....
.....
      }//fin de la méthode main
}//fin de la classe principale
```

**Sujet no 2.**

Préciser les messages qui seront affichés à l'écran suite à l'exécution du projet contenant les deux classes suivantes (qui appartiennent à un même package).

```
package cms_ctr2;

class Couple
{
	private int premier;
	private int second;
	private static int n=0;

	public Couple()
	{
		this(0);
		System.out.println("On bâtit un Couple !");
	}

	public Couple(int nb)
	{
		this(nb, nb+1);
		System.out.println("On fabrique un Couple !");
	}

	public Couple(int a, int b)
	{
		if(n++ < 3)
		{
			System.out.println("On construit un Couple !");
		}
		if(a <= b)
		{
			premier = a;
			second = b;
		}else
		{
			premier = b;
			second = a;
		}
	}
```

```
	public void setPremier(int premier)
	{	if(premier <= this.second)
			this.premier = premier;
		else
		{
			this.premier = this.second;
			this.second = premier;
		}
	}

	public void setSecond(int second)
	{	if(second <= this.premier)
		{
			this.second = this.premier;
			this.premier = second;
		}else
			this.second = second;
	}

	public static Couple addCross(Couple c1, Couple c2)
	{	Couple c =  new Couple();
		c.setPremier(c1.premier + c2.second);
		c.setSecond(c1.second + c2.premier);
		return c;
	}

	public Couple subtract(Couple c2)
	{	Couple c =  new Couple();
		c.setPremier(this.premier - c2.premier);
		c.setSecond(this.second - c2.second);
		return c;
	}

	public void afficher()
	{	System.out.println("Je suis le couple : ("
					+ premier + ", " + second + ").");
	}
}//fin de la classe Couple
```

```
public class CP_Ctr2Exo2
{
	public static void main(String[] args)
	{
		Couple c1 = new Couple(3, -5);
		System.out.println("-------------------------");
		Couple c2 = new Couple(10);
		System.out.println("-------------------------");
		Couple c3 = new Couple ();
		System.out.println("*****************************");
		c1.afficher();
		c2.afficher();
		c3.afficher();
		System.out.println("***** On mélange *****");
		c3 = c2;
		c2 = c1;
		c1.afficher();
		c2.afficher();
		c3.afficher();
		System.out.println("***** On change le premier *****");
		Couple c4 = new Couple(1, -2);
		c4.setPremier(4);
		c4.afficher();
		System.out.println("***** On change le deuxième *****");
		Couple c5 = new Couple(1, -2);
		c5.setSecond(-10);
		c5.afficher();
		System.out.println("***** On additionne en croix *****");
		Couple c7 = new Couple(-1, 2);
		Couple c8 = new Couple(-3, 1);
		Couple c9 = Couple.addCross(c7, c8);
		c9.afficher();
		System.out.println("***** On soustrait *****");
		Couple c10 = new Couple(-1, 2);
		Couple c11 = new Couple(-3, 1);
		Couple c12 = c10.subtract(c11);
		c12.afficher();
	}//fin de la méthode main
}//fin de la classe principale
```

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

**Sujet no 3.**

Préciser les messages qui seront affichés à l'écran suite à l'exécution du projet contenant les deux classes suivantes (qui appartiennent à un même package).

***Remarque*** *: Dans le code ci-dessous :*

- *l'opérateur / correspond à la division entière ;*
- *l'opérateur % est l'opérateur modulo.*

```
package cms_ctr2;

class Auxiliaire
{
	public static final int TAB[] = {1000, 200, 100, 50, 20, 10};

	public static int[] compter(int montant)
	{
		int tabRetour[] = new int[TAB.length];

		for(int i=0; i<TAB.length; i++)
		{
			tabRetour[i] = montant / TAB[i];
			montant = montant % TAB[i];
		}//fin de la boucle for

		return tabRetour;

	}//fin de la méthode compter

}//fin de la classe Auxiliaire
```

```java
public class CP_Ctr2Exo3
{
	public static void main(String[] args)
	{
		int tabMontants[] = {1983, -2013, 790};

		System.out.println("Bonjour !");
		System.out.println("************************");

		for(int j=0; j<tabMontants.length; j++)
		{
			if(tabMontants[j]<=0)
			{
				System.out.println("Valeur non valide !");
				System.out.println("************************");
				continue;
			}

			int tabElements[] =
					Auxiliaire.compter(tabMontants[j]);

			System.out.println(tabMontants[j] + " :");

			for(int k=0; k<tabElements.length; k++)
			{
				System.out.println(tabElements[k] + "x"
						+ Auxiliaire.TAB[k]);
			}//fin de la boucle for interne

			System.out.println("************************");
		}//fin de la boucle for externe

		System.out.println("Au revoir !");
	}//fin de la méthode main
}//fin de la classe principale
```

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....