EPF – Lausanne *COURS DE MATHEMATIQUES SPECIALES*

15 janvier 2014

# Contrôle d'informatique no 2

Durée : 1 heure 45'

Nom : ……………………………………………… Groupe : ☐
Prénom : ………………………………………………

| No | 1.1 | 1.2 | 2 |
|---|---|---|---|
| Total points | 55 points | 36 points | 39 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 de mécanique élémentaire faisant intervenir des poulies idéales (dont les moments d'inertie sont négligeables). 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 **Poulie** 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 **Poulie** est un objet de type **Poulie** qui regroupe (ou encapsule) des informations concernant une poulie, à savoir les valeurs des deux masses **m1** et **m2** associées (voir Figure 1). De plus, la classe **Poulie** est munie des méthodes permettant de manipuler les objets de type **Poulie**.

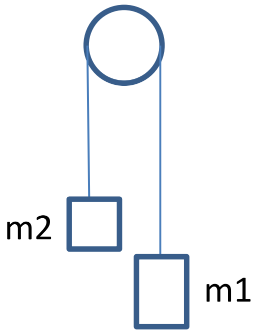

Figure 1

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

- **a)** un champ (d'instance) nommé **m1** de type numérique réel, déclaré **privé** et sans valeur initiale explicite et qui sert à stocker la valeur de la masse m1 ;
- **b)** un champ (d'instance) nommé **m2** de type numérique réel, déclaré **public** et sans valeur initiale explicite et qui sert à stocker la valeur de la masse m2 ;
- **c)** une méthode publique (d'instance) "**setter**" nommée en respectant la convention Java pour le nommage des setters et qui permet la modification de la valeur du champ privé **m1** ; cette méthode doit respecter les consignes suivantes :
    - **i.** si la valeur de son argument noté lui aussi ***m1*** est strictement positive, on copie la valeur de cet argument dans le champ **m1**;
    - **ii.** autrement, on stocke dans le champ **m1** la valeur **1** ;
- **d)** une méthode publique (d'instance) "**getter**" nommée en respectant la convention Java pour le nommage des getters et qui retourne (sans aucune vérification) la valeur du champ privé **m1** ;
- **e)** un constructeur **public** (surchargé) avec deux arguments de type numérique réel : le premier argument nommée ***m1*** (comme le champ **m1**) et le deuxième argument nommé ***m2*** (comme le champ **m2**) ; ce constructeur permet de créer un objet correspondant à une nouvelle poulie (en proposant comme premier argument la valeur de la masse m1 et comme deuxième argument la valeur de la masse m2) et doit respecter les consignes suivantes :

    i. la valeur de son premier argument ***m1*** est "stockée" dans le champ **m1** par un appel à la méthode setter correspondante ;
    ii. si la valeur de son deuxième argument ***m2*** est strictement positive, on copie la valeur de cet argument dans le champ **m2** ;
    iii. autrement, on stocke dans le champ **m2** la valeur **1** ;

**f)** un constructeur **public** (surchargé) sans argument qui stocke la valeur **2** dans le champ **m1** (du nouvel objet) et la valeur **1** dans le champ **m2** (du nouvel objet) par un appel adéquat du constructeur avec deux arguments ;

**g)** une méthode **publique d'instance** nommée **calculerAccPlus** qui n'a pas d'argument et retourne une valeur numérique réelle ; cette méthode permet de calculer la valeur absolue de l'accélération du système mécanique associé à la poulie correspondant à l'objet appelant et doit respecter les consignes suivantes :

    i. si la valeur du champ **m1** de l'objet appelant est plus grande ou égale à la valeur du champ **m2** de l'objet appelant, la méthode retourne la valeur de l'expression **(m1-m2)*10/(m1+m2)** ;
    ii. autrement, la méthode retourne la valeur de l'expression **(m2-m1)*10/(m1+m2)** ;

**h)** une méthode **publique** et **statique** nommée **calculerSens** qui permet de déterminer le sens du mouvement du système mécanique associé à la poulie correspondant à l'argument de la méthode, conformément à une convention expliquée ci-dessous ; plus précisément, cette méthode a un seul argument de type **Poulie**, retourne une valeur numérique entière et doit respecter les consignes suivantes :

    i. si la valeur du champ **m1** de l'objet argument est strictement plus grande que la valeur du champ **m2** de l'objet argument, la méthode retourne la valeur **1** ;
    ii. (autrement) si la valeur du champ **m1** de l'objet argument est égale à la valeur du champ **m2** de l'objet argument, la méthode retourne la valeur **0** ;
    iii. (autrement si la valeur du champ **m1** de l'objet argument est strictement plus petite que la valeur du champ **m2** de l'objet argument), la méthode retourne la valeur **-1** ;

**i)** une méthode **publique d'instance** nommée **changerSens** qui n'a pas d'argument et ne retourne pas de valeur ; cette méthode doit interchanger les valeurs des champs **m1** et **m2** de l'objet appelant (la nouvelle valeur du champ **m1** doit devenir l'ancienne valeur du champ **m2** et la nouvelle valeur du champ **m2** doit devenir l'ancienne valeur du champ **m1**) ;

**j)** une méthode **publique d'instance** nommée **créerClone** qui doit créer un clone de l'objet appelant et retourner l'adresse de ce clone ; plus précisément, cette méthode :

   **i.** crée un nouvel objet de type **Poulie** en assurant que les valeurs de ses deux champs **m1** et **m2** soient respectivement égales aux valeurs des champs correspondants de l'objet appelant ;

   **ii.** retourne l'adresse du nouvel objet créé ;

**k)** une méthode **publique** et **statique** nommée **comparer** qui permet de comparer les valeurs absolues des accélérations des systèmes mécaniques associés à deux poulies ; plus précisément, cette méthode a deux arguments de type **Poulie**, retourne une valeur numérique entière et doit respecter les consignes suivantes :

   **i.** si la valeur absolue de l'accélération du système mécanique associé à la poulie correspondant au premier argument est strictement plus grande que la valeur absolue de l'accélération du système mécanique associé à la poulie correspondant au deuxième argument, la méthode retourne la valeur **1** ;

   **ii.** (autrement) si la valeur absolue de l'accélération du système mécanique associé à la poulie correspondant au premier argument est égale à la valeur absolue de l'accélération du système mécanique associé à la poulie correspondant au deuxième argument, la méthode retourne la valeur **0** ;

   **iii.** (autrement  si la valeur absolue de l'accélération du système mécanique associé à la poulie correspondant au premier argument est strictement plus petite que la valeur absolue de l'accélération du système mécanique associé à la poulie correspondant au deuxième argument), la méthode retourne la valeur **-1** ;

**l)** une méthode **publique d'instance** nommée **afficher** qui n'a pas d'argument et retourne une valeur de type chaîne de caractères qui correspond au message suivant ***Une poulie de masse m1=XXX et de masse m2=YYY !*** où ***XXX*** est la valeur du champ **m1** de l'objet appelant et ***YYY*** est la valeur du champ **m2** de l'objet appelant.

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

…………………………………………………………………………………………………………………

…………………………………………………………………………………………………………………

…………………………………………………………………………………………………………………

…………………………………………………………………………………………………………………

…………………………………………………………………………………………………………………

…………………………………………………………………………………………………………………

…………………………………………………………………………………………………………………

…………………………………………………………………………………………………………………

…………………………………………………………………………………………………………………

…………………………………………………………………………………………………………………

…………………………………………………………………………………………………………………

…………………………………………………………………………………………………………………

…………………………………………………………………………………………………………………

…………………………………………………………………………………………………………………

…………………………………………………………………………………………………………………

…………………………………………………………………………………………………………………

…………………………………………………………………………………………………………………

…………………………………………………………………………………………………………………

…………………………………………………………………………………………………………………

…………………………………………………………………………………………………………………

…………………………………………………………………………………………………………………

…………………………………………………………………………………………………………………

…………………………………………………………………………………………………………………

…………………………………………………………………………………………………………………

…………………………………………………………………………………………………………………

…………………………………………………………………………………………………………………

…………………………………………………………………………………………………………………

…………………………………………………………………………………………………………………

…………………………………………………………………………………………………………………

…………………………………………………………………………………………………………………

…………………………………………………………………………………………………………………

…………………………………………………………………………………………………………………

…………………………………………………………………………………………………………………

…………………………………………………………………………………………………………………

**1.2** Ecrire le ***code complet*** de la classe publique **CP_Ctr2Exo1** qui appartient au package **cms_ctr2** et qui contient la méthode **main**. Cette classe réalise la partie interactive du projet (voir aussi l'exemple d'affichage à l'exécution présenté plus loin) et utilise la classe **Poulie**.

Plus précisément, dans la méthode **main** :

**a)** on déclare (et, le cas échéant, on initialise) les variables dont on a besoin pour réaliser le dialogue avec l'utilisateur ;

**b)** on déclare deux variables (locales) nommées **m1** et **m2** de type numérique réel (sans valeurs initiales explicites) ;

**c)** on déclare une variable (locale) nommé **p** de type **Poulie** et on y stocke l'adresse d'un nouvel objet créé à l'aide du constructeur sans argument de la classe **Poulie** ;

**d)** on affiche le message **Le but du programme est …** (qui remplace un message plus complet contenant des précisions concernant l'emploi du programme par l'utilisateur) ;

**e)** on affiche à l'écran un message qui demande à l'utilisateur d'introduire la valeur de la masse m1 de la poulie ;

**f)** on récupère la valeur introduite par l'utilisateur et on la stocke dans la variable (locale) **m1** prévue à cet effet ;

**g)** on affiche à l'écran un message qui demande à l'utilisateur d'introduire la valeur de la masse m2 de la poulie ;

**h)** on récupère la valeur introduite par l'utilisateur et on la stocke dans la variable (locale) **m2** prévue à cet effet ;

**i)** on "copie" la valeur de la variable (locale) **m1** dans le champ privé **m1** de l'objet référencé par la variable objet **p**, par un appel à la méthode "setter" correspondante ;

**j)** si la valeur de la variable (locale) **m2** est strictement positive, on "copie" cette valeur dans le champ public **m2** de l'objet référencé par la variable objet **p** (par une affectation habituelle) ;

**k)** autrement (c'est-à-dire si la valeur de la variable (locale) **m2** est négative ou nulle), on "copie" la valeur **1** dans le champ public **m2** de l'objet référencé par la variable objet **p** (par une affectation habituelle) ;

**l)** on déclare une variable (locale) nommée **c** de type **Poulie** et on y stocke l'adresse d'un clone de l'objet référencé par la variable objet **p**, clone créé pour l'occasion par un appel adéquat de la méthode **creerClone** ;

**m)** on affiche à l'écran des informations concernant l'objet référencé par la variable objet **c** (et ces information sont obtenues suite à un appel adéquat de la méthode **afficher**) ;

**n)** on déclare une variable (locale) nommée **code** dont le type permet d'y stocker des lettres (sans préciser une valeur initiale pour cette variable) ;

**o)** on demande (éventuellement plusieurs fois) à l'utilisateur d'introduire le code de l'opération à effectuer (en supposant que l'utilisateur sait déjà quelle signification aura sa réponse pour le programme et cette signification sera expliquée ci-dessous) ;

**p)** on stocke la lettre introduite par l'utilisateur (au clavier) dans la variable **code** prévue à cet effet ;

**q)** <u>à l'aide d'une instruction structurée **switch**</u>, on prévoit les choix suivants:

  **i.** si la valeur de la variable **code** est la lettre **a**, on affiche à l'écran la valeur absolue de l'accélération du système mécanique associé à la poulie correspondant à la variable objet **p** (grâce à un appel adéquat de la méthode **calculerAccPlus**) et on revient ensuite au point **o)** ;

  **ii.** si la valeur de la variable **code** est la lettre **b**, on change le sens de la poulie correspondant à la variable objet **p** (par un appel adéquat à la méthode **changerSens**) et on revient ensuite au point **o)** ;

  **iii.** si la valeur de la variable **code** est la lettre **c**, on affiche à l''écran le résultat retourné par un appel de la méthode **comparer**, appel qui a comme but de comparer les valeurs absolues des accélérations des poulies correspondant aux variables objets **c** et **p** (dans cet ordre) et on revient ensuite au point **o)** ;

  **iv.** si la valeur de la variable **code** est n'importe quel autre caractère (différent des lettres **a**, **b** et **c**), on ne revient plus au point **o)** et on passe au point suivant **r)** ;

**r)** on affiche (une seule fois) le message **Au revoir !** et le programme s'arrête (car il arrive à sa fin).

A l'exécution, la sortie du programme doit respecter la mise en page donnée dans l'exemple ci-dessous.

```
Le but du programme est ...
Introduisez la masse m1 de la poulie :
3
Introduisez la masse m2 de la poulie :
-1
Une poulie de masse m1=3.0 et de masse m2=1.0 !
Introduisez le code de l'opération :
b
Introduisez le code de l'opération :
a
5.0
Introduisez le code de l'opération :
c
0
Introduisez le code de l'opération :
a
5.0
Introduisez le code de l'opération :
q
Au revoir !
```

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

**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 Babel
{    static final char tab[] = {'d','e','f','i','r'};
     char source;            char cible;
     static int i=33;        static boolean flag;

     Babel( )
     {    this('e', 'f');
          if(i==34 || i==35 )
               System.out.println("On interprète !");}

     Babel(int i1, int i2)
     {    this(tab[i1 % 5], tab[i2 %  5]);
          if(i>33 && i<36)
               System.out.println("On déchiffre !");}

     private Babel(char langue1, char langue2)
     {    if(i==33 || i==34 || flag==true)
               System.out.println("On traduit !");
          source = langue1;    cible = langue2;
          i++;}

     static Babel composer(Babel b1, Babel b2)
     {
          flag = true;
          Babel b = new Babel(b1.source, b2.cible);
          flag = false ;
          return b;}

     void changer(Babel b, int i)
     {    if(i % 2 == 0)
          {    source = b.source;
               b.cible = cible;
          }else
          {    b.source = source;
               cible = b.cible;
          }
     }

     void babeliserEnCroix(Babel b, int i)
     {    if(source=='f'  || b.source=='i')
          {    source=tab[i %5 ];
               b.source=tab[(i+1) % 5];
          }
          if(b.cible=='i'  && cible=='r')
          {    b.cible=tab[i % 5];
               cible=tab[(i+2) % 5];
```

```
		}else
		{	cible=tab[(i+3)%5];
			b.cible=tab[(i+2)%5];
		}
	}

	void babeliserDirectement(Babel b, int i)
	{	if(source=='f' || source=='i')
		{	source=tab[i % 5];
			b.source=tab[(i+1) % 5];
		}
		if(cible=='i' && cible=='r')
		{	cible=tab[(i+3) % 5];
			b.cible=tab[(i+2) % 5];
		}else
		{	changer(b, i);
		}
	}

	void montrer()
	{	System.out.println("Traduction : " + source + " vers "
							+ cible + ".");}
}//fin de la classe Babel

public class CP_Ctr2Exo2 {
	public static void main(String[] args)
	{	System.out.println("***CREER***");
		Babel b1 = new Babel(7,1); Babel b2 = new Babel();
		b1.montrer();			b2.montrer();
		System.out.println("***MELANGER***");
		b1 = b2;		b2 = b1;
		b1.montrer();	b2.montrer();
		System.out.println("*** COMPOSER***");
		Babel b3 = new Babel(2,1); Babel b4 = new Babel(0,3);
		b3.montrer();			b4.montrer();
		Babel b5 = Babel.composer(b3, b4);
		b5.montrer();
		System.out.println("***CHANGER***");
		b3.changer(b4, 5);
		b3.montrer();	b4.montrer();
		System.out.println("***BABELISER_EN_CROIX***");
		Babel b6 = new Babel(2, 4);
		Babel b7 = new Babel(0, 1);
		b6.babeliserEnCroix(b7, 3);
		b6.montrer();	b7.montrer();
		System.out.println("***BABELISER_DIRECTEMENT***");
		Babel b8 = new Babel(2, 4);
		Babel b9 = new Babel(0, 1);
		b8.babeliserDirectement(b9, 3);
		b8.montrer();	b9.montrer();
	}}//fin de la méthode main et de la classe principale
```

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

- *l'opérateur **%** est l'opérateur modulo ;*
- *l'indice du premier élément d'un tableau a toujours la valeur **0**.*

........................................................................................................................................

........................................................................................................................................

........................................................................................................................................

........................................................................................................................................

........................................................................................................................................

........................................................................................................................................

........................................................................................................................................

........................................................................................................................................

........................................................................................................................................

........................................................................................................................................

........................................................................................................................................

........................................................................................................................................

........................................................................................................................................

........................................................................................................................................

........................................................................................................................................

........................................................................................................................................

........................................................................................................................................

........................................................................................................................................

........................................................................................................................................

........................................................................................................................................

........................................................................................................................................

........................................................................................................................................

........................................................................................................................................

........................................................................................................................................

........................................................................................................................................

........................................................................................................................................

........................................................................................................................................

........................................................................................................................................

........................................................................................................................................

..........................................................................................................................................

..........................................................................................................................................

..........................................................................................................................................

..........................................................................................................................................

..........................................................................................................................................

..........................................................................................................................................

..........................................................................................................................................

..........................................................................................................................................

..........................................................................................................................................

..........................................................................................................................................

..........................................................................................................................................

..........................................................................................................................................

..........................................................................................................................................

..........................................................................................................................................

..........................................................................................................................................

..........................................................................................................................................

..........................................................................................................................................

..........................................................................................................................................

..........................................................................................................................................

..........................................................................................................................................

..........................................................................................................................................

..........................................................................................................................................

..........................................................................................................................................

..........................................................................................................................................

..........................................................................................................................................

..........................................................................................................................................

..........................................................................................................................................

..........................................................................................................................................

..........................................................................................................................................

..........................................................................................................................................

..........................................................................................................................................

..........................................................................................................................................

..........................................................................................................................................

..........................................................................................................................................