+1/1/60+

DS Java, mercredi 23 janvier 2013 **CCI-LO**

Durée: deux heures. Aucun document autorisé. Le barème est indicatif.

Partie I (répondre directement sur le sujet)

Les réponses aux questions à choix multiples doivent être données sur la dernière page. Sauf mention explicite du contraire, ces questions ont une unique bonne réponse. Des points négatifs sont affectés en cas de mauvaise réponse.

Exercice 1 (5 points)**.**

On considère les les trois fichiers Pile.java, PileTest.java et Application.java donnés en annexe. Pour chacun des bouts de code suivants, indiquez quels sont les fichiers qui compilent après insertion du code juste après le commentaire // POINT D'INSERTION.

Question 1 ♣ *System.out.println(Pile.hauteur);*

[A] *Pile.java* [B] *PileTest.java* [C] *Application.java*
[D] Aucune de ces réponses n'est correcte.

Question 2 ♣ *Pile pile = **new** Pile(); System.out.println(pile.hauteur);*

[A] *Pile.java* [B] *PileTest.java* [C] *Application.java*
[D] Aucune de ces réponses n'est correcte.

Question 3 ♣ *Pile pile = **new** Pile(); pile.hauteur = 3;*

[A] *Pile.java* [B] *PileTest.java* [C] *Application.java*
[D] Aucune de ces réponses n'est correcte.

Question 4 ♣ *Pile pile = **new** Pile(); System.out.println(pile.HAUTEURMAX);*

[A] *Pile.java* [B] *PileTest.java* [C] *Application.java*
[D] Aucune de ces réponses n'est correcte.

Question 5 ♣ *Pile.HAUTEURMAX = 4;*

[A] *Pile.java* [B] *PileTest.java* [C] *Application.java*
[D] Aucune de ces réponses n'est correcte.

Question 6 ♣ *System.out.println(Pile.HAUTEURMAX);*

[A] *Pile.java* [B] *PileTest.java* [C] *Application.java*
[D] Aucune de ces réponses n'est correcte.

Question 7 ♣ *Pile pile = **new** Pile(); pile.HAUTEURMAX = 3;*

[A] *Pile.java* [B] *PileTest.java* [C] *Application.java*
[D] Aucune de ces réponses n'est correcte.

Question 8 ♣ *Pile pile = **new** Pile(); System.out.println(pile.getHauteur());*

[A] *Pile.java* [B] *PileTest.java* [C] *Application.java*
[D] Aucune de ces réponses n'est correcte.

Question 9 ♣ *System.out.println(Pile.getHauteur());*

[A] *Pile.java* [B] *PileTest.java* [C] *Application.java*
[D] Aucune de ces réponses n'est correcte.

Question 10 ♣ *getHauteur();*

[A] *Pile.java* [B] *PileTest.java* [C] *Application.java*
[D] Aucune de ces réponses n'est correcte.

Exercice 2 (1 point). *Quel principe mets-on en oeuvre en mettant l'attribut hauteur en «private»? Quel est son intérêt?*

Exercice 3 (2.5 points).

- *Quelle est la limitation principale de la classe Pile?*

- *Quel est le rôle de la valeur booléenne renvoyée par empiler.*

- *On rappelle les principales méthodes de la classe Vector:*

```
class Vector<T> {
    public boolean add(T t);
    public T remove(int index);
    public T get(int index);
    public boolean isEmpty();
    ...
}
```

. *Proposez une implantation alternative de Pile corrigeant cette limitation à l'aide d'un Vector. Pour cela vous annoterez directement le code de la classe Pile avec vos changements.*

- *Précisez en quoi la spécification de la classe Pile a changé.*

Exercice 4 (1,5 point). *On considère la classe de test PileTest.*
Question 11 *Le test testDepiler passe-t'il avec votre implantation de la classe Pile?*

[A] *Oui* [B] *Non, le code est incorrect* [C] *Non, le test est incorrect*

Question 12 *Le test testDepiler passe-t'il avec l'implantation originale?*

[A] *Oui* [B] *Non, le code est incorrect* [C] *Non, le test est incorrect*

Question 13 *Le test testEmpiler passe-t'il avec l'implantation originale?*

[A] *Oui* [B] *Non, le code est incorrect* [C] *Non, le test est incorrect*

Annexe 1: contenu de Pile.java:

```
public class Pile {
    private int [] valeurs;
    private int hauteur;
    final static public int HAUTEURMAX = 3;

    public Pile() {
        valeurs = new int [HAUTEURMAX];
        hauteur = 0;
    }

    public boolean empiler(int n){
        if(hauteur >= HAUTEURMAX)
            return false;
        else {
            valeurs[hauteur] = n;
            hauteur = hauteur + 1;
            return true;
        }
    }

    public int depiler(){
        if(hauteur <= 0) return -1;
        else {
            hauteur = hauteur - 1;
            return valeurs[hauteur];
        }
    }

    public int dessus(){
        if(hauteur <= 0) return -1;
        else             return valeurs[hauteur-1];
    }

    public int getHauteur(){
        // POINT D'INSERTION
        return hauteur;
    }
}
```

Annexe 2: contenu de PileTest.java:

```
import org.junit.*;
import static org.junit.Assert.*;

public class PileTest {

    @Test
    public void testEmpiler(){
        Pile unePile = new Pile();
        assertEquals(unePile.empiler(12), true);
        assertEquals(unePile.dessus(), 12);
        assertEquals(unePile.getHauteur(), 1);
        assertEquals(unePile.empiler(4), true);
        assertEquals(unePile.dessus(), 4);
        assertEquals(unePile.getHauteur(), 2);
        // POINT D'INSERTION
    }

    @Test
    public void testDepiler(){
        Pile unePile = new Pile();
        assertEquals(unePile.empiler(11), true);
        assertEquals(unePile.empiler(4), true);
        assertEquals(unePile.empiler(8), true);
        assertEquals(unePile.dessus(), 8);
        assertEquals(unePile.depiler(), 8);
        assertEquals(unePile.empiler(13), true);
        assertEquals(unePile.empiler(15), false);
        assertEquals(unePile.depiler(), 13);
    }
}
```


Annexe 3: contenu de Application.java:

```
class Application {
    public void main(String[] args) {
        Pile unePile = new Pile();
        // POINT D'INSERTION
    }
}
```

Partie II (répondre sur une feuille séparée)

Source: `http://fr.wikipedia.org/wiki/Jeu_de_la_vie`.

Le jeu de la vie, automate cellulaire imaginé par John Horton Conway en 1970, est probablement, à l'heure actuelle, le plus connu de tous les automates cellulaires. Malgré des règles très simples, le jeu de la vie permet le développement de motifs extrêmement complexes.

En préambule, il faut préciser que le jeu de la vie n'est pas vraiment un jeu au sens ludique, puisqu'il ne nécessite aucun joueur; il s'agit d'un automate cellulaire, un modèle où chaque état conduit mécaniquement à l'état suivant à partir de règles pré-établies.

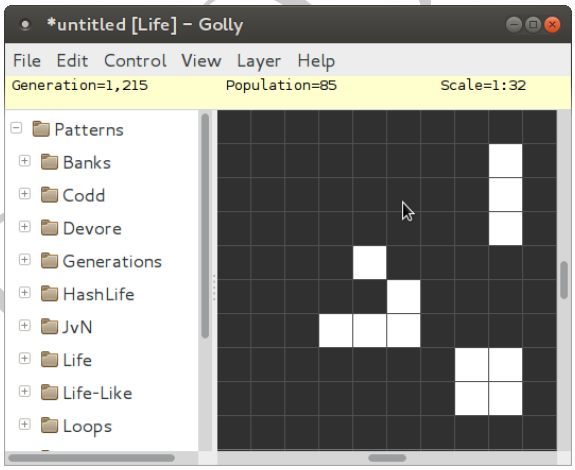
Le jeu se déroule sur une grille à deux dimensions, théoriquement infinie (mais de longueur et de largeur finies et plus ou moins grandes dans la pratique), dont les cases — qu'on appelle des «cellules», par analogie avec les cellules vivantes — peuvent prendre deux états distincts : «vivantes» ou «mortes».

À chaque étape, l'évolution d'une cellule est entièrement déterminée par l'état de ses huit voisines de la façon suivante:

- Une cellule morte possédant exactement trois voisines vivantes devient vivante (elle naît).
- Une cellule vivante possédant deux ou trois voisines vivantes le reste, sinon elle meurt.

Ainsi, la configuration  donne au tour suivant la configuration  qui redonne ensuite la première.

On souhaite implanter, en respectant l'architecture MVC, une application permettant d'éditer un grille de jeu de la vie et de visualiser son évolution. Voici un exemple de capture d'écran d'une telle application, golly:



Les cases préremplies sont marquées en gras. L'utilisatrice (ou utilisateur) interagit principalement avec l'application en sélectionnant une case d'un clic de la souris. Si celle ci était vivante, elle devient morte, et réciproquement. La barre d'espacement fait évoluer le modèle d'une seule étape. La touche Retour lance l'évolution continue du modèle La barre de menus permet de faire les actions usuelles (nouvelle grille, lecture et sauvegarde de grille, configuration, ...), et la partie de gauche de choisir diverses grilles classiques.

Exercice 5 (1 point). *Donner l'arbre des composants graphiques (widgets) correspondant à la vue de l'application.*

Exercice 6 (1 point). *Vous êtes en charge de l'*analyse *du modèle, sachant que le développement de celui-ci sera réalisé par une autre équipe. Il faudra principalement modéliser une grille du jeu de la vie. Écrire l'interface «GrilleJeuDeLaVie» avec toutes les méthodes appropriées.*

Exercice 7 (2 points). *Vous êtes maintenant en charge de l'implantation d'un tel modèle.*

- *Quel type et quels droits d'accès choisissez-vous pour son attribut grille permettant de stocker l'état de la grille?*
- *Implanter la méthode* ***private int*** *nombreVoisinsVivants(****int*** *i,* ***int*** *j).*

Exercice 8 (2 points)**.** *Compléter le découpage du projet en classes respectant le patron Modèle-Vue-Contrôleur. On précisera uniquement le nom des classes et interfaces en jeu (celles du projet, et celles de Java directement utilisées par le projet), leurs rôles, et leurs relations d'héritage, d'implantation, et de contenance. On fera un schéma résumant les interactions (appels de méthodes) entre les composants du modèle, de la vue, et du contrôleur.*

Exercice 9 (2 points)**.** *Donner les grandes lignes d'une classe VueGrilleJeuDeLaVie implantant le composant affichant une grille du jeu de la vie.*

Exercice 10 (2 points)**.** *Donner le code complet de la classe du contrôleur qui réagit au clic sur une case. On rappelle qu'il doit implanter l'interface suivante:*

```
interface ActionListener {
    public abstract void actionPerformed(ActionEvent arg0);
}
```

Préciser la ligne de code permettant de construire un tel contrôleur et l'attacher à un bouton bouttonCase de coordonnées i,j.

Feuille de réponses, Questions à Choix Multiples, CCI-LO, 23/01/2013

Écrivez votre nom et prénom ci-dessous, et codez votre numéro d'étudiant:

Nom et prénom:..

☐0 ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9
☐0 ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9
☐0 ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9
☐0 ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9
☐0 ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9
☐0 ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9
☐0 ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9
☐0 ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9

Les réponses aux questions à choix multiple sont à donner exclusivement sur cette feuille: les réponses données sur les feuilles précédentes ne seront pas prises en compte.

Question 1 : ☐A ☐B ☐C ☐D
Question 2 : ☐A ☐B ☐C ☐D
Question 3 : ☐A ☐B ☐C ☐D
Question 4 : ☐A ☐B ☐C ☐D
Question 5 : ☐A ☐B ☐C ☐D
Question 6 : ☐A ☐B ☐C ☐D
Question 7 : ☐A ☐B ☐C ☐D
Question 8 : ☐A ☐B ☐C ☐D
Question 9 : ☐A ☐B ☐C ☐D
Question 10 : ☐A ☐B ☐C ☐D
Question 11 : ☐A ☐B ☐C
Question 12 : ☐A ☐B ☐C
Question 13 : ☐A ☐B ☐C