UFR IM²AG

UNIVERSITÉ Grenoble Alpes

# INF362 - Quick1
# Année 2015-2016

**Durée : 1h.**
**Tout document interdit. Calculatrices interdites.**
**Pour chaque question, une partie des points peut tenir compte de la présentation.**
**Le barême est indicatif.**

## 1 Questions de cours (5 points)

1. (1,5 points) comment peut-on allouer un tableau en mémoire en java ? Y-a-t'il une seule manière de le faire (justifiez votre réponse) ?
2. (1,5 points) que signifie qu'une classe A hérite d'une autre classe B ?
3. (1 points) une classe hérite-t-elle toujours d'une autre classe ?
4. (1 points) à quel moment un constructeur est-il appelé ?

## 2 Piles (15 points)

Vous disposez des interfaces et classes suivantes, qui implémentent une pile :

```
package Pile;

public interface Pile {
    public void empiler(Object element);
    public Object depiler()
        throws ExceptionPileVide;
    public boolean estVide();
    public IterateurPile iterateur();
}
```

```
package Pile;

public class ExceptionPileVide
    extends Exception {
}
```

```
package Pile;

class Maillon {
    Object element;
    Maillon suivant;
/* BEGIN SOLUTION */
    Maillon precedent;
/* END SOLUTION */
}
```

```
package Pile;

public class FabriquePile {
    public static Pile creer() {
        return new PileListe();
    }
}
```

```
package Pile;

class PileListe implements Pile {
    private Maillon sommet;
/* BEGIN SOLUTION */

    private Maillon fond;
/* END SOLUTION */

    public PileListe() {
        sommet = null;
/* BEGIN SOLUTION */

        fond = null;
/* END SOLUTION */
    }

    public void empiler(Object element) {
        Maillon nouveau;

        nouveau = new Maillon();
        nouveau.element = element;
        nouveau.suivant = sommet;
        sommet = nouveau;
/* BEGIN SOLUTION */

        if (nouveau.suivant == null)
            fond = nouveau;
        else
            nouveau.suivant.precedent =
                nouveau;
        nouveau.precedent = null;
/* END SOLUTION */
    }

    public Object depiler()
            throws ExceptionPileVide {
        Object resultat;

        if (sommet == null)
            throw new ExceptionPileVide();
        resultat = sommet.element;
        sommet = sommet.suivant;
/* BEGIN SOLUTION */

        if (sommet == null)
            fond = null;

/* END SOLUTION */
        return resultat;
    }

    public boolean estVide() {
        return sommet == null;
    }
/* BEGIN SOLUTION */

    public IterateurPile iterateur() {
        return new IterateurPileListe(fond);
    }
/* END SOLUTION */
}
```

**Questions :**

1. (7 points) Comme vous pouvez le constater, la classe `PileListe` n'implémente pas l'intégralité de l'interface `Pile`. Complétez la classe `PileListe` en écrivant une interface `IterateurPile`, une classe `IterateurPileListe` qui implémente cette interface et la méthode manquante dans `PileListe`. L'itérateur qu'il vous est demandé d'écrire devra permettre de parcourir le contenu de la pile depuis l'élément du fond (le plus ancien) vers le sommet (élément le plus récent) et toutes ses opérations devront se faire en temps constant. Vous prendrez soin de détailler dans votre copie les éventuels compléments ou modifications que vous apporterez au code fourni.
2. (6 points) Écrivez un programme principal permettant de créer un pile grâce à la fabrique, de la remplir avec 10 entiers générés aléatoirement en affichant, à chaque entier empilé, le contenu complet de la pile puis, une fois la pile pleine, de dépiler chaque entier empilé en affichant sa valeur. Votre programme devra respecter l'interface `Pile` et la classe `FabriquePile` fournies précédemment (sans modification).
3. (1 point) La fabrique fournie ici n'est pas abstraite, quelle implication cela a-t-il ?
4. (1 point) Que se passe-t-il si un élément est dépilé (avec la méthode `depiler`) dans le corps d'une boucle de parcours des éléments de la pile à l'aide de l'itérateur ?

## Choses Utiles

```
Classe Random (dans java.util.*) :
    int nextInt(int n);
```