

IFT1025 - Programmation 2 - Examen Intra

Professeurs : Nicolas Hurtubise, Alena Tsikhanovich

27 février 2019

Prénom, Nom : _____, _____

Matricule : _____ *Programme d'études* : _____

0 Directives

- Vous pouvez détacher la dernière page de l'examen, il s'agit de la feuille d'Annexe
- À la fin de l'examen, assurez-vous de remettre **toutes** les feuilles (questionnaire + feuille de réponses)
- L'examen est d'une durée de *1h50*
- Aucune documentation permise, à l'exception de deux feuilles de notes au format $8\frac{1}{2} \times 11$ recto-verso, **écrites de votre main**. Aucun appareil électronique n'est permis.
- Sur la feuille de réponses, dans la boîte en haut à gauche, indiquez le sigle du cours (IFT1025), la faculté (FAS), la date, et votre signature.
- Sur la feuille de réponses, remplissez la section a. (NOM et PRÉNOM), la section b. (MATRICULE), et la section c. (VERSION 0). Noircir les ovales appropriés. Laissez la section d. vide.
- Assurez-vous d'indiquer votre *Matricule* sur **chaque page du questionnaire**. Profitez-en pour vous assurer que vous avez bien un total de **10 pages**
- **Tout plagiat entraînera la note de 0% automatiquement**
- Répondez aux questions de la *Section 1* sur la **feuille de réponses** en noircissant un seul des ovales par question
- Répondez aux autres questions directement sur le **questionnaire**

Bon examen !

1 Questions diverses (70%)

Barème pour la section 1 : *pas de correction négative*, toutes les questions ont la même pondération (2% de la note totale)

1. Qu'affiche `System.out.println((int) 10.6);` ?
a. 10 b. 10.0 c. 11 d. 11.0 e. erreur
2. Qu'affiche `System.out.println((int) "10.5");` ?
a. 10 b. "10.5" c. "10" d. null e. erreur
3. Quel est le type de l'expression `(10 + 5.5f)` ?
a. int b. long c. float d. double e. erreur
4. Qu'affiche `System.out.println(7 / 10);` ?
a. 0 b. 0.7f c. 0.7 d. 1 e. erreur
5. Une méthode sans implémentation s'appelle :
a. une interface
b. une méthode surchargée (overloaded)
c. une méthode statique
d. une méthode polymorphique
e. une méthode abstraite (abstract)
6. _____ est une collection de classes reliées par un sens commun :
a. une collection
b. un import
c. un package
d. une interface
e. une instance
7. Indiquez lequel de ces énoncés est vrai :
a. long est une sous-classe de Object
b. int est une sous-classe de long
c. `Integer i = (int) 10;` est une ligne valide en Java
d. Number est une classe déclarée final
e. `int "implements" Integer`

Les six prochaines questions se rapportent à la classe qui suit.

Pour chaque ligne suivie d'un commentaire `// Question #...`, indiquez ce qui est affiché par l'appel de fonction.

Notez qu'il n'y a pas d'erreur dans le code. Les choix de réponse possibles sont illustrés à la page suivante. Si la bonne réponse ne se trouve pas dans les choix, choisissez E. aucune

```
public class Main {
    public static void imprimer(String msg, int[] [] tab) {
        System.out.println(msg);
        for (int i=0; i<tab.length; ++i)
            for(int j = 0; j<tab[i].length; j++)
                System.out.print(tab[i][j]+"\\t");
        System.out.println("\\n--");
    }

    public static int f1(int x) {
        x = 1;
        return x;
    }

    public static void f2(int[] [] tab1, int[] [] tab2) {
        tab1[1] = tab2[0];
    }

    public static void f3(int[] [] tab1, int[] [] tab2) {
        tab1 = tab2;
        tab1[0][0] = -1;
    }

    public static void main(String[] args) {
        int[] [] t1 = {{1,2},{3}};
        int[] [] t2 = {{4,5},{6,7},{8}};
        imprimer("t1",t1); // Question #8
        System.out.println(f1(t1[1][0])); // Question #9
        imprimer("après f1, t1",t1); // Question #10
        f2(t1,t2);
        imprimer("après f2, t1",t1); // Question #11
        f3(t1,t2);
        imprimer("après f3, t1",t1); // Question #12
        imprimer("après f3, t2",t2); // Question #13
    }
}
```

	Question 8	Question 9	Question 10
A	t1 1 2 3 ---	1 2 3	après f1, t1 123 ---
B	1 2 3 ---	1	après f1, t1 1 2 1 ---
C	t1 12 3 ---	2	après f1, t1 1 1 3 ---
D	t1 1 2 3 ---	3	après f1, t1 12 3 ---
E	aucune	aucune	aucune

Figure 1: Choix de réponse pour les questions 8 à 10

	Question 11	Question 12	Question 13
A	après f2, t1 1 2 3 ---	après f3, t1 -1 2 4 5 ---	après f3, t2 -1 2 4 5 ---
B	après f2, t1 4 5 3 ---	après f3, t1 1 2 -1 5 ---	après f3, t2 4 5 6 7 8 ---
C	après f2, t1 1 2 4 5 6 7 8 ---	après f3, t1 -1 2 4 5 ---	après f3, t2 -1 5 6 7 8 ---
D	après f2, t1 1 2 4 5 ---	après f3, t1 -1 2 1 5 ---	après f3, t2 -1 6 7 8 ---
E	aucune	aucune	aucune

Figure 2: Choix de réponse pour les questions 11 à 13

Pour les quatre prochaines questions, tenez compte des dessins A, B, C, D suivants :

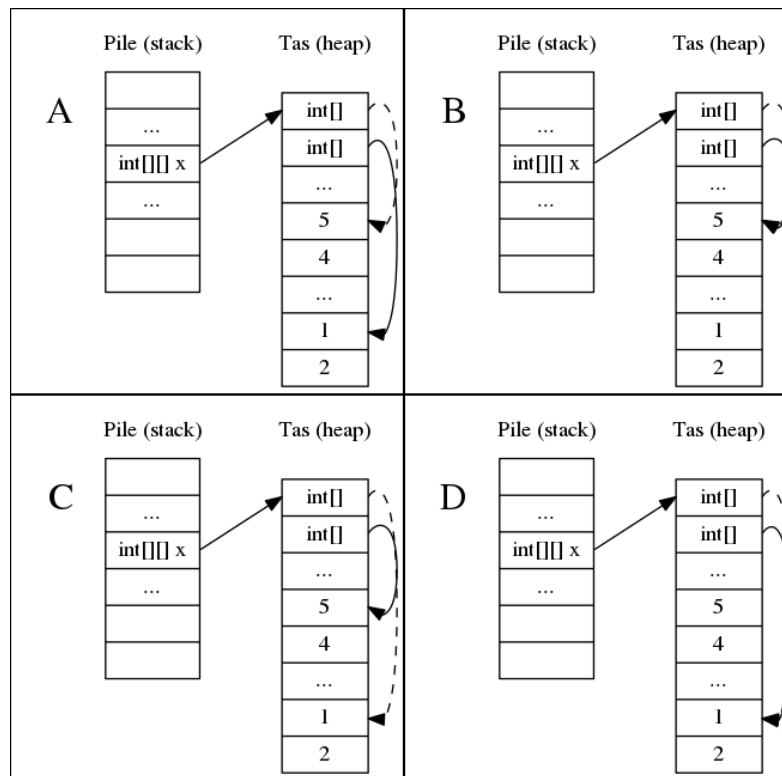


Figure 3: Choix de réponse possibles pour les question 14 à 17

Ces dessins représentent le contenu de la mémoire et la relation entre les données structurées (tableaux) créés par un programme et qui sont accessibles depuis la variable `x` du programme.

Pour chaque question, vous devez indiquer le modèle mémoire qui correspond exactement aux données structurées créées par le code donné dans la question.

Si aucun des dessins A, B, C, D ne s'applique au code, répondez E.

14. Modèle mémoire pour le code :

```
int[] [] x = { {5,4},{1,2} };
```

15. Modèle mémoire pour le code :

```
int[] [] x = {{5,4}, new int[2]};
int[] y = {1,2};
x[0] = y;
x[1] = x[0];
```

16. Modèle mémoire pour le code :

```
int[] [] x = new int[2][2];
x[0] = new int[2];
x[0][0] = 5;
x[0][1] = 4;
x[1] = new int[2];
x[1][0] = 1;
x[1][1] = 2;
```

17. Modèle mémoire pour le code :

```
int[] [] x = {{5,4}, {1,2}};
int[] [] y = {x[1], x[0]};
x = y;
```

Soit la hiérarchie de classes décrite dans l'*Annexe*

Soient les variables :

```
QuestionDeveloppement qd = new QuestionDeveloppement();
VraiOuFaux vf = new VraiOuFaux();
ChoixMultiple cm = new ChoixMultiple();
```

Pour les énoncés 18 à 24, indiquez si l'expression évalue à true, false ou cause une erreur. Utilisez les choix de réponses suivants :

- a. true
- b. false
- c. erreur

- | | |
|---|--|
| 18. new ChoixMultiple() instanceof VraiOuFaux | 19. vf instanceof Question |
| 20. qd instanceof QuestionDeveloppement | 21. vf instanceof Object |
| 22. cm instanceof Affichable | 23. new Affichable() instanceof Affichable |
| 24. new Question() instanceof Affichable | |

Pour les énoncés 25 à 30, indiquez s'il s'agit d'un upcast ou d'un downcast, d'un cast facultatif ou nécessaire, ou si le cast cause une erreur. Utilisez les choix de réponses suivants :

- a. upcast, nécessaire
- b. upcast facultatif
- c. downcast nécessaire
- d. downcast facultatif
- e. erreur

- | | |
|-------------------------------------|---------------------------------------|
| 25. Question q = (Question) vf; | 26. VraiOuFaux vf2 = (Question) qd; |
| 27. Affichable a = (Affichable) qd; | 28. QuestionDeveloppement qd2 = cm; |
| 29. ChoixMultiple cm2 = vf; | 30. VraiOuFaux vf3 = (VraiOuFaux) cm; |

31. Il serait possible de définir :

- a. Une interface qui extends `VraiOuFaux`
- b. Une classe qui implements `QuestionDeveloppement`
- c. Une interface qui implements `VraiOuFaux`
- d. Une classe qui extends `Affichable`
- e. Une classe qui extends `ChoixMultiple`

32. Il serait possible d'ajouter :

- a. Une méthode `'public String afficher()'` dans la classe `VraiOuFaux`
- b. Une méthode `'public int toString()'` dans la classe `Question`
- c. Une méthode `'public String afficher()'` dans l'interface `Affichable`
- d. Une méthode `'public ArrayList<String> afficher(String x)'` dans la classe `VraiOuFaux`
- e. Une méthode `'public static double getNum()'` dans la classe `Question`

33. Le nombre d'opérations nécessaires pour trouver si une valeur est contenue dans une liste simplement chaînée *triée* contenant n éléments se fait en pire cas dans un ordre de grandeur de :

- a. $O(1)$
- b. $O(\log(n))$
- c. $O(n)$
- d. $O(n \cdot \log(n))$
- e. $O(n^2)$

34. Le nombre d'opérations nécessaires pour lire ou écrire un élément dans un tableau (array) dans une position i arbitraire se fait en pire cas dans un ordre de grandeur de :

- a. $O(1)$
- b. $O(\log(n))$
- c. $O(n)$
- d. $O(n \cdot \log(n))$
- e. $O(n^2)$

35. Parmi les structures de données suivantes, choisissez le type abstrait :

- a. File avec priorité
- b. Monceau
- c. Arbre binaire de recherche
- d. `ArrayList`
- e. Liste doublement chaînée

2 Écrire une classe (30%)

On s'intéresse à avoir une classe nommée `Quadratique` qui permet de manipuler un polynôme quadratique, soit un polynôme de degré 2 de la forme :

$$ax^2 + bx + c$$

- Cette classe doit avoir :
 1. Un constructeur qui permet de spécifier les valeurs a , b et c (dans cet ordre)
 2. Une façon d'afficher le polynôme sur la console en utilisant :

```
Quadratique p = new Quadratique(10.5, 20.5, 30.5); // a=10.5, b=20.5, c=30.5
System.out.println(p);
// => Doit afficher sur la console : 10.5x^2 + 20.5x + 30.5
```
 3. Une méthode statique pour additionner deux polynômes quadratiques :

$$p_1 + p_2 = (a_1 + a_2)x^2 + (b_1 + b_2)x + (c_1 + c_2)$$

4. Une méthode d'instance `public String multiplication(Quadratique autre)` pour calculer le produit du polynôme quadratique par un autre. Notez que le produit de deux polynômes quadratiques n'est pas forcément un polynôme quadratique, d'où le besoin de retourner le résultat dans une `String` directement.

$$p_1 * p_2 = a_2x^2(a_1x^2 + b_1x + c) + b_2x(a_1x^2 + b_1x + c) + c_2(a_1x^2 + b_1x + c)$$

Réduisez le polynôme résultant au maximum, vous devriez avoir quelque chose de la forme de :

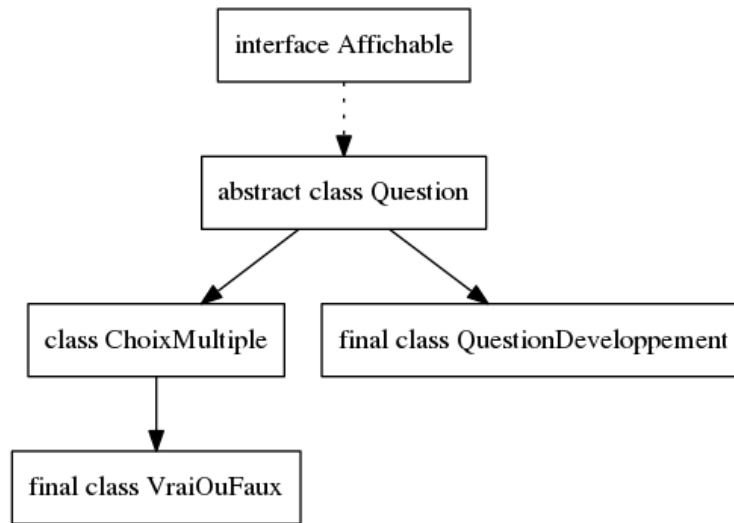
$$p_1 * p_2 = c_1x^4 + c_2x^3 + c_3x^2 + c_4x + c_5$$

- Respectez le principe d'encapsulation
- Écrivez un *getter* (accesseur) et un *setter* (mutateur) pour un des attributs pour montrer que vous comprenez le concept (pas besoin d'en écrire pour tous les attributs)
- Mettez des commentaires appropriés (vous pouvez utiliser le standard *JavaDoc* si vous le voulez, mais ce n'est pas obligatoire, l'important est de commenter correctement chacune des fonctions)
- Si vous avez besoin de plus d'espace, utilisez le verso de la page et indiquez-le clairement

Matricule :

class Quadratique

3 Annexe



```

public interface Affichable {
    public void afficher();
}

```

```

public abstract class Question implements Affichable {
    protected String enonce;
    public int nbPoints;
    private static int num = 0;

    public Question() {
        Question.num++;
    }

    public abstract boolean reponseValide(String reponse);

    public static int getNum() {
        return num;
    }

    public void afficher() {
        System.out.println(enonce + " (" + nbPoints + " points)");
    }
}

```

```

public class ChoixMultiple extends Question { ... }

```

```

public final class VraiOuFaux extends ChoixMultiple { ... }

```

```

public final class QuestionDeveloppement extends Question { ... }

```