BPOO - Contrôle du 18 Mars 2014

DUT/INFO/M2103

Table des matières

Barème indicatif

```
TODOTODOTODOTODO
Question 1 (2 Points)
Question 2 (2 Points)
Question 3 (3 Points)
Question 4 (3 Points)  (*)
Question 5 (2 Points)
Question 6 (2 Points)  (*)
Question 7 (3 Points)  (*)
Question 8 (3 Points)  (*)
```

Les questions marquées (*) sont indépendantes les unes des autres.

```
                    ===== Durée 1h =====

                Feuille manuscrite non photocopiée recto autorisée

                    ==================
```

Question sur encapsulation ?

C’est quoi ? Le fait de ne pas accéder aux attributs d’un objet Le fait de déclarer les attributs private

(et des méthodes public) Le fait de masquer les données internes d’un objet aux clients de l’objet Le fait que les objets ont une référence

A quoi sert la pseudo-variable this

Une classe dans laquelle aucun constructeur n’est explicitement déclarée On peut déclarer des variables du type de cette classe mais pas créer d’objet de cette classes Possède un constructeur ayant pour paramètre this Possède un constructeur par défaut Possède un constructeur paramétré avec un paramètre pour tous les attributs

Quelles sont les 3 catégories de classes classiques ?

Lorsque le concepteur documente une précondition sur une méthode, cela : Renseigne sur l'état de l’objet après appel de la méthode Permet de connaître la valeur de retour de la méthode Permet de connaître les situations où elle risque de ne pas fonctionner correctement Permet au compilateur de lever automatiquement une exception si nécessaire Permet de connaître les situations où elle fonctionnera correctement Permet à la JVM (au runtime) de lever automatiquement une exception si nécessaire

Exercices

Un exo de représentation mémoire Tableaux d’int avec plusieurs réfs Tableau de Personnes Personnes

Trace avec Passage de paramètres (réfs) Faire un tableau → add `1 Un objet → set Un int → +` Un String → toUpperCase

Un exo ArrayList

Un exo de réimplémentation de classe (Heure ?)

BLABLABLABLABLABLABLABLA

En fin de soldes, un magasin souhaite réaliser l’inventaire de son stock.

Les articles qu’il vend sont répertoriés avec :

1. un nom d’article
2. un numéro d’article
3. une catégorie d’article
4. une quantité d’articles

Question 1 (Barème : 2 Points)

Ecrire en **Java** la définition des champs d'un enregistrement Article.

Question 2 (Barème : 2 Points)

Les employés souhaitent pouvoir créer un article :

- en ne fournissant que son nom et son numéro (catégorie et quantité sont alors égales à 1)
- en fournissant la valeur de tous ses champs

1. Ecrire en **Java** la définition des 2 constructeurs d'Article demandés.

Question 3 (Barème : 3 Points)

Les employés souhaitent pouvoir disposer des opérations suivantes permettant de vendre ou de reprendre un seul Article :

`enStock() : Article → booleen`

retourne VRAI si sa quantité est supérieure à 0

`vendre() : Article → entier`

retourne le nombre d'articles restants

`reprendre() : Article → entier`

retourne le nombre d'articles restants

1. Fournir les préconditions de ces actions
2. Ecrire en **java** les 3 opérations `enStock()`, `vendre()` et `reprendre()` du type Abstrait Article.

Question 4 (*) (Barème : 3 Points)

On souhaite écrire en **java** une classe Magasin

et un gentil étudiant (dont je tairais le nom) a fourni l'algorithme suivant :

```
DEBUT
 fini <- FAUX
 TANTQUE NON fini FAIRE
  Afficher (ecran) "Nom et Numéro de l'article ?"
  Saisir (clavier) nom, numero
  tabStock[indice] <- new Article(nom,numero)
  indice <- indice + 1
  Afficher (ecran) "Encore ? (O/N)"
  Saisir (clavier) reponse
  SI (reponse = "N") OU (reponse = "n") ALORS fini <- VRAI FINSI
 FINTANTQUE
FIN
```

Le tableau `tabStock[]` est de taille 1000

1. Fournir le glossaire de l'algorithme précédent
2. Corriger les 2 erreurs d'inattention de l'étudiant qui a fourni l'algorithme
3. Ecrire la classe **java** Magasin

```
.
.
.
.
.
.
.
```

Question 5 (Barème : 2 Points)

Après quelques jours d'utilisation

les employés souhaitent disposer des opérations supplémentaires suivantes :

`vendre() : Article X Entier → entier`

met à jour et retourne le nombre d'articles restants après retrait du nombre fourni

`reprendre() : Article X Entier → entier`

met à jour et retourne le nombre d'articles restants après ajout du nombre fourni

1. Fournir les préconditions de ces actions
2. Ecrire en **Java** les 2 opérations supplémentaires du type Abstrait Article.

Question 6 (*) (Barème : 2 Points)

Le responsable du magasin souhaite lors de l'inventaire de son stock cumuler le nombre d'articles par catégorie.

1. En supposant que le tableau tabStock[] ait été préalablement rempli, fournir le code **java** de la fonction `stockCategories(tabStock)` qui retourne un tableau d'entiers contenant le nombre total d'articles de chaque catégorie et le nombre total d'articles toutes catégories confondues à l'indice 0.

Le magasin vends exactement 4 catégories d'articles

Les numéros de catégorie sont 1,2,3 et 4

Exemple:

```
SI stockCategories() retourne [6,2,0,1,3]
ALORS
le magasin dispose
 d'un total de 6 articles,
dont 2 de catégorie 1,
 0 de catégorie 2,
 1 de catégorie 3
 et 3 de catégorie 4.
```

```
.
.
.
.
.
.
.
.
.
.
.
```

Question 7 (*) (Barème : 3 Points)

Le responsable du magasin souhaite obtenir l'affichage suivant de son stock :

```
 Catégorie 1 : souris, clavier
 Catégorie 2 :
 Catégorie 3 : cleUSB
 Catégorie 4 : tablette, tour, portable
```

En supposant que le tableau `tabStock[]` ait été préalablement rempli et que la classe Pile (d'entiers) vue en TD/TP soit disponible

A partir de l'algorithme suivant :

```
DEBUT
POUR indice de 0 à nbArticles FAIRE
 article <- tabStock[indice]
 SI article.categorie = 1 ALORS pileCategorie1.empiler(article.nom) FINSI
 SI article.categorie = 2 ALORS pileCategorie2.empiler(article.nom) FINSI
 SI article.categorie = 3 ALORS pileCategorie4.empiler(article.nom) FINSI
 SI article.categorie = 4 ALORS pileCategorie4.empiler(article.nom) FINSI
FINPOUR
/* Affichage */
FIN
```

1. Compléter la partie Affichage de l'algorithme
2. Fournir le code **java** de la procédure `nomsParCategories(tabStock)` qui affiche à l'écran l'état

demandé.

Question 8 (*) (Barème : 3 Points)

En supposant que les 4 piles `pileCategorie[1234]` ont été remplies,

1. Ecrire une fonction `nomsArticlesTries()` qui retourne une chaîne de caractères contenant la liste des noms d'articles du stock triée par ordre alphabétique croissant à partir des 4 piles fournies en paramètre.

   Exemple :

   ```
   nomsArticlesTries(pileCategorie1,pileCategorie2,pileCategorie3,pileCategorie4)
   -> "clavier + cleUSB + portable + souris + tablette + tour"
   ```

2. Fournir la signature d'une fonction `nomsArticlesTries()` qui prend un tableau de piles en paramètres
3. Fournir le code **java** de cette nouvelle fonction `nomsArticlesTries()` qui prend un tableau de piles en paramètres

---

Dernière mise à jour 2014-03-14 11:12:26 CET