CENTRE SCOLAIRE SAINTE-JULIENNE

TA 7 – POO Récursivité

Exercices Java – Série 3 – Enoncés

I- Mise en situation

Tu es analyste-programmeur dans une société et tu dois passer un test en langage Java. A travers une série d'exercices, tu dois comprendre et maitriser le langage Java pour obtenir la prime salariale.

II- Objets d'apprentissage

Appliquer	**Transférer**
• Modéliser une logique de programmation orientée objet • Déclarer une classe • Instancier une classe (objet) • Utiliser les méthodes de l'objet instancié • Traduire un algorithme dans un langage de programmation • Commenter des lignes de codes. • Tester le programme conçu	• Développer une classe sur la base d'un cahier des charges en respectant le paradigme de la programmation orientée objet (POO) • Programmer en recourant aux classes nécessaires au développement d'une application orientée objet • Corriger un programme défaillant • Améliorer un programme pour répondre à un besoin défini
Connaître	
• Différencier la programmation impérative de la programmation orientée objet • Caractériser une classe • Décrire la création d'un objet (instanciation) • Identifier l'instance d'une classe • Caractériser les attributs dans une classe (encapsulation) • Caractériser les méthodes dans une classe (encapsulation) • Décrire la création d'un constructeur • Différencier les types de visibilités	

III- Travail à accomplir

1. Analyser l'énoncé du point IV correspondant au numéro de l'exercice demandé.
2. Modéliser en diagramme de classes l'exercice.
3. Réaliser l'exercice.
4. Commenter le travail.
5. Visualiser le travail.
6. Sauvegarder le document suivant les instructions données.
7. Imprimer le(s) document(s)

IV- Enoncés

1. Ex01

   Définir une classe **Fibonacci** contenant 2 variables d'instance (**nbAAfficher** (int) et **suite** (String)), 1 *constructeur* sans paramètre alors nbAAfficher vaut 5 et un message explicatif est affiché; 1 *constucteur* à 2 paramètres, les méthodes ***accesseurs*** et la méthode ***toString***.
   Prévoir une méthode **privée calculer**(int nb) récursive qui renvoie le résultat de la suite de Fibonacci pour un nombre donné et une méthode **lister**(int nbAAfficher) qui ne renvoie rien et qui appelle la méthode calculer() un certain nombre de fois pour garnir la suite qui sera affichée.
   Définir une classe **TestFibonacci** sous forme d'application qui saisit un nombre au clavier qui détermine le nombre d'éléments de la suite à afficher (premier élément = 0). Si le nombre saisi est négatif ou supérieur à 40 alors on appelle le constructeur sans paramètre sinon celui à deux paramètres.

2. Ex02

   Définir une classe **Hanoi** contenant 4 variables d'instance (**nbDisques** (int), **de**, **tmp** et **vers** (String)), 1 *constructeur* sans paramètre alors nbDisques vaut 3 et de= « A », tmp= « B » et vers= « C »; 1 *constucteur* à 1 paramètre alors de= « A », tmp= « B » et vers= « C »; 1 *constucteur* à 3 paramètres (le poteau temporaire est défini automatiquement par rapport à celui de début et celui de fin), les méthodes ***accesseurs*** et la méthode ***toString***.
   Prévoir une méthode **privée déplacer**(int nbDisques, String de, String temp, String vers) récursive qui effectue et affiche les déplacements et une méthode **effectuerDeplacement**() qui initialise le lancement des déplacements.
   Définir une classe **TestHanoi** sous forme d'application qui saisit un nombre au clavier qui détermine le nombre de disques à transférer (pour les constructeurs avec paramètre(s)).