INFO 641 Conception et Programmation Orientée Objet

Sujet TD1 et TD2

Partie 1 : Prise en main de Java. Le but de cette partie est de vous familiariser avec le langage Java, le compilateur et l'interpréteur. Il est important pour cet exercice de suivre les indications

a. Ecrivez une classe qui permet d'afficher le message « Hello world » sur la console.

b. Compilez et lancez l'exécution.

c. Modifiez le programme de la manière suivante :

- Le message qui est affiché est fourni en paramètre lors de l'exécution ; modifiez la méthode *main* en conséquence ;
- déclarez un attribut nommé *information* contenant un deuxième message à afficher ;
- déclarez une méthode *display* qui permet d'afficher concaténation entre le message fourni en paramètre et le message stocké dans l'attribut information ;
- modifiez la méthode *main* afin que l'affichage se fasse en utilisant la méthode display.

***Indications** : Pour la définition des classes, utilisez un éditeur de texte simple (par exemple Notepad ou UltraEdit). N'oubliez pas de sauvegarder les classes avec l'extension .java. Le support de la programmation sera le Java 6.0. Pour compiler et faire exécuter, voici la démarche à suivre:*

- Ouvrez une fenêtre de commande

- Positionnez vous dans le répertoire ou vous avez crée vos classes

- Tapez jdk61 - Chaque classe se compile avec des commandes du type javac nomDuPaquetage\NomDeClasse.java Pour tester votre application il faut exécuter la classe contenant la méthode main, en utilisant la commande java nomPaquetage.NomDeClasse 1

Ces trois premières opérations sont à effectuer une seule fois

Partie 2 : petite application java. Cette partie a pour but de vous faire construire une petite applications impliquant 3 classes. Elle fait travailler les liens entre classes et la gestion de collections.

Pour une application donné orientée objet il nous représenter des étudiants. Chaque étudiant possède un nom, un prénom, un numéro d'étudiant et une spécialité/promo (par exemple FI3-IAI ou FI3-IDU) .

1. Donnez la représentation UML de la classe Etudiant

2. Donnez l'implémentation Java de la classe Etudiant.

3. Ecrivez une autre classe Java dont le but est seulement l'exécution, JeuDeTests1 et permettant :

- de déclarer et créer des étudiants
- d'afficher les informations relatives a chaque étudiant déclaré ; pour cela faites le nécessaire pour pouvoir obtenir les informations concernant les étudiants par appel de

méthode. La structuration en groupe d'étudiants est une fonctionnalité que doit être ajoutée.

4. Modifiez votre diagramme de classe et puis le code Java afin d'ajouter une classe GroupeTD, sachant qu'un étudiant ne peut appartenir qu'à un seul groupe. La classe doit proposer des méthodes permettant :

- de rajouter un étudiant au groupe
- d'enlever un étudiant du groupe
- d'afficher la liste des étudiants avec leur nom et prénom.

Les étudiants suivent différents modules. Pour chaque module il y a un code, un intitulé, et un volume horaire donné en nbHCM, nbHTD, nbHTP.

5. Modifiez votre diagramme et puis le code en ajoutant la classe Module. En plus des modificateurs et accesseurs, la classe doit proposer :

- une méthode nbHeuresEtudiant() qui retourne le nombre d'heures qu'un étudiant participant au module devra suivre
- une méthode int coutEqTD(int nbGroupesTD, int nbGroupesTP) qui retourne le cout en heures équivalent TD pour une promotion ayant les nombres de groupes TD et TP donnés en paramètre, sachant qu'une heure CM vaut 1,5 heures TD, et une heure de TD vaut 1,5 heures TP.

6. Modifiez la classe Etudiant de façon à ce qu'on puisse obtenir ne nombre d'heures qu'il aura a suivre

7. Pour les plus avancés, ajoutez des fonctionnalités supplémentaires