Inscrivez-vous au cours Moodle *Introduction à la programmation java Semestre 2*. Vous y retrouverez tous les énoncés de TD et TP, et des documents importants. Il vous permettra de communiquer avec vos enseignants, ainsi qu'à déposer les travaux qui seront notés en Contrôle Continu.

Il vous faut vous inscrire à la fois au groupe de TD, et au groupe de TP qui **correspondent à votre inscription pédagogique**. Les modules d'inscriptions seront disponible une quinzaine de jours, ils sont déclaratifs et permettrons de commencer à communiquer en attendant d'avoir les listes officielles définitives. (Nous les croiserons à ce moment là)

Exercice 1 Soit le code suivant, qui permet de représenter un fruit par son nom et son poids :

```
1  public class Fruit{
       public String nom; // le nom du fruit
3      public int poids; // le poids du fruit en grammes

5      public Fruit(String n, int p){
           //A COMPLETER
7      }

9      public static void main(String[] args){
           Fruit f = new Fruit("pamplemousse", 330);
11         Fruit g = new Fruit("pamplemousse", 330);
           Fruit h = f;
13         System.out.println("Test Termine");
       }
15 }
```

1. Nous vous invitons à utiliser Eclipse, qui est présent sur les machines du script. (Chez vous, ou si l'installation ne marche pas vous pouvez toujours revenir au couple emacs/javac et travailler alors sur une console)
   - Vous pouvez le lancer par le menu ou via un terminal par la commande `eclipse &`
     Notez l'emplacement de l'espace de travail.
   - Commencez par créer un nouveau projet java nommé `TP1` (attention refusez la création de modules lorsqu'Eclipse vous le proposera, sinon vous devrez recommencer), puis vous y ajouterez une nouvelle classe `Fruit` contenant le code précédent.
   - Complétez le constructeur inachevé pour qu'il affecte les arguments qui lui sont passés à l'objet créé.
   - Lancez une exécution (bouton vert), et appréciez le confort apporté par l'IDE Eclipse.
   - Retrouvez les codes sources en ouvrant un explorateur de fichier. (Cela vous sera utile pour faire vos dépôts de travaux)
2. Que pouvez-vous dire des variables `f`, `g` et `h` utilisées dans la méthode `main` et des objets vers lesquels elles pointent ?
3. Vous pouvez visualiser graphiquement l'état de la mémoire à l'aide d'un outil en ligne : `http://pythontutor.com/java.html#mode=edit` . Introduisez-y ce code, puis choisissez "Visualize Execution" et enfin "Last". Comparez avec ce que vous aviez prévu.
4. Déplacez la méthode `main` pour en faire une méthode d'une nouvelle classe écrite dans un fichier `Test.java` (il n'y aura donc plus de fonction `main` dans la classe `Fruit`). Quelles contraintes syntaxiques de `Java` vue en cours devez vous-respecter ? Lancez une exécution.
5. On souhaiterait afficher une description du fruit : dans la classe `Test`, écrivez une méthode d'en-tête `public static void afficher(Fruit f)` qui prend un fruit en argument, ne renvoie rien, et affiche un message sous la forme "Ce fruit est un(e) xxx et pèse yyy grammes." (Vous aurez bien sûr fait en sorte d'adapter "xxx" et "yyy" en fonction de `f`). Testez votre méthode.
6. La méthode précédente aurait mieux sa place dans la classe `Fruit`. Déplacez son code dans cette classe. Pour pouvoir l'appeler depuis la classe où vous faites vos tests, vous exécuterez `Fruit.afficher(f);`. Testez cette nouvelle configuration.

7. Écrivez dans la classe `Fruit` une méthode `static Fruit hybridation(Fruit f1, Fruit f2)` qui renvoie un nouveau `Fruit` imaginé à partir de deux autres. Le résultat aura pour poids la somme de leurs poids, et pour nom la concaténation des deux noms, le tout mis entre parenthèses.

   Testez cette méthode en hybridant quelques fruits et affichez les résultats.

8. Dans un nouveau fichier, définissez une nouvelle classe `Panier` qui contient comme seul attribut un tableau de `Fruit`.

   Puisqu'un objet tableau est un type référence il peut correspondre à une instance ou être `null`. On imagine les constructeurs suivants pour notre classe :
   - `Panier(Fruit[] f)` qui défini le panier comme étant le contenant du tableau passé en argument.
   - `Panier()` qui construit un panier vide.
   - `Panier(Fruit f, Panier p)` qui construit un panier contenant en plus des fruits de `p` le fruit supplémentaire `f`. (Il vous faut faire ici un petit travail, et décider si vous partagez des références ou faites des copies)

   Indications : Ici on considère que chaque case du tableau contient un fruit (c-à-d pas de case à `null`).

   Pour le constructeur sans argument, le tableau est de taille 0 (pas égal à `null`)

9. Ecrivez dans la classe `Panier` une méthode `public static void afficher(Panier p)` qui affiche proprement le contenu de `p`. Pensez à utiliser la méthode `afficher(Fruit f)` ; testez.
10. Ecrivez une méthode `public static Panier hybridePanier(Fruit f, Panier p)` qui renvoie un nouveau panier obtenu par hybridation du fruit `f` avec chacun de ceux de `p`. Testez encore.

Remarques :
- Dans ce TP toutes les méthodes écrites sont statiques. Vous apprendrez la semaine prochaine que ce n'est en général pas la meilleure manière de procéder ;
- de la même manière, on a utilisé par commodité des attributs qui par défaut sont publics mais ils seront en général privés par la suite.

Exercice 2 *(à commencer en TP et à terminer chez vous)*

On considère la classe suivante, qui permet de stocker quelques informations relatives à un étudiant. :

```
public class Etudiant{
    String nom, prenom; // les nom et prenom de l'etudiant
    int num;            // le numero d'etudiant
    int note;       // la note de l'etudiant (sur 20)
    /* A COMPLETER ... */
}
```

1. Ajoutez un constructeur prenant en argument un nom, un prénom, un numéro d'étudiant ainsi qu'une note et les affectant aux attributs concernés (on suppose que l'utilisateur donne toujours une note entre 0 et 20).
2. Écrivez une méthode `public static void afficher(Etudiant etu)` pour afficher le contenu de l'objet au format "Nom xxx Prénom yyy (Numéro d'étudiant zzz) : Note ttt".
3. Dans une nouvelle classe `TestEtudiant` créez un `main` pour tester votre classe `Etudiant`.
4. Ajoutez à la classe `Etudiant` une méthode `estAdmis` qui prend en argument un étudiant et renvoie le booléen disant si oui ou non il a une note supérieure ou égale à 10.
5. Écrivez enfin une méthode `mention` qui prend en argument un étudiant et renvoie sa mention : "Très bien", "Bien", "Assez bien", "Passable" ou "Ajourné" si sa note est respectivement dans l'intervalle $[16, 20]$, $[14, 16[$, $[12, 14[$, $[10, 12[$ ou $[0, 10[$. Si la note n'est dans aucun de ces intervalles, la méthode renvoie "Note invalide".