Classes, héritage simple

1) En vous inspirant de la classe Crayon vue en cours, écrire une classe CompteBancaire. Cette classe devra permettre de créer des objets CompteBancaire. Un objet de type CompteBancaire aura un numéro de compte et un solde. Ces deux variables seront déclarées privées.
Le numéro du compte sera donné initialement par une variable de classe compteur, qui sera incrémentée à chaque création de compte.
Des méthodes de la classe CompteBancaire devront permettre de faire des dépôts, ou bien des retraits (si le solde le permet), sur un compte. Une méthode devra permettre d'afficher le solde d'un compte.

2) Créer une classe Ville. Une ville sera décrite par son nom et son nombre d'habitants. Le nom de la ville devra être mémorisé en majuscules. Le nombre d'habitants peut être inconnu, s'il est connu il sera toujours supérieur à zéro. Les variables d'instance seront déclarées privées.
On écrira deux constructeurs, selon que le nombre d'habitants de la ville à créer, est connu ou non. On écrira une méthode d'affichage des caractéristiques d'une ville. et une méthode donnant la catégorie de la ville : 'G' si le nombre d'habitants est supérieur à 500000, 'M' si le nombre d'habitants est compris entre 200000 et 500000, 'P' si le nombre d'habitants est inférieur à 200000.

3) Rajouter à l'exercice précédent la classe Capitale qui hérite de la classe Ville. Cette nouvelle classe aura une variable d'instance supplémentaire par rapport à la classe Ville : la variable pays.
Certaines méthodes seront simplement celles de la classe mère. D'autres devront être spécialisées. Par exemple la méthode donnant la catégorie devra retourner 'C' dans le cas d'une capitale.

4) Taper et compléter le programme InfoContact de la page suivante. Regarder L'API pour les classes Date et DateFormat.

5) Ajouter dans la classe InfoContact une variable tableau. Cette variable sera une variable de classe, c'est-à-dire une variable visible et donc utilisable par toutes les méthodes de la classe. Cette variable désignera un tableau d'au plus MAX objets du type InfoContact. MAX sera pour l'instant fixé a 10.

- Ecrire dans la classe InfoContact une méthode
  `public static void ajouter( InfoContac ic)`, qui ajoute l'objet ic dans le tableau.
- Ecrire dans la classe InfoContact une méthode
  `public static InfoContact chercher(String unNom)`, qui cherche si un objet de nom unNom est présent dans le tableau.

La méthode main créera plusieurs objets de type InfoContact, les ajoutera dans le tableau, cherchera la présence éventuelle d'un élément particulier dans le tableau.

```
import java.util.Date;
import java.util.Locale;
import java.text.DateFormat;

class InfoContact
{
    //Un format francais pour l'affichage d'une date:
  private static DateFormat df=DateFormat.getDateTimeInstance(DateFormat.LONG,
      DateFormat.SHORT,
      Locale.FRANCE);
  private static Date derniereModif;

  private String nom;
  private String email;
  private Date creation;

  public InfoContact(String unNom, String unEmail)
    {
      derniereModif = new Date();

      // completez
    }

  public void afficheToi()
    {
      System.out.println(nom + "->" + email
 +" (date de creation: " + df.format(creation) + ")");
    }

  public String tonNom()
    {
      // completez
    }

  public String tonEmail()
    {
      // completez
    }

  public static String dateDerniereModif()
    {
      // completez
    }
}

class TestInfoContact
{
  public static void main(String[] arg)
    {
      InfoContact ic = new InfoContact("toto", "toto@titi.fr");

      ic.afficheToi();
      System.out.println(InfoContact.dateDerniereModif());
    }
}
```