1 Récursivité : fonctions mystères

On considère les deux fonctions ci-dessous :

```
public static void mystere(int n){
   if (n == 0)
      return;
   else {
      System.out.println(n);
      mystere(n-1);
   }
}

public static int kezako(int n, int p){
    if (n <= 0)
       return 0;
    if (n == 1)
       return p;
    else
       return p + kezako(n-1, p);
}
```

Questions :

1. Que fera l'appel `mystere(4);` ? Donnez l'enchaînement des différents appels de la fonction.
   De manière générale, que fera un appel de la fonction `mystere` avec un argument positif. Et si l'argument est strictement négatif, que se passe-t'il ?
   À quoi sert le test `(n == 0)` ?
2. Que calcule `kezako(4,7);` ? Là encore, donnez l'enchaînement des différents appels de la fonction.
   Plus généralement, que calcule `kezako` avec des arguments positifs ?

2 Personnes

On définit :

— une classe `Date` qui aura trois attributs privés `jour`, `mois` et `annee` de type entier et
— une classe `Personne` qui aura deux attributs privés `anniversaire` de type `Date` et `nom` de type `String`.

1. Écrivez la classe `Date` avec un constructeur publique sans argument qui choisit des valeurs aléatoires pour les trois attributs. Les valeurs choisies doivent être correctes, par exemple, la valeur du `mois` est comprise entre 1 et 12. On choisira la valeur de l'attribut `annee` entre 1900 et 2100 et on simplifiera en admettant que chaque mois possède 30 jours.
   Pour simplifier, on suppose que dans la classe `Date`, on a définit une méthode statique `int choisitInt(int min, int max)` qui tire aléatoirement un entier entre `min`

2. Écrivez la classe `Personne` avec le même type de constructeur.
3. Écrivez une méthode publique de prototype `public boolean egal(Date d)` qui teste si la date `this` représente le même jour de l'année que la date `d`.

3 Fonctions sur les mots

1. Écrivez une fonction `ajoute` qui à un tableau de taille variable (i.e., `ArrayList`) de chaînes de caractères, ajoute les éléments d'un tableau de taille fixe de chaînes de caractères (i.e., `String[]`). On suppose que les éléments du premier paramètre (`ArrayList`) sont triés dans l'ordre alphabétique, par contre, le deuxième paramètre (tableau) n'a pas de propriété particulière. On souhaite que le premier paramètre soit toujours en ordre alphabétique à la sortie de cette fonction.

   Vous pouvez utiliser la méthode `compareTo` de la classe `String` :
   — `int compareTo(String anotherString)` : compare `this` avec `anotherString` par rapport à l'ordre alphabétique : la chaîne de caractères `this` est avant `anotherString` si le résultat est négatif et après sinon.
2. (facultatif) Écrivez une fonction qui prend une chaîne de caractères contenant une suite de mots séparés par des espaces et qui affiche la liste alphabétique de tous les mots contenus dans la chaîne. Tous les mots doivent être convertis en minuscules. Utilisez `ArrayList` pour stocker les mots. Les méthodes suivantes de la classe `String` peuvent vous aider :
   — `String[] split(String sep)` : renvoie le tableau des sous-chaînes de `this` délimitées par la chaîne de caractères donnée comme paramètre. En particulier, `w.split("\\s+")` renvoie le tableau des sous-chaînes de `w` délimitées par *des* espaces.
   — `String toLowerCase()` : renvoie la conversion en minuscules du mot `this`.

4 Le problème des anniversaires (Facultatif)

Écrivez trois programmes permettant, de manière expérimentale, de répondre aux questions ci-après. Pour cela, on stockera dans une `ArrayList` des personnes choisies aléatoirement. Les noms des personnes peuvent être tous identiques.

1. Combien de personnes faut-il choisir aléatoirement avant de trouver trois personnes qui sont nées le même jour de l'année ?
2. On suppose qu'on choisit aléatoirement 365 personnes. Combien d'anniversaires différents auront-elles ? (on ignore l'année de naissance)
3. Combien de personnes faut-il choisir aléatoirement avant de trouver pour chaque jour de l'année au moins une personne qui est née ce jour-là.

Chaque programme devra simuler le choix aléatoire des personnes et la vérification de leurs anniversaires.