Université des Sciences et de la Technologie Houari Boumédiene
Faculté d'Electronique et d'Informatique
Département d'Informatique

Module : Programmation Orientée Objet / POO Année Universitaire : 2023 / 2024
Sections : L2 ING A/B

TP 04-POO-

Chaines de caractères et Enumération

Objectifs :

-manipulation des chaines de caractères et Enumération.

Exercice 1 : Analyse de code

Quel résultat fournit ce programme ?

```
public class  Chaine
{
       public static void   main   (String args[])
    {
       String ch1 =new String();
    System.out.println("A - ch1 =:"+ch1+":") ;
    String ch2 ="hello"   ;
    System.out.println("B - ch2 =:"+ch2+":")  ;
    String ch3 = new String ("bonjour")       ;
    System.out.println("C - ch3 =:"+ch3+":") ;
    String ch4 = new String (ch3)      ;
    System.out.println("D- ch4 =:"+ch4+":") ;
    ch3="bonsoir"  ;
    System.out.println("E-ch4 =:"+ch4+ ":ch3 =:"+ch3+":")  ;
    ch4=ch3   ;
    ch3="au revoir";
    System.out.println("F-ch4=:"+ch4 + ":ch3=:"+ch3+":")   ;
    }
}
```

Exercice 2

Ecrire un programme qui lit au clavier un verbe du premier groupe (il s'assurera qu'il est bien terminé par er) et qui affiche la conjugaison au présent de l'indicatif. On supposera qu'il s'agit d'un verbe régulier. Autrement dit, on admettra que l'utilisateur ne fournit pas un verbe tel que manger (dans ce cas, le programme affichera nous mangons !). Les résultats se présenteront ainsi :

```
donnez un verbe regulier du premier groupe:dire
***     il ne se termine pas par er     -donnez un autre verbe:chanter
je      chante
tu      chantes
il/elle chante
nous    chantons
vous    chantez
ils/elles       chantent
```

Exercice 3

Réaliser une classe ***Repertoire*** permettant de gérer un répertoire téléphonique associant un numéro de téléphone (chaîne de caractères) à un nom. Pour faciliter les choses, on prévoira une classe ***Abonne*** destinée à représenter un abonné et disposant des fonctionnalités indispensables.

La classe ***Repertoire*** devra disposer des fonctionnalités suivantes :

- constructeur recevant un argument de type entier précisant le nombre maximum d'abonnés que pourra contenir le répertoire
- méthode ***addAbonne*** permettant d'ajouter un nouvel abonné ; elle renverra la valeur ***false*** si le répertoire est plein, la valeur ***true*** sinon,
- méthode ***getNumero*** fournissant le numéro associé à un nom d'abonné fourni en argument,
- méthode ***getNAbonnes*** qui fournit le nombre d'abonnés figurant dans le répertoire,
- méthode ***getAbonne*** fournissant l'abonné dont le rang est fourni en argument,
- méthode ***getAbonnesTries*** fournissant un tableau des références des différents abonnés, rangés par ordre alphabétique (pour simplifier, on supposera que les noms sont écrits en minuscules, sans caractères accentués).
- Redéfinir la méthode ***toString*** pour afficher les caractéristiques d'un abonné.

Écrire un programme (classe ***AppliRepertoire***) de test.

Exercice 4

On se propose d'établir les résultats d'examen d'un ensemble d'élève. Chaque élève sera représenté par un objet de type Eleve, comportant obligatoirement les champs suivants :

- Le nom de l'élève (type String)
- Son admissibilité à l'examen, sous forme d'une valeur d'un type énuméré comportant les valeurs suivantes : N(non admis), P(passable), AB(Assez bien), B(Bien), TB(Très bien).

Idéalement, les noms des élèves pourraient être contenus dans un fichier. Ici, par souci de simplicité, nous les supposerons fournis par un tableau de chaînes placé dans le programme principal. On demande de définir convenablement la classe Eleve et d'écrire un programme principal qui :

- Pour chaque élève, lit au clavier 3 notes d'examen, en calcule la moyenne et renseigne convenablement le champ d'admissibilité, suivant les règles usuelles.
  - Moyenne <10 : non admis
  - 10<=moyenne < 12 : Passable
  - 12<=moyenne< 14 : Assez bien
  - 14 <=moyenne< 16 : Bien
  - 16<= moyenne : Très bien
- Affiche l'ensemble des résultats en fournissant en clair la mention obtenue.