UMP EST Oujda **2015-16**
2ème année DAI

TP N° 5 de Java : Programmation générique

Exercices 1:
Dans la documentation java, cherchez la classe ArrayList.
Quelle est le rôle de cette classe.
Quelles sont les principales méthodes de cette classe.
Comparer cette classe avec les tableaux.
Donner un exemple d'utilisation de cette classe avec les types « String »

Exercice 2
```
public class MethGen2
{ static  <T> T hasard (T e1, T e2)
{ double x = Math.random() ;
if (x <0.5) return e1 ;
else return e2 ;
}
```
Ecrire une classe pour tester le code suivant :
```
Integer n1 = 2 ; // conversion automatique de int en Integer
Integer n2 = 5 ; // conversion automatique de int en Integer
Double x1 = 2.5 ; // conversion automatique de double en Double
hasard (n1, n2) ; // deux arguments du même type int
hasard (n1, x1) ; // les arguments ne sont pas  de même types
```
En fait, compte tenu de l'effacement, la méthode *hasard* est compilée comme si on l'avait écrite ainsi :
```
static Object hasard (Object e1, Object e2)
{ double x = Math.random() ;
if (x <0.5) return e1 ;
else return e2 ;
}
```
Si l'on souhaite davantage de vérifications à la compilation, il est possible d'imposer le type voulu pour *T* lors de l'appel de la méthode, en utilisant une syntaxe de la forme suivante :
nomClasse<type>.nomMéthode
Le nom *nomClasse* ést celui de la classe dans laquelle est définie la méthode.
Tester les codes et commenter:
```
MethGen2.<Double> hasard (n1, x1) ;//?
MethGen2.<Number> hasard (n1, x1) ;//?
```

Exercice 3 :
Dans cet exercice on veut créer la classe générique ListeGenerique<T>. Cette classe contient :
Une variable d'instance privée « Liste » de type ArrayList<T>.
Deux variables d'instances privées « position » et « nbElements » de type int.
Un constructeur avec un paramètre permettant de dimensionner la liste.
Une méthode public permettant d'ajouter un éléments
```
public void ajouter(){ }
```
Une méthode pour insérer un élément à un index donnée : insert(T element , int index){} ; l'index ne doit pas être supérieur au nombres d'éléments et ne doit pas être inferieure à zéro.
Une méthode remplace(T element, int index){} permet de remplacer un élément d'index donné ; l'index ne doit pas être supérieur au nombres d'éléments et ne doit pas être inférieure à zéro.
Une méthode pour supprimer (int index){} un élément à un index donnée ; l'index ne doit pas

être supérieur au nombres d'éléments et ne doit pas être inférieure à zéro.
Un méthode getTailleListe() pour récupérer la taille de la liste.
Une methode premier() pour retourner le premier élément. Cette méthode doit gérer le cas où la liste est vide.
Une methode dernier() pour retourner le dernier élément. Cette méthode doit gérer le cas où la liste est vide.
Une méthode suivant() pour retourner l'élément suivant de l'élément en cours. Cette méthode doit gérer le cas où la liste est vide et le cas où l'élément est le dernier de la liste.
Une méthode précèdent() pour retourner l'élément précèdent de l'élément en cours. Cette méthode doit gérer le cas où la liste est vide et le cas où l'élément est le premier de la liste.
Ecrire un programme TestList pour tester la classe générique avec des types string.
list.ajout(« premier ») ;
list.ajout(« deuxième ») ;
list.ajout(« Troisième ») ;
list.ajout(« Quatrième ») ;
Programmer un menu pour afficher le premier élément, l'élément suivant, l'élément précèdent et le dernier élément.
p (premier) **<** (précèdent) **>** (suivant) **d**(dernier) **f**(fin)
Ecrire un autre programme pour tester la classe générique des types Personne. La classe personne est comme suis :
public classe Personne {
private string nom, prenom ;
private GregorianCalendar date_nais ;
……. }
}
Ajouter à la classe générique la méthodes générique
public static <T extends Classable > void tri(ListeGenerique<T> List) trows Exception

Exercice

Dans une classe de teste, testez les codes suivants et commentez:
ListeGenerique<Number> listeNombre ;
ListeGenerique<Double> listeReels ;
listeReels = new ListeGenerique<Double>(10) ;
listeNombre= listeReels ;