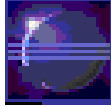
	<b>TP2</b> <b>Collections génériques en JAVA</b>	
---	---	---

Objectifs	Temps alloué	Outils
- Apprendre à manipuler les collections génériques	3h00	Eclipse

## Exercice 1 : Pile et collection générique

Voici une interface définissant un type abstrait "Pile de <A>" avec les fonctionnalités classiques d'une pile :

```
public interface IPile <A> {
    boolean estVide();
    void empile(A a);
    A depile(); // retourne l'élément en sommet de pile et dépile
    int nbElements();
    A sommet(); // retourne le sommet de pile mais ne le dépile pas
}
```

- 1) Écrivez une classe générique **CPile** qui implémente l'interface **IPile**. Vous stockerez les éléments de la pile dans une `ArrayList <A>`.
- 2) Écrivez un petit programme qui crée et manipule des piles en instanciant la classe générique de différentes façons (par exemple pile de `String`, pile de `Integer`, ...).

## Exercice 2 : Gestion d'une entreprise

On souhaite gérer les employés d'une entreprise.

L'entreprise est composée d'un ensemble de départements et d'employés.

Tout employé est caractérisé par son cin, nom, son salaire et le département dont il est affecté.

```

public class employe {
    private int cin;
    private String nom ;
    private double salaire;
    private int NumDep;
}

```

- 1) Écrivez la classe Département qui permet de gérer l'ensemble de ses employés.

```

import java.util.Set;

public class departement {
    private int idDep;
    private int Capacity;
    Set<employe> LEmployes;
    //constructeurs getters et setters

    /*ajouter un employe au département courant
    * la fonction n'ajoute que les employés ayants un CIN valide >0
    * ne pas oublier de mettre à jour le num dep de l'employe
    */
    public void ajoutEmploye(employe E)
    {
    }

    //retirer un employe du département courant
    public void retirerEmploye(employe E)
    {
    }

    //afficher détails du département
    public void afficheDep()
    {
    }

    //vérifier si un employé dont le cin est donné en paramètre
    // appartient au département courant ou pas
    public boolean existeE(int cin)
    {
        return true;
    }

    //retourner l'employé qui a le plus grand salaire ds le département
    // pensez un utiliser un treeSet
    public employe getEmpSalMax()
    {
        return new employe();
    }
}

```

2) Implémentez une classe entreprise définie ci-dessous

```
import java.util.HashMap;

public class entreprise {
    private String nomEntreprise;
    HashMap<Integer, departement> Liste_Deps;

    //constructeurs, getters et setters

    //ajouter un département donné à l'entreprise
    public void ajoutDep(departement d)
    {

    }

    //retirer un département de l'entreprise
    public void retirerDep(departement d)
    {

    }

    //afficher les détails de l'entreprise
    public void afficheE()
    {

    }

    // transfert d'un employé d'un département à l'autre
    public void UpdateDep(employe E, int idDep)
    {

    }

    //vérifier si un département donné en paramètre appartient à la société ou pas
    public boolean existeD(departement d)
    {
        return true;
    }

    //afficher le département ayant la plus faible capacité
    public void DepMinCapacity()
    {

    }

}
```