

## CORRECTION DEVOIR JAVA

### 3 – API : Compte

C'est un exercice très classique, on vous donne l'API (description) d'une classe et vous devez savoir l'utiliser. La classe `Compte` dispose de 2 constructeurs et de plusieurs méthodes. Les méthodes « void » font un traitement et ne retournent aucun résultat, alors que pour les autres méthodes on indique le type de la valeur retournée.

1- Créer un Objet `Compte` (`unCompte`) pour le client « *HubbleMan* »,

Vous n'avez qu'un seul constructeur qui reçoit le nom du client

```
Compte unCompte = new Compte("HubbleMan");
```

NomDeLaClasse ( `Compte` ) nom de l'objet créé (`unCompte` ) utilisation du constructeur `new` NomDeLaClasse

Ensuite c'est l'utilisation des différentes méthodes

```
unCompte.deposer(300);
```

```
unCompte.retirer(60);
```

```
System.out.println(unCompte.getNom()+" solde "+unCompte.getSolde());
```

```
Compte unCompteEpoque = new Compte("HubbleWife");
```

```
unCompteEpoque.deposer(400) ;
```

2 – Un exercice également très classique en programmation , on vous demandait de savoir parcourir et manipuler une chaîne de caractères, de découper cette chaîne caractère par caractère en commençant par le dernier donc d'utiliser la méthode `charAt ( index du dernier caractère)` puis de concaténer cette chaîne pour l'écrire à l'envers :

On doit commencer par le dernier caractère sachant que les index commencent à 0 , le dernier caractère se trouvera à la longueur de la chaîne - 1 , et ensuite ajouter chaque caractère à la nouvelle chaîne de caractère : `phraseEndroit`

```
String phraseVerlan = " tniop a ritrap tuaf li riruoc ed tres en neiR ";
```

```
int longueurPhrase = phraseVerlan.length();
```

```
String phraseEndroit = "";
```

```
for (int i = longueurPhrase - 1 ; i >= 0 ; i--){  
    char caractere = phraseVerlan.charAt(i);  
    phraseEndroit += caractere ;  
}
```

```
System.out.println(phraseEndroit);
```

### 3 - Exercice, écriture d'une Classe à partir d'un diagramme UML :

On indique dans un premier temps le nom de la classe ( Habitation ) puis les attributs de la classe et leurs types ( nomProprio, surface, nbPieces, piscine) ensuite les méthodes ( le symbole - signifie privé et le symbole + public ) . En général en Java les attributs sont privés et les méthodes publiques. Avec les IDE vous générer une grande partie du code notamment les constructeurs et les getters/setters, vous devez quand même savoir les écrire

```
public class Habitation {

    private String proprietaire;

    private String adresse;

    private double surface;

    private int nbPieces;

    private boolean piscine;


    public Habitation(String proprietaire, String adresse,

        double surface, int nbPieces, boolean piscine) {

        super();

        this.proprietaire = proprietaire;

        this.adresse = adresse;

        this.surface = surface;

        this.nbPieces = nbPieces;

        this.piscine = piscine;

    }


    public double impotBase(){

        return this.surface*2;

    }

}
```

```
public double impotSup(){  
    double impotSupplementaire = this.nbPieces * 50;  
    if(this.piscine)  
        impotSupplementaire+=100;  
  
    return impotSupplementaire ;  
}
```

```
public String toString() {  
    return "HabitationIndividuelle [proprietaire=" + proprietaire  
        + ", adresse=" + adresse + ", surface=" + surface  
        + ", nbPieces=" + nbPieces + ", piscine=" + piscine +  
"]";  
}
```