FICHE 2 : ASSOCIATIONS, ENCAPSULATION

Objectifs

* Comprendre et assimiler la notion d’**encapsulation**.
* Être capable d’écrire les "**getter**" et les "**setter**".
* Être capable d’écrire une méthode **toString()**

Vocabulaire

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| encapsulation | getter | setter | private | public | toString |

Exercices

|  |  |  |
| --- | --- | --- |
| **Attention, à partir cette séance, tous les diagrammes de classes ainsi que tous les** | | |
| **codes Java des exercices veilleront à l’encapsulation. De manière générale, on** | |  |
| **dispose d’attributs private et de méthodes public.** |  |

|  |  |
| --- | --- |
| **Pour tous les exercices, écrivez toujours une classe TestNomDeClasseATester qui** | |
| **validera ce que vous avez écrit.** |  |

# Personne – Adresse

Voici les diagrammes de Personne et Adresse :

Adresse Personne

* rue : String - nom : String
* numero : String - prenom : String
* codePostal : String - anneeNaissance : int
* ville : String - moisNaissance : int

+ Adresse(nRue : String, nNumero : String, - jourNaissance : int nCodePostal : String, nVille : String) - domicile : Adresse

+ getRue() : String, + Personne(nNom : String, nPrenom : String,

+ getNumero() : String nAnneeNaissance : int, nMoisNaissance : int,

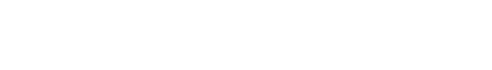
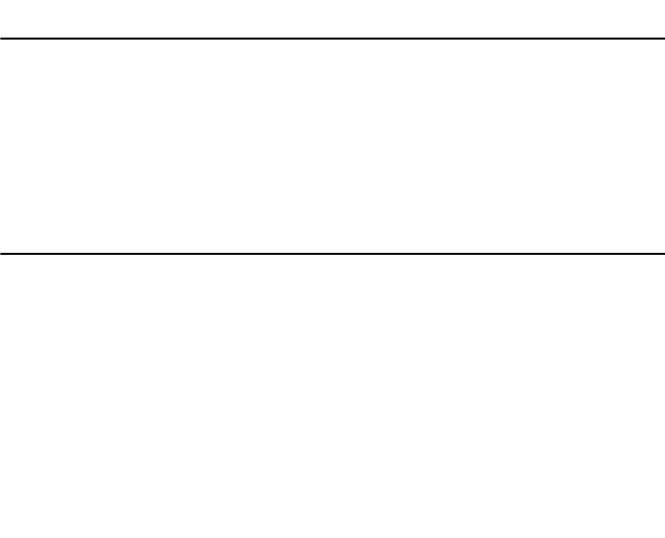
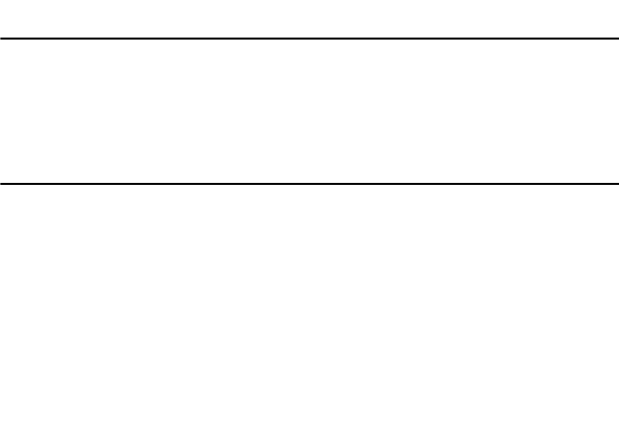
+ getCodePostal() : String nJourNaissance : int, nDomicile : Adresse)

+ getVille() : String + calculerAge() : int

+ toString() : String + calculerAgeEn(an : int) : int

+ fournirDomicile\* () : Adresse

\*À quoi correspond la méthode + fournirDateNaissance() : String fournirDomicile() : Adresse ? + toString() : String



* 1. En vous aidant de la théorie, implémentez ces deux classes en Java.
  2. Écrivez un programme de test en respectant les étapes précisées ci-dessous :
  3. Créer une adresse dans une variable adresse avec comme valeurs Rue de la gare n°34 à 5000 Namur.
  4. Créer une personne dans une variable personne1 ayant comme valeurs Paul Schmidt né le 6 février 1968 et dont le domicile est l’adresse ci-dessus.
  5. Créer une autre personne dans une variable personne2 ayant comme valeurs Valérie Gobert né le 7 mars 1970 et dont le domicile est la même que celle de Paul. Ils habitent ensemble.
  6. Affichez les informations des deux personnes.

* 1. Représentez les objets en mémoire à la fin de ce programme :

|  |  |
| --- | --- |
| schmidtPaul : Personne | gobertValerie : Personne |
| nom = « schmidt »  prenom = « Paul »  annee = 1968  mois = 2  jour = 6  adresse = adr | nom = « Gobert »  prenom = « Valerie »  annee = 1970  mois = 3  jour = 7  adresse = adr |

|  |
| --- |
| adr : Adresse |
| rue = « Rue de la gare »  numero = « 34 »  codePostal = « 5000 »  ville = « Namur » |

# Date

On définit la classe Date au moyen du diagramme UML suivant :

|  |
| --- |
| **Date** |
| * annee : int * mois :int * jour : int |
| Date(annee : int, mois : int, jour : int)  getAnnee() : int  getMois() : int  getJour() : int  toString() : String |

1. Complétez l’UML de la classe Date :
   * ajoutez un constructeur ;
   * ajoutez des getters ;
   * ajoutez une méthode toString qui renvoie la date au format «j/m/a ».
2. Implémentez la classe Date en Java.
3. Testez votre classe en créant 2 instances de Date et en les affichant.

# Personne

Reprenez la classe Personne et modifiez-la de sorte que la personne conserve un attribut de type Date pour sa date de naissance.

1. Modifiez d’abord votre diagramme UML :
   * Ajoutez les getters associés aux attributs domicile et dateDeNaissance.
   * Adaptez le constructeur de la classe Personne.
   * Ajoutez une méthode qui permet de changer l’adresse du domicile. Cette méthode s’intitule demenager et prend en paramètres une rue, un numéro, un code postal et une ville. Elle veille à changer le domicile de la personne. Cette méthode ne renvoie rien.

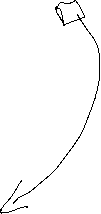
|  |  |  |
| --- | --- | --- |
| Personne | Adresse | Date |
| nom : String  prenom : String  date : Date  adresse : Adresse | rue : String  numero : String  codePostal :String  ville : String | annee : jnt  mois : int  jour : int |
| Personne(nom : String, prenom : String, date : Date, adresse : Adresse)  getAdresse() : Adresse  getDate() : Date  déménager(rue : String, numero : String, codePostal : String, ville : String) : void  toString() : String | Adresse(rue : String, numero : String, codePostal : String, ville : String)  toString() : String | Date(annee : int, mois : int, jour : int)  toString() : String |

1. Implémentez ces changements en Java. Il faudra également modifier votre programme de test.

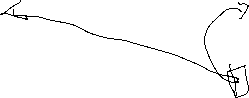
# Des personnes qui déménagent

1. Regardez la classe TestDemenagement et représentez en mémoire les objets créés dans TestDemenagement**.**

|  |  |
| --- | --- |
| dateBrad : Date | dateAngelina : Date |
| annee = 1963  mois = 12  jour = 18 | annee = 1975  mois = 6  jour = 4 |



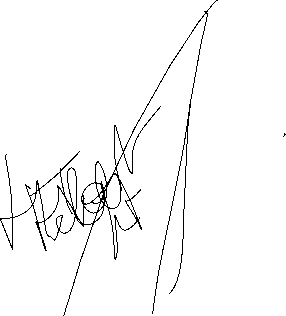
|  |  |
| --- | --- |
| brad : Personne | angelina : Personne |
| nom = « Pitt »  prenom = « Brad »  adresse = adr  date : dateBrad | nom : « Jolie »  prenom : « Angelina »  adresse : adr  date : dateAngelina |



|  |
| --- |
| adr : Adresse |
| rue : « Rue des stars »  numero : « 52 »  code : « 1000 »  ville : « Bruxelles » |



1. Quel défaut présente l’implémentation de la classe Personne telle que fournie ?



On définit deux fois la même adresse.

1. Comment corrigeriez-vous l’implémentation de la classe Personne ?

En changeant la méthode déménager et en mettant comme parametre une adresse.

demenager(Adresse adresse) {

}

# Point

Un point dans un plan est caractérisé par ses coordonnées réelles x et y. La classe Point fournit aussi :

* un constructeur ayant pour paramètres les coordonnées x, y dans cet ordre ;
* deux getters permettant de récupérer les valeurs de x et de y ;
* deux setters permettant de modifier les valeurs de x et de y ;
* une méthode toString qui renvoie les coordonnées x et y au format (x,y).

1. Donnez l’UML de la classe Point

|  |
| --- |
| Point |
| x : double  y : double |
| Point(x : double, y : double)  getX() : double  getY() : double  setX() : double  setY() : double  toString() : String |

1. Implémentez cette classe en java et testez-la.