Td1 – Java Avancée

Exercice 1 :

2- L’utilisation de « sysout » et ctrl+space permet de remplacer directement « sysout » par « System.***out***.println(); »

3- L’utilisation de cette commande permet de générer le corps de la méthode toString()

4 - L’utilisation de cette commande permet de générer le corps de la méthode main()

5 - Ceci nous propose de générer le constructeur.

Si on utilise « set » ctrl+ space L’utilisation de cette commande permet de générer le corps de la méthode setFoo(foo)

L’utilisation de ctrl + space permet donc de faire de l’auto-complétion

6- Ceci permet de renommer notre classe. Ceci nous permet de renommer le texte sélectionner partout où il est utilisé

Exercice 2 :

1-Il existe le constructeur par défaut qui permet de nous permet de créer un objet Point grâce à « new Point () » et initialise x et y à 0. Ceci fonctionne aussi car on accède au valeur x et y en ayant définit un Object avant, permettant de lire ces valeurs avec p.x – p.Y. De plus, les attributs sont en «private» donc on y a accès seulement dans la classe ou ils sont définis.

2-

Ceci n’est pas possible car on essaye d’accéder à des attributs qui ne sont pas dans notre classe. Il faudrait par exemple hériter de cette classe et les mettre « protected « aux attributs de la classe point pour que ceci fonctionne.

Sinon, on peut mettre en place des getters pour la classe Point et nous utiliserons les getters pour lire les valeurs des points x et y.

3.

Pour pouvoir protéger nos attributs et qu’ils ne puissent pas être modifié hors de la classe par n’importe qui. On respecte aussi l’encapsulation.

4.

Un accesseur est une méthode qui nous permet de retourner la valeur d’un attribut.

Comme getX().On peut les générer automatiquement et les utiliser.

Ici, il est judicieux dans utiliser pour résoudre notre problème.

5.

Le problème est que nous avons défini un constructeur avec 2 paramètres, il n’existe donc plus le constructeur par défaut sans paramètre. Lorsque l’on appelle « new point () » il manque les paramètres ce qui provoque une erreur.

6.

7.

On peut créer une variable « static » qui s’incrémente à chaque appel du constructeur.

8.

Si le constructeur est suivis de deux paramètres il sait qu’il devra appeler le Point(x,y). Sinon, s’il reçoit un objet en paramètre il utilisera le Point(p2)

Exercice 3 :

1.

Le code affiche « true, false ». Ceci est dû au fait que p1 et p2 partage la même adresse de stockage. Tandis que p3 fait appel au constructeur, ce qui engendre une nouvelle allocution d’un espace mémoire. Le rendant différent de p1 et p2 d’où la réponse « true, false ».

3.

La méthode indexOf() renvoie l’index du première occurrence d’un élément dans l’ArrayList, ou -1 si l’élément n’existe pas.

Le problème est que nous ajoutons le Point p1, mais puisqu’il partage le même espace mémoire, il n’y a pas de distinction entre p1 et p2. D’où la présence de l’indice 0 et non -1.

Exercice 4

Les tableaux en Java sont de tailles fixes, si on ajoute plus de point que de place on ne pourra pas les ajouter. Il est recommandé d’utiliser des collections qui sont plus modulables aux besoins.

5.

Si nous mettons « null » et non un point, une exception de type « nullPointerException() »apparait et empêche la compilation.

6. Les tableaux en Java sont très contraignants. Il est donc plus adapté d’utiliser des collections comme linkedList ou ArrayList par exemple. L’un des avantages des collections est qu’elle dispose de fonction déjà coder comme add, contains, size permettant de nous faciliter leurs utilisations.

Exercice 5.

Le problème avec ce code est nous voulons utiliser un objet qui est mutable, il est donc modifié à chaque fois que l’on appelle la fonction translate (). Pour modifier ce comportement il faudrait utiliser une copie défensive pour modifier les valeurs de X et Y d’un point sans modifier tous les objets utilisant ce point.