TP JAVA n°1
Création d'une Classe

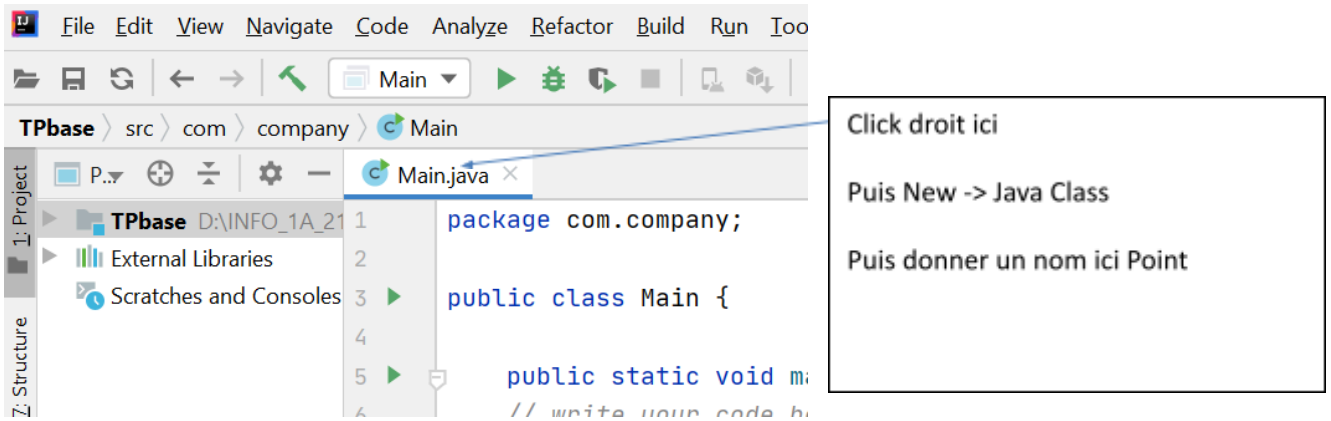
Exercice 1 : La Classe Point

Pour créer le projet:

- Lancer IntelliJ Idea
- File > New > project
- Vérifier que la fenêtre Project SK contient bien une version de SDK sinon choisir grace à la flèche à droite
- Next
- Sélectionner la case "Create project from template" puis next
- Donner un nom à son projet dans project name, puis finish. Le projet est alors créé et le fichier Main.java apparait à l'écran avec la fonction main déjà déclarée :

```
public static void main(String[] args) {
// write your code here
   }
// signifie ligne de commentaire
```

Pour créer une nouvelle Classe :



Un point est caractérisé par ses deux coordonnées, représentées par deux nombres de type double.

1) Créer une Classe Point.

Point
- x : double - y :double

+ Point (double, double) + Point (Point) + symetrique() : void + distance(Point) : double + getX () : double + getY () : double + setX (double): void + setY (double): void + toString () : String

2) Créez un **constructeur** qui affecte les valeurs qui lui sont transmises (arguments du constructeur) aux deux attributs de l'objet en cours de création.
3) Créez un constructeur de **recopie**.
4) Générer automatiquement la méthode **toString()** : positionner le curseur à l'endroit où l'on veut que la méthode soit générée, click droit -> generate -> toString()
5) Dans le programme principal de la classe **Main**, créez un premier point p1 de coordonnées 3 et 4, et un second point p2 de coordonnées -1 et 2. Affichez les avec System.out.println().
6) Créez un point pp1 clone de p1. Affichez-le.
7) Programmez la méthode **symetrique**() qui transforme l'objet courant en son symétrique par rapport à l'origine. Testez en appliquant à p1 et vérifiez en affichant le point.
8) Programmez une méthode **distance**(Point) qui retourne la distance entre deux Points. Utilisez **Math.sqrt(a)** qui retourne la racine carrée de a et **Math.pow(a,b)** qui retourne la valeur de a élevé à la puissance b (double). Testez votre méthode dans la Classe **Main**.
9) Créez, dans le programme principal, un **tableau** de trois points, initialisez le et affichez-le à l'aide d'une boucle **for**.
10) Complétez la classe **Point** en générant automatiquement les **getters/setters**. Modifiez pp1 et affichez p1.

Exercice 2 : La Classe Segment

La classe **Segment** possède (pour l'instant) deux attributs privés de type Point représentant l'origine et l'extrémité du segment et un attribut de classe nbreSegments :

Segment
- origine : Point - extremite : Point - coeffA : double

- coeffB :double - nbreSegments : int
+ Segment (Point, Point) + Segment(Segment) + translation(Point) : void + longueur () : double + toString () : String - calculCoeffA () : double - calculCoeffB () : double + appartient (Point) : boolean + getNbreSegments () : int

1) Créez la Classe **Segment**.
2) Créez un **constructeur** qui affecte les 2 points qui lui sont transmis (arguments du constructeur) aux deux attributs de l'objet en cours de création. Générez la méthode **toString()**. Créez un segment S1 à partir de deux points et affichez-le.
3) Créez un constructeur de **recopie**, créez un segment S2 (clone de S1) dans le main en utilisant le constructeur de recopie et affichez-le.
4) Écrivez la méthode **translation**(Point) qui a en argument la nouvelle origine du segment et translate celui-ci. Dans le **Main** créez un point P3 et appliquer à S1 la méthode translation ayant en argument P3.
5) Écrivez la méthode **longueur**() qui retourne la longueur du segment.(n'oubliez pas que vous avez la classe **Point**). Dans le **Main** affichez la longueur de S1.
6) On ajoute à la classe **Segment** deux attributs privés. Ils représentent les coefficients *a* et *b* de la droite d'équation *ax+b* qui contient le Segment. Il faudra modifier les constructeurs de la classe Segment pour qu'ils initialisent ces deux nouveaux attributs aux bonnes valeurs. On pourra écrire deux méthodes, **calculCoeffA()** et **calculCoeffB()**, renvoyant pour l'une le coefficient *a* et pour l'autre le coefficient *b* (indication : ces deux méthodes ne seront appelées que dans la classe Segment). Les constructeurs devront alors appeler ces méthodes ainsi que toute méthode modifiant un segment. Testez dans le **Main**.
7) Ajoutez ensuite à la classe **Segment** la méthode **appartient(Point P)** qui teste si un **Point** passé en argument est sur la droite contenant le Segment. Cette méthode retournera un booléen. Testez dans le main : créez un point en (2, 10) et un autre en (-1 et -2). Créez le segment seg à partir de ces deux points. Soit le point pp (4, 2), testez si pp appartient à seg. Soit le point pp2(-2, -6); testez si pp2 appartient à seg.
8) Ajoutez la méthode **getNbreSegments()** qui est une méthode de classe publique et qui retourne un entier. Utilisez-la quand vous créez un segment.

Exercice 3 : La Classe Polygone

La classe **Polygone** permet de stocker et de manipuler un **ArrayList** de points. Voici une liste non exhaustive des méthodes utiles de **ArrayList**:

- add()
- size()
- contains(p)
- get(i) permet de lire l'élément i de notre **ArrayList**.

1) Créez la Classe **Polygone** ainsi que son **constructeur** qui créera un tableau de 0 point.
2) Générez automatiquement une méthode **toString()**.
3) Écrivez la méthode **addPoint(Point)** qui ajoutera un point à ceux déjà présents. Testez la.
4) Écrivez la méthode **estDansPolygone(Point)** qui retourne true si le point passé en argument appartient à la liste de points du polygone. Tester avec un des points se trouvant dans le polygone, puis faire un clone de ce point puis tester.
5) Modifier **estDansPolygone()** afin que pour le clone précédent la réponse soit true.
6) Modifiez **addPoint()** de sorte qu'il soit impossible d'ajouter un point déjà présent dans le tableau.

*Exercice 4 : La Classe Personne

On veut modéliser des Personnes composées des attributs suivants :

```
private String Nom;
private String Prenom;
private String NumSecu;
private int anneeNaissance;
private char sexe;
private int deptNaissance;
```

1) Créer la classe **Personne**,
2) Écrire ensuite un **constructeur** permettant de ne saisir que le nom et le prénom et initialisant les autres attributs de manière cohérente (valeurs "neutres")
3) Générer la méthode **toString()** , plus la modifier afin qu'elle n'affiche que les champs vraiment renseignés.
4) Écrire ensuite un **constructeur** permettant de ne saisir que le nom , le prénom , le numéro de sécurité sociale et l'adresse.
5) Écrire une méthode **anneeNaissance()** à usage restreint (juste dans la classe) permettant de calculer l'année de naissance en connaissant le numéro de sécurité sociale.
6) Ecrire une méthode **sexe()** à usage restreint (juste dans la classe) permettant de calculer le sexe en connaissant le numéro de sécurité sociale.
7) Écrire une méthode **deptNaissance()** à usage restreint (juste dans la classe) permettant de calculer le département de naissance en connaissant le numéro de sécurité sociale.
8) Compléter le deuxième **constructeur** en initialisant les attributs pouvant être déduits du numéro de sécurité sociale.

9) Écrire les **setters** et les **getters** sachant que certains n'ont pas lieu d'être. Compléter le setter du numéro de sécurité sociale afin qu'il permette aussi de mettre à jour les attributs qui peuvent en être déduits.
10) Écrire une méthode **calculAge()**.
11) Ecrire une classe **CompteBanq()** avec les attributs :

```
private String num;
private Personne client;
private float solde;
private float decouvertAutorise;
private String code1;
private String code2;
```

12) Écrire un **constructeur** ne saisissant que le numéro, le client et les codes, initialisant les autres attributs à 0.
13) Écrire les **setters** et **getters** en sachant qu'on ne doit pouvoir modifier que le solde et le découvert autorisé et ceci en donnant le code1 pour la première opération et le code 2 pour la deuxième. On ne doit pas pouvoir non plus modifier les codes.
14) Écrire une méthode **versement()** pour laquelle il faudra bien sûr connaître le code
15) Écrire une méthode **retrait()** pour laquelle il faudra connaître le code 1 et si la somme à retirer est supérieure au solde il faut alors vérifier si l'opération peut être effectuée en regardant le découvert autorisé.
16) Au niveau du **Main** il faut arriver, après création de la personne et du compte, à donner un nom, afficher le solde, proposer une versement ou un retrait, l'effecteur et afficher le nouveau solde.