TD 4

1. Généricité et figures 2010 (4 points)

On souhaite donner la possibilité d'uniformiser les sommets d'un polygone en forçant ceux-ci à être des pointNommé ou des pointPondéré, … Pour cela on utilise une version générique de la classe polygone. Les sommets sont stockés dans une liste chaînée et le paramètre générique doit garantir que les sommets soient des points.

1. Ecrivez l'entête de la classe polygone générique
2. Ecrivez la déclaration de la liste chaînée
3. Ecrivez le constructeur de la classe polygone
4. Ecrivez la méthode getPoints() qui renvoie la liste de points

2. Extraction d'une liste de points 2011 (5 points)

On souhaite créer une liste de points à partir de toutes les figures créées à l'aide de l'éditeur de figure.

1. On ajoute une méthode getPoints() dans chaque figure qui renvoie la liste des points de cette figure. Codez cette méthode pour les classes Point, Segment, Polygone et Cercle
2. Codez la méthode createListe() prenant en paramètre une liste de Figure et permettant de construire la liste des Points

3. SauveQuiPeut 2011 (5 points)

On souhaite coder de deux manières la méthode SauveQuiPeut dont l'entête est : *public void SauveQuiPeut(Collection Col, ObjectOutputStream Out) { …}* et qui permet de sauver tous les objets serialisables de *Col* dans le flux Out.

1. Coder cette méthode en utilisant instanceOf
2. Coder cette méthode en utilisant les exceptions

4. Les exceptions 2010 (6 points)

On suppose définie une liste de figures : *LinkedList<Figure> figures ;*

On souhaite afficher la liste des noms des figures. Certaines figures possèdent un nom (cercleNommé, polygoneNommé, …) et d'autres pas (point, …). Les figures possédant un nom implémentent l'interface *Nomme* qui possède une méthode *public void String getNom()*.

1. En utilisant *instance of*, coder la méthode *public String Noms()* qui concatène les noms de toutes les figures qui en possède un.
2. En utilisant les exceptions (et sans utiliser *instance of*), coder la même méthode. On rappelle que lorsqu'un appel est fait sur une méthode qui n'est pas présente dans une classe, l'exception NoSuchMethodException est levée.

5. Les structures de données 2010 (6 points)

1. Coder une méthode prenant en paramètre : une structure de donnée ne pouvant contenir que des *Figures* et une *Figure F*. La méthode renvoie le nombre de Figures de même type que *F* contenues dans la structure. La méthode getClass() est présente dans toutes les classes java et renvoie la classe de l'objet appelant la méthode (exemple : Integer I ; I.getClass() renvoie Integer).
2. On souhaite stocker les Figures dans une liste triée (*sortedSet*). Quelle interface doivent implémenter les figures pour que cela puisse fonctionner ?
3. On suppose codée la méthode *getPoids()* qui renvoie le poids de la figure (nombre de points de celle-ci par exemple). Coder la méthode présente dans l'interface précédente et permettant de faire la comparaison entre deux figure en utilisant le poids.