Master Informatique **Compilation – J1IN7M13**
Université Bordeaux 1 **2013 – 2014**

TD N^o4 Arbre de syntaxe abstraite

Le but du TD est de construire un arbre de syntaxe abstraite pour une expression.
Nous avons :

1. Les nombres entiers, les réels et complexes
2. Les valeurs booléennes `true, false`
3. Les opérateurs arithmétiques `+, -, *, /, %`
4. Les signes de comparaison `==, !=, <, <=, >, >=`
5. les opérateurs booléens `&&, ||, !, ->, <->`
6. l'opérateur ternaire `X?Y:Z` qui vaut `Y` si `X` est vrai, `Y` sinon

Sur la page des TDs, récupérer les fichiers Java suivants :
- `ArbreSyntAbstr.java` (incomplet) qui implémente les arbres de syntaxe abstraite
- `EnumTag.java` (incomplet) qui liste les étiquettes des arbres de syntaxe abstraite

Questions

1. Réaliser l'analyseur syntaxique en respectant les règles de priorité et d'associativité des différents opérateurs.
2. On souhaite maintenant construire un arbre de syntaxe abstraite pour une expression. Il s'agit d'un arbre binaire dont les sommets sont étiquetés par des éléments ("tags") de la classe `EnumTag.java`.

   Compléter la classe `ArbreSyntAbstr.java` et l'analyseur syntaxique pour construire un arbre de syntaxe abstraite pour les entiers, les réels, les complexes, les valeurs booléennes, les expressions arithmétiques et logiques.

3. On tentera maintenant d'implémenter des sémantiques différentes pour les calculs sur les entiers, les réels, les complexes et les booléens. Il s'agira d'associer à tout arbre de syntaxe abstraite le type correspondant à l'expression.