Complément de programmation

----- Maven ------

Générer les POM avec archetype :

```
mvn archetype : generate \
-DarchetypeGroupeId=org.apache.maven.archetypes \
-DarchetypeArtifactId=maven-archetype-quickstart \
-DinteractiveMode=false \
-DartifactId=MonProjey \
-Dversion=1.0-SNAPSHOT
```

Dépendance :

```
<dependencies>
 <dependency>
  <groupId>junit</groupId>
  <artifactId> junit</artifactId>
  <version>v</version>
  <scope>test</scope>
 </depedency>
</depedencies>
```

Tests :

```
mvn test
```

JAR :

```
mvn jar
```

scope : but spécifique/tâche à exécuter

fixer versions sources et binaires :

```
<properties>
 <maven.compiler.source>11</maven.compile.source>
 <maven.compile.target>11</maven.compiler.target>
</properties>
```

----- git -----

```
git init
git branch [br_name]
git checkout [br_name]
git add [files]
git commit -m [message]
git push
```

fusionner :

```
git checkout [master/main]
git merge [br_name]
```

```
git tag -a [tag_name] -m'[tag_str]' => place le tag sur la HEAD
git push origin [tag_name]
git remote add origin [github]
git pull
```

Arborescence projet :

```
src/main/mon/package/source
src/test/mon/package/test
```

----- Java -----

Méthodes courantes à @Override :

- String toString()
- boolean equals(Object obj) et

int hashCode(){ exemple : return XOR entre les deux points d'un Rectangle } => doivent être redéfinies ensemble

public interface Comparable<T> à implements et méthode public int compareTo(T o) à Override => return -1,0,1 si inf., égal ou sup.
public interface Comparator<T> ---------- int compare (T o1, T o2) ----------

----- JUnit 4 -----

assertThrows :

```
@Test(expected = boringException.class)
public void boringTest{ } ; => pass : exception thrown, fail : exception not thrown
```