

## TD n°15 – Héritage, polymorphisme, collections

### Le projet Pokemons

#### Rappel des consignes

- Ce projet se réalise sur plusieurs séances, par groupe de 2 étudiants.
- Votre enseignant responsable en TD vous donnera ses indications de rendu.
- Le code doit être correctement commenté (pas trop, ni trop peu), robuste (gestion des exceptions), et associé à des TestCase JUnit.
- Un travail minimum est demandé lors de chaque séance de TD. La réalisation de ce travail est obligatoire et sa bonne réalisation assure la moyenne au groupe.
- Des extensions seront suggérées ou laissées au libre choix des étudiants. Toute extension correctement implémentée améliore la note du groupe.
- Au contraire, tout code similaire entre groupes diminue la note de tous les groupes concernés.
- Lors du DS, une partie des questions portera sur ce projet.

#### Les parties

- 3<sup>ème</sup> partie obligatoire

#### Le fichier des Pokemons

Un fichier texte nommé *ListePokemon.txt* est disponible en annexe de ce sujet. Il contient sur chaque ligne la description d'un Pokemon :

nom	type	taille	poids	pv	pc
-----	------	--------	-------	----	----

En fonction du type, les paramètres additionnels sont ajoutés à la suite de la puissance de combat sur la ligne. Les données sont séparées entre elles par un espace.

- Dans une classe `Partie`, définir une méthode qui va se charger d'initialiser une liste de Pokemons instanciés à partir du fichier texte, en objets référençant le bon type avec tous ses attributs bien remplis.
- Pour chaque Pokemon, calculer sa vitesse, et vérifier s'il est vivant.
- Calculer la vitesse moyenne de la collection initiale ainsi que la vitesse moyenne de chaque type contenu.

#### Jeu à 2 joueurs

Modifier la classe `Partie` afin de lancer une partie « aléatoire » entre 2 joueurs dans laquelle :

- Chaque joueur démarre avec 10 Pokemons dans sa collection. Ces Pokemons sont choisis de manière aléatoire dans la liste initiale des Pokemons enregistrée à partir du fichier texte.
  - Trier les joueurs et leurs collections de Pokemons dans l'ordre naturel défini au TD précédent.
  - Calculer la vitesse moyenne de la collection de chaque joueur et de chaque type de Pokemon contenu dans sa collection.

- Les deux joueurs s'attaquent séquentiellement (peu importe qui attaque qui) et choisissent aléatoirement un Pokemon dans leur collection respective pour combattre.
- La partie s'arrête lorsqu'un joueur n'a plus de Pokemon vivant dans sa collection.
- Afficher le résultat de la partie.

➤ Extensions possibles

*Varier les Pokemons*

- Pour construire la collection initiale des joueurs, les données fournies dans le fichier peuvent être modifiées comme suit :
  - La taille, le poids, les points de vie et les points de combat du Pokemon tiré aléatoirement peuvent être modifiés de + ou – 20% pour générer plus de Pokemons différents.

*Enrichir les modes de jeu.*

Vous pouvez par exemple définir des parties de la manière suivante :

- Partie « ordonnée » :
  - Au premier tour, le joueur qui attaque est le premier dans l'ordre naturel des joueurs.
  - Puis, le joueur qui a perdu le moins de points de vie au combat précédent est celui qui attaque au tour suivant.
- Partie « plus fort d'abord » où le Pokemon qui attaque à chaque tour est celui qui a la plus grande puissance parmi les Pokemons toujours en vie de la collection du joueur (en les comparant et/ou triant la collection avec l'ordre naturel ou un ordre temporaire).
- Partie « plus vivant d'abord » où le principe est le même que ci-dessus mais le critère de choix est le nombre de points de vie et non la puissance.
- Partie « interactive » où chaque joueur décide à chaque tour quel Pokemon il va utiliser pour attaquer l'adversaire.
- Partie à  $n$  joueurs (dans tous les modes) : quelle structure de données utiliser pour les joueurs ?