Projet POO Java

Vous allez devoir réaliser une application Java tout en respectant :

- les conventions du langage Java (noms de classe, méthode, variable, package)
- les fondements de la programmation orientée objet (encapsulation, héritage, polymorphisme, classes abstraites, interfaces)
- l'organisation de votre code
- l'utilisation des énumérations
- l'utilisation des tableaux et/ou collections
- l'utilisation des exceptions

Contexte

La société Ibisoft a besoin de vous pour son nouveau projet rétro d'aventure **Donjon Efrei**. Ibisoft a déjà confié certaines parties de l'application à d'autres équipes (carte du donjon, déplacement…) et vous confie **le développement du coeur des combats**.

Pour cela, la société vous fournit :

- **l'objectif du programme** : comment il doit se dérouler à l'exécution
- **un cahier des charges** : une liste des personnages, monstres, armes et armures ainsi que de leurs caractéristiques nécessaires pour une première version avec les règles fonctionnelles associées.

En tant que bon développeur, vous vous assurerez de fournir un code **testable**, **maintenable** et **facile à évoluer**.

Objectif du programme

Le programme doit permettre à l'utilisateur de :

- Sélectionner un héros et/ou une arme et/ou armure (vous pouvez faire la sélection d'une arme/armure ou directement les créer quand vous créez votre héros)
- Un argument sera passé en paramètre de l'application pour définir **le nombre de monstres à combattre** et le Monstre sera généré aléatoirement par une logique de votre choix (par exemple avoir plus de chances de tomber sur un Gobelin qu'un Dragon…)
- Pour chaque combat :
    - **Tour du héros** : une action sera proposée à l'utilisateur selon le héros choisi ((A) pour Attaquer, (M) pour une attaque Magique, (S) pour se soigner, (N) pour narguer le monstre, (F) pour fuir)
        - *Attaquer* : les dégâts infligés tiennent compte de la somme de l'attaque du héros et des

dégâts de l'arme. Les dégâts sont doublés si le type élémentaire de l'arme est efficace sur celui du Monstre (voir les règles de dominance des éléments dans la section IMPORTANT du TypeElementaire). Si ce n'est pas efficace, l'attaque est réduite de 2.

- *Magie* : voir la logique dans l'entité Magicien
- *Soigner* : tous les points de vie du personnage sont restaurés et cela coûte 5 points de magie
- *Narguer* : affiche un message de moquerie au Monstre
- *Fuir* : 2 chances sur 5 de pouvoir fuir le combat. Si la fuite est OK, on passe au prochain monstre s'il y en a sinon le jeu est terminé. Si la fuite est KO, le combat continue et c'est au tour du monstre

◦ **Tour du monstre** : le monstre ne fera qu'attaquer ou lancer sa capacité spéciale. Les chances pour un Monstre de lancer sa capacité spéciale est de 1/10.

- **pour l'attaque de base seulement** : les dégâts sont doublés si le type élémentaire du Monstre est efficace sur l'armure du héros (voir les règles de dominance des éléments dans la section IMPORTANT du TypeElementaire). Si ce n'est pas efficace, l'attaque est réduite de 2.

- Fin du combat :
  - si tous les Monstres ont été vaincus, afficher un message de victoire avec le nombre de Monstres vaincus sur le nombre total de Monstres (les fuites ne comptent pas !)
  - si le héros a été vaincu, afficher un message de défaite avec le nombre de Monstres vaincus sur le nombre total de Monstres (les fuites ne comptent pas !)

IMPORTANT	on s'attend évidemment que l'expérience utilisateur soit la plus optimale possible, c'est-à-dire que le programme soit accompagné de messages clairs (par exemple pour sélectionner un héros ou la liste des actions possibles) mais aussi la conséquence de chaque action (caractéristiques du héros ou du monstre lorsqu'il reçoit des dégâts ou se soignent…)

Cahier des charges

Entités

Personnage

- Nom (minimum 4 lettres)
- Prénom (maximum 13 lettres)
- Points de vie (0 à 100)
- Points de magie (0 à 50)
- Attaque
- Défense
- Arme

- Armure
- un Personnage peut narguer un Monstre
  - affiche juste un message pour se moquer du Monstre
- un Personnage peut fuir un combat (il n'y a pas de honte à ça !)
  - il a 2 chances sur 5 de pouvoir fuir le combat

Combattant

- est un Personnage
- à la création, les points de magie d'un Combattant doivent être compris entre 0 et 25 (minimum 0 et maximum 25)

Barbare

- est un Combattant
- à la création, l'attaque d'un Barbare doit être compris entre 10 et 15 (minimum 10 et maximum 15)
- à la création, la défense d'un Barbare doit être compris entre 5 et 10 (minimum 5 et maximum 10)

Paladin

- est un Combattant
- a le pouvoir de se soigner
  - il restaure complètement ses points de vie
  - le sort de soin lui coûte 5 points de magie
- à la création, l'attaque d'un Paladin doit être compris entre 5 et 10 (minimum 5 et maximum 10)
- à la création, la défense d'un Paladin doit être compris entre 10 et 15 (minimum 10 et maximum 15)

Amazone

- est un Combattant
- à la création, l'attaque d'une Amazone doit être compris entre 7 et 12 (minimum 7 et maximum 12)
- à la création, la défense d'un Amazone doit être compris entre 7 et 12 (minimum 7 et maximum 12)

Magicien

- est un Personnage
- à la création, les points de vie d'un Magicien doivent être compris entre 1 et 70 (minimum 1 et maximum 70)
- peut jeter un sort qui enlève 7 points de magie et qui :
  - triple les dégâts infligés au Monstre si l'arme a un type élémentaire dominant sur le type élémentaire du monstre (voir les règles de dominance des éléments dans la section IMPORTANT du TypeElementaire)

    - double les dégâts infligés au Monstre sinon

Pretre

- est un Magicien
- a le pouvoir de se soigner
    - il restaure complètement ses points de vie
    - le sort de soin lui coûte 5 points de magie

Mage

- est un Magicien

Monstre

- Nom
- Prénom
- Points de vie (0 à 100)
- Attaque
- Défense
- Type élémentaire
- Capacité spéciale

Gobelin

- est un Monstre
- à la création, les points de vie d'un Gobelin doivent être compris entre 1 et 50 (minimum 1 et maximum 50)
- à la création, l'attaque d'un Gobelin doit être compris entre 3 et 10 (minimum 3 et maximum 10)
- sa capacité spéciale, nommée "Empaler" qui ajoute 10 de dégâts à son attaque principale

Vampire

- est un Monstre
- à la création, les points de vie d'un Vampire doivent être compris entre 30 et 70 (minimum 30 et maximum 70)
- à la création, l'attaque d'un Vampire doit être compris entre 10 et 20 (minimum 10 et maximum 20)
- sa capacité spéciale, nommée "Absorption" lui permet de se soigner de 33% par rapport aux dégâts infligés aux héros en arrondissant au nombre supérieur (ex: si le Vampire active sa capacité spéciale et fait un dégât de 20, alors il se régénère de 7 points de vie)

Dragon

- est un Monstre
- à la création, les points de vie d'un Dragon doivent être compris entre 70 et 100 (minimum 70 et maximum 100)
- à la création, l'attaque d'un Dragon doit être compris entre 20 et 40 (minimum 20 et maximum 40)

- sa capacité spéciale, nommée "Souffle", fait descendre les points de vie du héros jusqu'à 1

Arme

- Nom
- Degats
- Type élémentaire

Armure

- Nom
- Protection
- Type élémentaire

IMPORTANT	la société Ibisoft vous laisse une chance de montrer votre créativité en appliquant obligatoirement une règle sur les armes et/ou les armures. Un exemple pourrait être ajouter un taux de coup critique sur l'Arme ou alors un indice d'usage/réparabilité au niveau de l'Armure. Mais bien évidemment, vous pouvez choisir votre propre règle.

TypeElementaire

- FEU
- BOIS
- EAU
- METAL
- TERRE

IMPORTANT	l'interaction entre les 5 éléments est la suivante : • le BOIS paralyse la TERRE • le FEU fait fondre le METAL • la TERRE emprisonne l'EAU • le METAL coupe le BOIS • l'EAU eteint le FEU

Testabilité

Un projet doit être testable, normalement on utilise des tests unitaires, intégration mais vous verrez ça plus tard. Pour le moment, la société Ibisoft vous demande :

- d'inclure une classe de test pour chaque entité (TestPersonnage, TestMonstre, TestArme, TestArmure)
  - chaque entité sera décrite sous la forme suivante : "[Nom de la classe] - {message/caractéristiques}"

Documentation

Un projet doit être bien documenté :

- inclure à la racine du projet un fichier README.md qui donne :
  - votre nom et prénom
  - la version de Java utilisée
  - les instructions nécessaires pour lancer votre application sur un terminal en ligne de commande
  - la règle métier que vous avez appliquée concernant l'entité Arme ou Armure
- écrire des commentaires sur des méthodes de votre choix afin de pouvoir générer une Javadoc
  - commenter la logique dans le bout de code que vous utilisez pour générer aléatoirement un monstre
  - commenter d'autres bouts de code ou méthodes qui vous semblent pertinents

Livrable attendu (ce qu'il faut rendre)

- **Faire un zip du projet à la fin de la première séance**
- **Faire un zip du projet à la fin de la deuxième séance**

Améliorations possibles (si vous avez le temps)

1. Réaliser un diagramme de classes avec la syntaxe UML (via un logiciel comme LucidChart ou alors sur paper) : https://www.lucidchart.com/pages/fr/diagramme-de-classes-uml
2. Garantir l'unicité de vos objets/instances dans les cas suivants :
   - on ne peut pas avoir 2 personnages avec le même nom, même prénom, même attaque et même défense
   - on ne peut pas avoir 2 armes avec le même nom
   - on ne peut pas avoir 2 armures avec le même nom
   - on ne peut pas avoir 2 monstres avec le même nom, même prénom et même type élémentaire
3. Utiliser le design pattern Builder pour améliorer la construction des objets : https://refactoring.guru/fr/design-patterns/builder
4. Permettre aux héros d'utiliser des Accessoires (potions, etc...)
5. Améliorer la logique concernant les sorts des Magiciens
6. Permettre aux Monstres de s'équiper d'armes ou d'armures
7. Permettre d'avoir non pas un héros mais une équipe de héros qui affrontent un Monstre
8. Utiliser JavaFX pour faire une interface au lieu d'utiliser la console de l'IDE/terminal