***Documentation : Projet Aventurier***

**Bibliothèque de classes :**

1. **Case**

Les éléments qui composent cette classe :

* Position X
* Position Y
* Locked (indique si la case en question est verrouillée et donc impossible de placer l’aventurier sur celle-ci) => Identifié par : #
* Aventurier (indique si l’aventurier est sur cette case) => Identifié par : @
* Used (indique si l’aventurier à déjà utilisé cette case) => Identifié par : \_

Une image contenant texte

Description générée automatiquement

1. **Plateau**

Les éléments qui composent cette classe :

* Une liste de cases
* Un stub (permettant de d’instancier la liste de cases)

Une image contenant texte

Description générée automatiquement

Les fonctions qui composent cette classe :

* Une image contenant texte

  Description générée automatiquement***bool*** Consigne (int x, int y, string directions) => applique une consigne avec la position initiale de l’aventurier et les directions qui lui sont attribuées, retourne ***false*** si la position initiale est incorrect ou si le déplacement de l’aventurier est incorrect.
* ***bool*** Deplacement (string directions) => déplacement de l’aventurier, retourne ***false*** si le déplacement de l’aventurier est impossible.

Une image contenant texte

Description générée automatiquement

Une image contenant texte

Description générée automatiquement

La méthode qui compose cette classe :

* Une image contenant texte

  Description générée automatiquementAffichePlateau () => permet d’afficher le plateau dans l’invite de commande

1. **Stub**

La fonction qui compose cette classe :

* ***List<Case>*** Consigne (int x, int y, string directions) => lit le fichier carte.txt, retourne ***une liste de cases*** qui correspondent aux cases du fichier

Une image contenant texte

Description générée automatiquement

**Application console :**

1. Une image contenant texte

   Description générée automatiquement**Affichage plateau vierge**
2. Une image contenant texte

   Description générée automatiquement**1ère consigne**

Informations données :

3,0

SSSSEEEEEENN

Résultat attendu :

Le personnage doit se trouver en (9,2)

1. Une image contenant texte

   Description générée automatiquement**2nd consigne**

Informations données :

6,9

OONOOOSSO

Résultat attendu :

Le personnage doit se trouver en (7,5)

**Tests unitaires :**

Utilisation de packages nuggets :

* FluentAssertions => Permet de personnaliser la gestion des résultats de tests unitaires.
* AutoFixture => Facilite le développement piloté par les tests et rend les tests unitaires plus sûrs.

1. **Test en fonction de la position initiale**

Une image contenant texte

Description générée automatiquement

1. Une image contenant texte

   Description générée automatiquement**Test en fonction des directions**