Level

Cas de test : `Level::initOK`

- **Couverture** : préconditions
- **Objectif** : précondition satisfaite
- **Conditions initiales** : vide
- **Opérations** : C0 := init(10, 10)
- **Oracle** :
  - **postconditions** :
    - width(C0) = 10
    - heigth(C0) = 10
    - isEditting = true
  - **invariants** :
    - height(C0) > 5
    - width() > 4
- **Rapport** :
  - *width(C0) ≠ 10*
    - "Le width initiale devrait être 10."
  - *heigth(C0) ≠ 10*
    - "Le heigth initiale n'est pas 10."

Cas de test : `Level::initKO`

- **Couverture** : préconditions
- **Objectif** : précondition non-satisfaite
- **Conditions initiales** : vide
- **Opérations** : C0 := init(-2 , -10)
- **Oracle** :
  - Une exception `PreconditionError` doit être signalée
- **Rapport** :
  - pas d'exception
    - "Une exception `PreconditionError` doit être signalée."

Cas de test : `SUT::préparation de terrain`

- **Couverture** : scénario utilisateur
- **Objectif** : initialisation level et choix de l'entrée de la sortie
- **Conditions initiales** :
  - T0 := Level::init(10, 10)
- **Opérations** :
  - T1 :=

```
for (int i = 0; i < 10; i++) {
        Level::setNature(T0, 0, i, Nature.METAL);
        Level::setNature(T0, 9, i, Nature.METAL);
}
for (int i = 0; i < 10 ; i++) {
        Level::setNature(T0, i, 0, Nature.METAL);
        Level::setNature(T0, i, 9, Nature.METAL);
}
```

  - T2 := Level::goPlay(T0, 2, 2, 8, 8);

- **Oracle** :

```
for (int i = 0; i < 10; i++) {
        Level::nature(T1, 0, i) = Nature.METAL;
```

Level::nature(T1, 9,) = Nature.METAL
}
Level::exitX(T2) = 8
Level::exitY(T2) = 8
Level::entranceX(T2) = 2
Level::entranceY(T2) = 2

- **Rapport** :
  - *Level::exitX(T2) ≠ 8 || Level::exitY(T2) ≠ 8*
    - "L'entrée n'a pas sa position"
  - *Level::entranceX(T2) ≠ 2 || Level::entranceY(T2) ≠ 2*
    - "La quantité transvasée dans la cuve de sortie n'est pas correcte."