## Algorithmique Appliquée

**BTS SIO SISR** 

Tests, exceptions, assertions





### Plan

- Gestion d'erreurs avec des codes de retour
- Notion d'exception
- Gestion d'exceptions et classes d'exception
- Programmation offensive et défensive
- Assertions
- Invariants
- Tests et qualité logicielle
- Tests en boîte transparente par les développeurs
- Automatisation des tests
- Tests unitaires
- Tests pilotant le développement
- Pyramide de tests

#### Gestion d'erreurs avec des codes de retour





### Sources d'erreurs (1/2)

- Première source d'erreur : entrées utilisateur.
  - Utilisateur distrait.
  - Utilisateur malveillant.

### Sources d'erreurs (2/2)

- Autres sources d'erreur :
  - Limites de l'ordinateur :
    - int  $\neq \mathbb{Z}$ .
    - float  $\neq \mathbb{R}$ .
    - Mémoire de la machine limitée.
    - Espace disque limité.
  - o Pannes réseau.
  - Bug dans une bibliothèque utilisée.

### Problématique

- Comment gérer efficacement les erreurs sans impacter les performances ni la clarté du code ?
- Quand doit-on gérer les erreurs ?

### Notion d'exception





#### Gestion d'exceptions et classes d'exception





### Programmation offensive et défensive





### Assertions





### Invariants

Préconditions et post-conditions





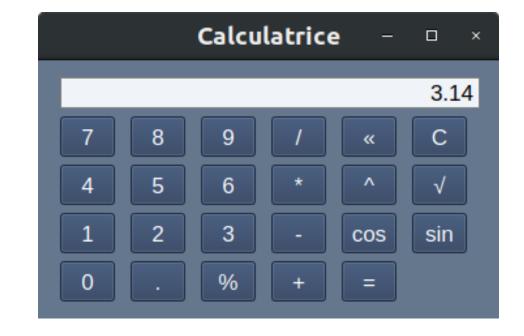
# TP: Exceptions dans une calculatrice





# TP: Exceptions dans une calculatrice

Lien vers le sujet de TP.



## Tests et qualité logicielle

Boîte opaque et équipe QA





### Tests en boîte transparente

Par les développeurs





### Automatisation des tests





### Tests unitaires





### Tests pilotant le développement

**Test Driven Development** 





# Pyramide de tests





# TP: Ecriture de tests unitaires





#### TP: Ecriture de tests unitaires

Lien vers le sujet de TP.

### Devoir à la Maison 04





# DM : Retour sur la complexité et les tests

Lien vers le sujet de DM.