# TP Tests Unitaires en C et Python

**Numéro de Référence**
**#UNIVPM001**

(Document de 7 pages)


**Résumé**

Rapport sur les Travaux Pratiques de l'unité d'enseignement Fiabilité Logicielle du Master 1 Informatique portant sur les tests unitaires sur les thèmes de vérification de type triangle et de recherche d'éléments dans un tableau. Ce TP fut réalisé en langages C et Python.



**Mots Clés**

Fiabilité Logicielle Test Unitaire CUnit Unittest



Université d'Aix-Marseille
Département Informatique et Interactions

3 Place Victor Hugo
13331 MARSEILLE CEDEX 3

## SECTION DES REDACTEURS

| Nom | Prénom | Contribution |
|---|---|---|
| Gonzalez | Sébastien | readData, typeTriangle, chercherElt, tests, rapport |
| Rouabhia | Younes | readData, typeTriangle, chercherElt, tests, rapport |
| | | |
| | | |

## CONTACTS

| Nom | Prénom | Email | Fonction |
|---|---|---|---|
| Gonzalez | Sébastien | Sebastien.gonzalez@etu.univ-amu.fr | Implémentation, rédaction |
| Rouabhia | Younes | Younes.rouabhia@etu.univ-amu.fr | Implémentation, rédaction |
| | | | |
| | | | |

## HISTORIQUE DES MODIFICATIONS

| Modifications | Date | Version | Approbateur de la diffusion |
|---|---|---|---|
| | | | |
| | | | |
| | | | |
| | | | |

# TABLE DES MATIERES

# 1. INTRODUCTION

Vous trouverez dans ce rapport le détail de la réalisation du premier TP de Fiabilité Logicielle du Semestre 2 du Master 1 Informatique de l'Université d'Aix-Marseille.

Pour la première partie, nous commençons d'abord par implémenter, en langage C, une application Triangle qui réalisera la lecture d'un fichier contenant les informations nécessaires à propos des longueurs des côtés d'un triangle afin de déterminer le type de ce triangle (Equilatéral, isocèle, scalène ou inexistant).

Ensuite, nous écrivons une série de tests unitaires afin de tester la lecture du fichier ainsi que les affectations des types de triangle.

Pour la seconde partie, nous mettons en place une application Python devant chercher puis récupérer un élément de type dans un tableau d'entiers triés par ordre croissant.

Enfin, une seconde suite de test écrite avec Unittest cette fois, est ajoutée afin de vérifier la fonction de recherche.

# 2. LE TRIANGLE

## 2.1. readData

La première fonction à écrire était `readData` qui prend en entrée une chaîne de caractère `nomDeFichier` et renvoie un tableau de 3 réels qui correspondent aux 3 longueurs des côtés d'un triangle.

Le fichier passé en paramètre doit avoir 3 lignes, chacune contenant un nombre réel.

Exemple :

data.txt

3

4

5

`readData(''data.txt'')` renverra `[3, 4, 5]`

### 2.1.1. Tests unitaires

`testFichierValide` : Compare les valeurs d'un fichier passé en paramètre et celles d'un fichier contenant les mêmes valeurs valides codé à la main.

`testFichierVide` : Renvoie -1 si le fichier est vide.

`testFichierMauvaisNombreDeLignes` : Vérifie que le nombre de lignes est bien égal à 3.

`testFichierMauvaisFormatDeLigne` : Rejette les fichiers mal formattés ou contenant autre chose que des réels.

`testFichierInexistant` : Teste si le fichier existe.

`testFichierBinaire` : Tester d'ouvrir un fichier binaire (renvoit une erreur).

## 2.2. Triangle

La fonction `typeTriangle` prend en entrée trois arguments de type double afin de coder les côtés du triangle de longueurs réelles.

Elle renvoie 3 si le triangle est équilatéral, 2 s'il est isocèle, 1 s'il est scalène et -1 si les données ne peuvent pas représenter pas un triangle.

Exemple : `typeTriangle(5.4, 3.15, 5.4)` renvoie 2.

### 2.2.1. Tests unitaires

`testTypeTriangleEquilateral` : Renvoie 3 si le triangle est équilatéral.

`testTypeTriangleIsocèle` : Renvoie 2 si le triangle est isocèle.

`testTypeTriangleScalène` : Renvoie 1 si le triangle est scalène.

`testTriangleIncorrect` : Teste si les données entrées représente un triangle valide en se basant sur l'inégalité triangulaire ; chaque côté doit être de longueur inférieure à la somme des deux autres. Renvoie -1 si le triangle n'est pas valide.

`testTriangleValeurNulle` : Renvoie -1 dans le cas où les données sont nulles.

`testTriangleValeurNegative` : Renvoie -1 si une des longueurs est négative.

# 3. CHERCHER UN ELEMENT DANS UN TABLEAU TRIE

## 3.1. chercherElt

Cette fonction prend en arguments un entier `elt` et un `Tableau tab` trié par ordre croissant, elle renvoie -1 si `elt` n'est pas un élément du tableau, sinon renvoie le premier indice `i` du tableau tel que `tab[i] = elt`.

### 3.1.1. Tests unitaires

`test_value_connue` : Teste si la fonction renvoie l'indice de `elt` si la valeur `elt` passée existe pas dans le tableau.

`test_value_inconnue` : Teste si la fonction renvoie -1 si la valeur `elt` passée n'existe pas dans le tableau.

`test_value_indice_0` : Teste si la fonction renvoie bien 0 si `elt` est le premier élement du tableau.

`test_value_indice_dernier` : Teste si la fonction renvoie bien le dernier indice du tableau si `elt` est le dernier élement de ce dernier.

`test_tableau_vide` : Teste que la fonction renvoie toujours -1 dans le cas où le tableau est vide.

`Test_tableau_nul` : Teste que la fonction renvoie toujours -1 dans le cas où le tableau est nul.