TP : Sureté de fonctionnement/Vérification de Programme

Partie I : Familiarisation avec le langage de test unitaire RTRT/TENIS

Exercice 1 :

SPECIFICATION
Exigence_00030

La valeur initiale de la variable compteur est 0. Si la valeur maximum 254 est atteinte alors le compteur doit être remis à 0.
Le compteur est de type char non signé codé sur 8 bits prenant de valeurs paires.
Le compteur doit être incrementé de 2 à chaque appel de la function counter.

1. **Combien de cas de test ?**
2. **Ecrire le test en langage RTRT et l'exécuter ?**

Exercice 2

Soit la spécification suivante :

La fonction maximum doit renvoyer le maximum des deux nombres entiers en paramètre.

La fonction valeur_milieu doit renvoyer la valeur milieu des trois entiers en paramètre.

1. **Ecrire les cas de tests pour tester la fonction maximum ?**
2. **Modifier le ptu dans Local/unit/test/stat.ptu. Exécuter le test avec tenis.**
3. **Que constatez-vous ?**
4. **Ecrire les cas de tests pour tester la fonction valeur_milieu?**
5. **Exécuter le test.**

Partie II Test boîte noire / Pratique classe d'équivalence

Exercice 1

Soit la spécification suivante :
Le service prend en paramètre un tableau de 10 entiers.
Le service renvoie en paramètre en sortie la moyenne de ces 10 entiers.
Le service renvoie en paramètre en sortie la somme de ces 10 entiers.
Voici son prototype `void Calc_Sum_Mean(int Array[], int *Sum, int *Mean) ;`

- Produire une suite de cas de tests pour ce programme

Exercice 2

- Ecrire les tests fonctionnels en RTRT
- Réaliser et exécuter les tests avec TENIS en mode sans couverture
- Et avec couverture

Exercice 3

Soit la spécification suivante :

Un programme prend en entrée trois octets (0-255) . Ces trois octets sont interprétés comme représentant les longueurs des côtés d'un triangle. Le programme rend un résultat précisant s'il s'agit d'un triangle scalène, isocèle ou équilatéral et renvoie un message approprié sinon. La fonction s'appelle triangle_al_kashi prenant trois paramètres a,b,c.

1. **Produire une suite de cas de tests pour ce programme ?**
2. **Exécuter le test avec TENIS.**
3. **Exécuter le même test en appelant la fonction traingle_petit_cote qui fait la même chose. Quelle différence constatez-vous ?**

Partie IV : Test boîte blanche : Couverture structurelle

Exercice 1 :

SPECIFICATION

Requirement_00050

Si l'avion est en phase décollage (v > 1) et qu'il n'y pas de brouillard (b =0), le kérosène (k) est égal à sa valeur divisée par la phase de vol v.
Si la phase de vol est en croisière (v =2) ou qu'il y a encore de kérosène (k > 1) alors le kérosène doit être incrémenté. Pendant la phase croisière, la variable entière non signée LG (lumière générale) doit être figée à zéro. Sa valeur par défaut est 1.

1. **Combien de cas de test en appliquant la méthode de classe d'équivalence ?**
2. **Ecrire le test en langage RTRT et l'exécuter ?**
3. **Couverture instruction : écrire un cas de test qui permet de couvrir toutes les instructions ? Exécuter sur TENIS et la couverture doit être à 100 %**
4. **Couverture de décision : Remarquez que le code est faux pour autant, la première ligne doit être && non ||. Ce qui veut dire que couvrir toutes les instructions ne suffisent pas. Il faut prendre en compte les décisions. Essayez d'exécuter les deux tests suivants : v= 3, b = 0, k = 3 et v = 2, b = 1, k = 1. Ce test est censé couvrir toutes les décisions. Est-ce que ce test peut détecter l'erreur sur le code si k<1 ?**
5. **Couverture de condition : Il faut donc couvrir les conditions : trouver les cas de tests pour que toutes les valeurs des variables de conditions sont testées ? Il y a quand même un chemin qui n'est parcouru.**
6. **MCDC : pour que notre test soit complet, il faut combiner couverture instruction/ Décision/ Condition et en plus à chaque changement de condition. C'est-à-dire toutes les variables de conditions doivent participer à la décision.**
7. **Quel type d'erreur ne peut pas être détecté par la couverture structurelle aussi complète soit-elle ?**