Informatique – TPentraînement

Vésale Nicolas - Henrik Thys

Pour chacune des fonctions suivantes, donner:

1. un variant de boucle,
2. un invariant de boucle,
3. en déduire la correction totale de la fonction.

1. **Puissance:**

```
def puissance(x, n):
    result = 1
    i = 0
    while i < n:
        result = result * x
        i = i+1
    return result
```

2. **Somme des n premiers entiers:**

```
def somme_n_entiers(n):
    total = 0
    i = 1
    while i <= n:
        total = total + i
        i = i + 1
    return total
```

3. $\log_2$ **d'un entier strictement positif :**

```
def log_base_2(n):
    count = 0
    while n > 1:
        n = n // 2
        count = count + 1
    return count
```

4. **Nombre de chiffres:**

```
def nombre_de_chiffres(n):
    compteur = 0
    while n > 0:
        n =n// 10
        compteur = compteur + 1
    return compteur
```

5. **Minimum d'une liste:**

```
def minimum(liste):
    min_val = liste[0]
    i = 1
    while i < len(liste):
        if liste[i] < min_val:
            min_val = liste[i]
        i = i + 1
    return min_val
```

6. **Nombre d'occurrences d'un élément dans une liste :**

```
def compter_occurrences(liste, element):
    count = 0
    i = 0
    while i < len(liste):
        if liste[i] == element:
            count = count + 1
        i = i + 1
    return count
```

7. **Vérifier si une liste est triée :**

```
def est_triee(liste):
    i = 1
    while i < len(liste):
        if liste[i] < liste[i - 1]:
            return False
        i = i + 1
    return True
```

8. **Vérifier si une chaîne est un palindrome:**

```
def est_palindrome(chaine):
    i = 0
    j = len(chaine) - 1
    while i < j:
        if chaine[i] != chaine[j]:
            return False
        i = i + 1
        j = j - 1
    return True
```

9. **Inversion d'une liste:**

```
def inversion(liste):
    i = 0
    j = len(liste) - 1
    while i < j:
        liste[i],liste[j]=liste[j],liste[i]
        i = i + 1
        j = j - 1
    return liste
```