Exercices Corrigés – Algorithmique et Programmation

E LEARNING BTS

Septembre 2025

1 Les Boucles

Exercice 1 – Somme des nombres entre deux bornes

Énoncé : Écrire un algorithme, un programme en langage C, et un script en PHP qui permet de saisir deux bornes et réaliser la somme de tous les nombres compris entre les bornes.

Tips : Utilisez une boucle (`pour`) pour parcourir tous les nombres entre les deux bornes. Pensez à gérer le cas où la première borne est supérieure à la seconde.

Solution Algorithmique

```
Algorithme SommeEntrebornes
Variables
    borne1, borne2, i, somme : entier
Début
    afficher("Entrez la première borne : ")
    saisir(borne1)
    afficher("Entrez la deuxième borne : ")
    saisir(borne2)

    somme <- 0

    si (borne1 > borne2) alors
        pour i allant de borne2 à borne1 faire
            somme <- somme + i
        fin pour
    sinon
        pour i allant de borne1 à borne2 faire
            somme <- somme + i
        fin pour
    fin si

    afficher("La somme des nombres entre ", borne1,
             " et ", borne2, " est : ", somme)
Fin
```

Solution en Langage C

```c
#include <stdio.h>

int main() {
    int borne1, borne2, i, somme = 0;

    printf("Entrez la première borne : ");
    scanf("%d", &borne1);
    printf("Entrez la deuxième borne : ");
    scanf("%d", &borne2);

    if (borne1 > borne2) {
        for (i = borne2; i <= borne1; i++) {
            somme += i;
        }
    } else {
        for (i = borne1; i <= borne2; i++) {
            somme += i;
        }
    }

    printf("La somme des nombres entre %d et %d est : %d\n",
           borne1, borne2, somme);

    return 0;
}
```

Solution en PHP

```php
<?php
echo "Entrez la première borne : ";
$borne1 = intval(fgets(STDIN));
echo "Entrez la deuxième borne : ";
$borne2 = intval(fgets(STDIN));

$somme = 0;

if ($borne1 > $borne2) {
    for ($i = $borne2; $i <= $borne1; $i++) {
        $somme += $i;
    }
} else {
    for ($i = $borne1; $i <= $borne2; $i++) {
        $somme += $i;
    }
}

echo "La somme des nombres entre $borne1 et $borne2 est : $somme\n";
?>
```

2 Les Tableaux

Exercice 1 – Prix Maximum

Énoncé : Écrire un algorithme, un programme en langage C, et un script en PHP qui permet de stocker dans un tableau dix prix et affiche le prix max, le prix min et le prix moyen.

Tips : Initialisez le prix max à une petite valeur (ou au premier élément) et le prix min à une grande valeur. Parcourez le tableau pour trouver les valeurs extrêmes et calculer la somme pour la moyenne.

Solution Algorithmique

```
Algorithme PrixMaxMinMoyen
Variables
    prix : tableau[10] de réel
    i : entier
    prixMax, prixMin, somme, moyenne : réel
Début
    // Saisie des prix
    pour i allant de 0 à 9 faire
        afficher("Entrez le prix ", i+1, " : ")
        saisir(prix[i])
    fin pour

    // Initialisation
    prixMax <- prix[0]
    prixMin <- prix[0]
    somme <- 0

    // Calcul
    pour i allant de 0 à 9 faire
        si (prix[i] > prixMax) alors
            prixMax <- prix[i]
        fin si

        si (prix[i] < prixMin) alors
            prixMin <- prix[i]
        fin si

        somme <- somme + prix[i]
    fin pour

    moyenne <- somme / 10

    afficher("Prix maximum : ", prixMax)
    afficher("Prix minimum : ", prixMin)
    afficher("Prix moyen : ", moyenne)
Fin
```

Solution en Langage C

```c
#include <stdio.h>

int main() {
    float prix[10];
    int i;
    float prixMax, prixMin, somme = 0, moyenne;

    // Saisie des prix
    for (i = 0; i < 10; i++) {
        printf("Entrez le prix %d : ", i+1);
        scanf("%f", &prix[i]);
    }

    // Initialisation
    prixMax = prix[0];
    prixMin = prix[0];

    // Calcul
    for (i = 0; i < 10; i++) {
        if (prix[i] > prixMax) {
            prixMax = prix[i];
        }

        if (prix[i] < prixMin) {
            prixMin = prix[i];
        }

        somme += prix[i];
    }

    moyenne = somme / 10;

    printf("\nPrix maximum : %.2f\n", prixMax);
    printf("Prix minimum : %.2f\n", prixMin);
    printf("Prix moyen : %.2f\n", moyenne);

    return 0;
}
```

Solution en PHP

```php
<?php
$prix = array();

// Saisie des prix
for ($i = 0; $i < 10; $i++) {
    echo "Entrez le prix " . ($i+1) . " : ";
    $prix[$i] = floatval(fgets(STDIN));
}
```

```php
// Initialisation
$prixMax = $prix[0];
$prixMin = $prix[0];
$somme = 0;

// Calcul
for ($i = 0; $i < 10; $i++) {
    if ($prix[$i] > $prixMax) {
        $prixMax = $prix[$i];
    }

    if ($prix[$i] < $prixMin) {
        $prixMin = $prix[$i];
    }

    $somme += $prix[$i];
}

$moyenne = $somme / 10;

echo "\nPrix maximum : " . number_format($prixMax, 2) . "\n";
echo "Prix minimum : " . number_format($prixMin, 2) . "\n";
echo "Prix moyen : " . number_format($moyenne, 2) . "\n";
?>
```

Exercice 2 – Rotation d'un Tableau

Énoncé : Écrire un algorithme, un programme en langage C, et un script en PHP qui permet de stocker dans un tableau dix entiers et réalise la rotation des éléments à gauche ou à droite un nombre de fois fixé par l'utilisateur.

Tips :

- Pour une rotation à gauche : sauvegarder le premier élément, décaler tous les éléments vers la gauche, placer l'élément sauvegardé à la fin.
- Pour une rotation à droite : sauvegarder le dernier élément, décaler tous les éléments vers la droite, placer l'élément sauvegardé au début.
- Répéter l'opération selon le nombre de rotations demandé.

Solution Algorithmique

```
Algorithme RotationTableau
Variables
    tab : tableau[10] d'entier
    i, j, choix, nbRotations, temp : entier
Début
    // Saisie du tableau
    pour i allant de 0 à 9 faire
        afficher("Entrez l'élément ", i+1, " : ")
        saisir(tab[i])
    fin pour

    afficher("Choisissez : 1-Gauche, 2-Droite : ")
    saisir(choix)
    afficher("Nombre de rotations : ")
    saisir(nbRotations)

    pour j allant de 1 à nbRotations faire
        si (choix = 1) alors
            // Rotation à gauche
            temp <- tab[0]
            pour i allant de 0 à 8 faire
                tab[i] <- tab[i+1]
            fin pour
            tab[9] <- temp
        sinon
            // Rotation à droite
            temp <- tab[9]
            pour i allant de 9 à 1 (décroissant) faire
                tab[i] <- tab[i-1]
            fin pour
            tab[0] <- temp
        fin si
    fin pour
```

```
    afficher("Tableau après rotation : ")
    pour i allant de 0 à 9 faire
        afficher(tab[i], " ")
    fin pour
Fin
```

Solution en Langage C

```c
#include <stdio.h>

int main() {
    int tab[10];
    int i, j, choix, nbRotations, temp;

    // Saisie du tableau
    for (i = 0; i < 10; i++) {
        printf("Entrez l'élément %d : ", i+1);
        scanf("%d", &tab[i]);
    }

    printf("Choisissez : 1-Gauche, 2-Droite : ");
    scanf("%d", &choix);
    printf("Nombre de rotations : ");
    scanf("%d", &nbRotations);

    for (j = 0; j < nbRotations; j++) {
        if (choix == 1) {
            // Rotation à gauche
            temp = tab[0];
            for (i = 0; i < 9; i++) {
                tab[i] = tab[i+1];
            }
            tab[9] = temp;
        } else {
            // Rotation à droite
            temp = tab[9];
            for (i = 9; i > 0; i--) {
                tab[i] = tab[i-1];
            }
            tab[0] = temp;
        }
    }

    printf("\nTableau après rotation : ");
    for (i = 0; i < 10; i++) {
        printf("%d ", tab[i]);
    }
    printf("\n");

    return 0;
```

```
}
```

Solution en PHP

```php
<?php
$tab = array();

// Saisie du tableau
for ($i = 0; $i < 10; $i++) {
    echo "Entrez l'élément " . ($i+1) . " : ";
    $tab[$i] = intval(fgets(STDIN));
}

echo "Choisissez : 1-Gauche, 2-Droite : ";
$choix = intval(fgets(STDIN));
echo "Nombre de rotations : ";
$nbRotations = intval(fgets(STDIN));

for ($j = 0; $j < $nbRotations; $j++) {
    if ($choix == 1) {
        // Rotation à gauche
        $temp = $tab[0];
        for ($i = 0; $i < 9; $i++) {
            $tab[$i] = $tab[$i+1];
        }
        $tab[9] = $temp;
    } else {
        // Rotation à droite
        $temp = $tab[9];
        for ($i = 9; $i > 0; $i--) {
            $tab[$i] = $tab[$i-1];
        }
        $tab[0] = $temp;
    }
}

echo "\nTableau après rotation : ";
for ($i = 0; $i < 10; $i++) {
    echo $tab[$i] . " ";
}
echo "\n";
?>
```

3 Procédures et Fonctions

Exercice 1 – Procédure des Diviseurs

Énoncé : Écrire une procédure (algo, C, PHP) qui permet de recevoir un entier et d'afficher ses diviseurs. Faites appel à cette procédure.

Tips : Un nombre `d` est un diviseur de `n` si `n % d == 0`. Parcourez tous les nombres de 1 à `n`.

Solution Algorithmique

```
Procédure afficherDiviseurs(n : entier)
Variables
    i : entier
Début
    afficher("Les diviseurs de ", n, " sont : ")
    pour i allant de 1 à n faire
        si (n % i = 0) alors
            afficher(i, " ")
        fin si
    fin pour
    afficher("\n")
Fin afficherDiviseurs

// Programme principal
Algorithme Principal
Variables
    nombre : entier
Début
    afficher("Entrez un nombre entier : ")
    saisir(nombre)
    afficherDiviseurs(nombre)
Fin
```

Solution en Langage C

```c
#include <stdio.h>

void afficherDiviseurs(int n) {
    int i;
    printf("Les diviseurs de %d sont : ", n);
    for (i = 1; i <= n; i++) {
        if (n % i == 0) {
            printf("%d ", i);
        }
    }
    printf("\n");
}
```

```c
int main() {
    int nombre;

    printf("Entrez un nombre entier : ");
    scanf("%d", &nombre);

    afficherDiviseurs(nombre);

    return 0;
}
```

Solution en PHP

```php
<?php
function afficherDiviseurs($n) {
    echo "Les diviseurs de $n sont : ";
    for ($i = 1; $i <= $n; $i++) {
        if ($n % $i == 0) {
            echo $i . " ";
        }
    }
    echo "\n";
}

// Programme principal
echo "Entrez un nombre entier : ";
$nombre = intval(fgets(STDIN));

afficherDiviseurs($nombre);
?>
```

Exercice 2 – Fonction Moyenne d'un Tableau

Énoncé : Écrire une fonction (algo, C, PHP) qui permet de recevoir un tableau de dix entiers et renvoie en sortie la moyenne. Faites appel à cette fonction.

Tips : La moyenne est la somme de tous les éléments divisée par le nombre d'éléments (ici 10).

Solution Algorithmique

```
Fonction calculerMoyenne(tab : tableau[10] d'entier) : réel
Variables
    i : entier
    somme : réel
Début
    somme <- 0
    pour i allant de 0 à 9 faire
        somme <- somme + tab[i]
    fin pour
    retour somme / 10
Fin calculerMoyenne

// Programme principal
Algorithme Principal
Variables
    tableau : tableau[10] d'entier
    i : entier
    moyenne : réel
Début
    pour i allant de 0 à 9 faire
        afficher("Entrez l'élément ", i+1, " : ")
        saisir(tableau[i])
    fin pour

    moyenne <- calculerMoyenne(tableau)
    afficher("La moyenne est : ", moyenne)
Fin
```

Solution en Langage C

```c
#include <stdio.h>

float calculerMoyenne(int tab[]) {
    int i;
    float somme = 0;

    for (i = 0; i < 10; i++) {
        somme += tab[i];
    }
```

```
    return somme / 10;
}

int main() {
    int tableau[10];
    int i;
    float moyenne;

    for (i = 0; i < 10; i++) {
        printf("Entrez l'élément %d : ", i+1);
        scanf("%d", &tableau[i]);
    }

    moyenne = calculerMoyenne(tableau);
    printf("La moyenne est : %.2f\n", moyenne);

    return 0;
}
```

Solution en PHP

```
<?php
function calculerMoyenne($tab) {
    $somme = 0;

    for ($i = 0; $i < 10; $i++) {
        $somme += $tab[$i];
    }

    return $somme / 10;
}

// Programme principal
$tableau = array();

for ($i = 0; $i < 10; $i++) {
    echo "Entrez l'élément " . ($i+1) . " : ";
    $tableau[$i] = intval(fgets(STDIN));
}

$moyenne = calculerMoyenne($tableau);
echo "La moyenne est : " . number_format($moyenne, 2) . "\n";
?>
```

4 Fonctions Récursives

Exercice 1 – Suite de Fibonacci

Énoncé : Écrire la fonction récursive qui permet de calculer le nombre de Fibonacci :

- f(0) = 1
- f(1) = 1
- f(n) = f(n-1) + f(n-2)

Tips : Une fonction récursive s'appelle elle-même. N'oubliez pas les cas de base (conditions d'arrêt) pour éviter une récursion infinie.

Solution Algorithmique

```
Fonction fibonacci(n : entier) : entier
Début
    si (n = 0 ou n = 1) alors
        retour 1
    sinon
        retour fibonacci(n-1) + fibonacci(n-2)
    fin si
Fin fibonacci

// Programme principal
Algorithme Principal
Variables
    n, resultat : entier
Début
    afficher("Entrez n : ")
    saisir(n)
    resultat <- fibonacci(n)
    afficher("F(", n, ") = ", resultat)
Fin
```

Solution en Langage C

```
#include <stdio.h>

int fibonacci(int n) {
    if (n == 0 || n == 1) {
        return 1;
    } else {
        return fibonacci(n-1) + fibonacci(n-2);
    }
}

int main() {
```

```
    int n, resultat;

    printf("Entrez n : ");
    scanf("%d", &n);

    resultat = fibonacci(n);
    printf("F(%d) = %d\n", n, resultat);

    return 0;
}
```

Solution en PHP

```
<?php
function fibonacci($n) {
    if ($n == 0 || $n == 1) {
        return 1;
    } else {
        return fibonacci($n-1) + fibonacci($n-2);
    }
}

// Programme principal
echo "Entrez n : ";
$n = intval(fgets(STDIN));

$resultat = fibonacci($n);
echo "F($n) = $resultat\n";
?>
```

Exercice 2 – Puissance par Récursivité

Énoncé : Écrire la fonction récursive qui permet de calculer la puissance de a^b par multiplications successives.
Formule : $a^b = a^{b-1} \times a$

Tips : Le cas de base est $a^0 = 1$. Pour $b > 0$, on multiplie a par a^{b-1}.

Solution Algorithmique

```
Fonction puissance(a : réel, b : entier) : réel
Début
    si (b = 0) alors
        retour 1
    sinon
        retour a * puissance(a, b-1)
    fin si
Fin puissance

// Programme principal
Algorithme Principal
Variables
    a, resultat : réel
    b : entier
Début
    afficher("Entrez la base a : ")
    saisir(a)
    afficher("Entrez l'exposant b : ")
    saisir(b)

    resultat <- puissance(a, b)
    afficher(a, " puissance ", b, " = ", resultat)
Fin
```

Solution en Langage C

```
#include <stdio.h>

float puissance(float a, int b) {
    if (b == 0) {
        return 1;
    } else {
        return a * puissance(a, b-1);
    }
}

int main() {
    float a, resultat;
    int b;
```

```
    printf("Entrez la base a : ");
    scanf("%f", &a);
    printf("Entrez l'exposant b : ");
    scanf("%d", &b);

    resultat = puissance(a, b);
    printf("%.2f puissance %d = %.2f\n", a, b, resultat);

    return 0;
}
```

Solution en PHP

```php
<?php
function puissance($a, $b) {
    if ($b == 0) {
        return 1;
    } else {
        return $a * puissance($a, $b-1);
    }
}

// Programme principal
echo "Entrez la base a : ";
$a = floatval(fgets(STDIN));
echo "Entrez l'exposant b : ";
$b = intval(fgets(STDIN));

$resultat = puissance($a, $b);
echo "$a puissance $b = " . number_format($resultat, 2) . "\n";
?>
```

Note Importante sur les Fonctions Récursives

Avantages :

- Code élégant et concis
- Résout naturellement certains problèmes (arbres, suites mathématiques)

Inconvénients :

- Peut être moins performant (appels multiples)
- Risque de dépassement de pile pour
- Risque de dépassement de pile pour de grandes valeurs
- Consomme plus de mémoire que les versions itératives

Optimisation : Pour la fonction Fibonacci, privilégiez une version itérative ou utilisez la mémoïsation pour les grandes valeurs de n.

5 Exercices Supplémentaires – Les Fichiers

Exercice Bonus 1 – Sauvegarde de Prix dans un Fichier

Énoncé : Écrire un algorithme, un programme en langage C, et un script en PHP qui permet de saisir 5 prix, de les sauvegarder dans un fichier texte nommé "prix.txt", puis de les relire et afficher.

Tips :

- Ouvrez le fichier en mode écriture ("w") pour sauvegarder
- Fermez le fichier après écriture
- Réouvrez en mode lecture ("r") pour afficher
- Vérifiez toujours si le fichier s'est bien ouvert

Solution Algorithmique

```
Algorithme SauvegarderPrix
Variables
    fichier : fichier
    prix : réel
    i : entier
Début
    // Écriture dans le fichier
    fichier <- ouvrir("prix.txt", écriture)

    pour i allant de 1 à 5 faire
        afficher("Entrez le prix ", i, " : ")
        saisir(prix)
        ecrire(fichier, prix)
    fin pour

    fermer(fichier)

    afficher("\n--- Lecture du fichier ---\n")

    // Lecture du fichier
    fichier <- ouvrir("prix.txt", lecture)

    i <- 1
    tant que non fin_fichier(fichier) faire
        lire(fichier, prix)
        afficher("Prix ", i, " : ", prix)
        i <- i + 1
    fin tant que

    fermer(fichier)
Fin
```

Solution en Langage C

```c
#include <stdio.h>

int main() {
    FILE *fichier;
    float prix;
    int i;

    // Écriture dans le fichier
    fichier = fopen("prix.txt", "w");

    if (fichier == NULL) {
        printf("Erreur d'ouverture du fichier en écriture\n");
        return 1;
    }

    for (i = 1; i <= 5; i++) {
        printf("Entrez le prix %d : ", i);
        scanf("%f", &prix);
        fprintf(fichier, "%.2f\n", prix);
    }

    fclose(fichier);

    printf("\n--- Lecture du fichier ---\n");

    // Lecture du fichier
    fichier = fopen("prix.txt", "r");

    if (fichier == NULL) {
        printf("Erreur d'ouverture du fichier en lecture\n");
        return 1;
    }

    i = 1;
    while (fscanf(fichier, "%f", &prix) != EOF) {
        printf("Prix %d : %.2f\n", i, prix);
        i++;
    }

    fclose(fichier);

    return 0;
}
```

Solution en PHP

```php
<?php
// Écriture dans le fichier
$fichier = fopen("prix.txt", "w");
```

```
if (!$fichier) {
    echo "Erreur d'ouverture du fichier en écriture\n";
    exit(1);
}

for ($i = 1; $i <= 5; $i++) {
    echo "Entrez le prix $i : ";
    $prix = floatval(fgets(STDIN));
    fprintf($fichier, "%.2f\n", $prix);
}

fclose($fichier);

echo "\n--- Lecture du fichier ---\n";

// Lecture du fichier
$fichier = fopen("prix.txt", "r");

if (!$fichier) {
    echo "Erreur d'ouverture du fichier en lecture\n";
    exit(1);
}

$i = 1;
while (!feof($fichier)) {
    $ligne = fgets($fichier);
    if ($ligne !== false && trim($ligne) !== "") {
        $prix = floatval($ligne);
        echo "Prix $i : " . number_format($prix, 2) . "\n";
        $i++;
    }
}

fclose($fichier);
?>
```

Exercice Bonus 2 – Comptage de Mots dans un Fichier

Énoncé : Écrire un algorithme, un programme en langage C, et un script en PHP qui permet de lire un fichier texte "texte.txt" et de compter le nombre de lignes, de mots et de caractères.

Tips :

- Lisez ligne par ligne avec fgets
- Comptez les lignes à chaque lecture
- Utilisez des espaces comme séparateurs pour compter les mots
- Comptez tous les caractères sauf les retours à la ligne

Solution Algorithmique

```
Algorithme ComptageFichier
Variables
    fichier : fichier
    ligne : chaîne
    nbLignes, nbMots, nbCaracteres, i : entier
    dansUnMot : booléen
Début
    nbLignes <- 0
    nbMots <- 0
    nbCaracteres <- 0

    fichier <- ouvrir("texte.txt", lecture)

    si (fichier = NULL) alors
        afficher("Erreur d'ouverture du fichier")
        arrêt
    fin si

    tant que non fin_fichier(fichier) faire
        lire_ligne(fichier, ligne)
        nbLignes <- nbLignes + 1

        // Compter les caractères
        nbCaracteres <- nbCaracteres + longueur(ligne)

        // Compter les mots
        dansUnMot <- faux
        pour i allant de 0 à longueur(ligne)-1 faire
            si (ligne[i] != ' ' et ligne[i] != '\t') alors
                si (dansUnMot = faux) alors
                    nbMots <- nbMots + 1
                    dansUnMot <- vrai
                fin si
            sinon
                dansUnMot <- faux
            fin si
```

```
        fin pour
    fin tant que

    fermer(fichier)

    afficher("Nombre de lignes : ", nbLignes)
    afficher("Nombre de mots : ", nbMots)
    afficher("Nombre de caractères : ", nbCaracteres)
Fin
```

Solution en Langage C

```c
#include <stdio.h>
#include <string.h>
#include <ctype.h>

int main() {
    FILE *fichier;
    char ligne[1000];
    int nbLignes = 0, nbMots = 0, nbCaracteres = 0;
    int i, dansUnMot;

    fichier = fopen("texte.txt", "r");

    if (fichier == NULL) {
        printf("Erreur d'ouverture du fichier\n");
        return 1;
    }

    while (fgets(ligne, sizeof(ligne), fichier) != NULL) {
        nbLignes++;

        // Compter les caractères (sans \n)
        int longueur = strlen(ligne);
        if (ligne[longueur-1] == '\n') {
            longueur--;
        }
        nbCaracteres += longueur;

        // Compter les mots
        dansUnMot = 0;
        for (i = 0; i < strlen(ligne); i++) {
            if (!isspace(ligne[i])) {
                if (!dansUnMot) {
                    nbMots++;
                    dansUnMot = 1;
                }
            } else {
                dansUnMot = 0;
            }
```

```
        }
    }

    fclose(fichier);

    printf("Nombre de lignes : %d\n", nbLignes);
    printf("Nombre de mots : %d\n", nbMots);
    printf("Nombre de caractères : %d\n", nbCaracteres);

    return 0;
}
```

Solution en PHP

```
<?php
$fichier = fopen("texte.txt", "r");

if (!$fichier) {
    echo "Erreur d'ouverture du fichier\n";
    exit(1);
}

$nbLignes = 0;
$nbMots = 0;
$nbCaracteres = 0;

while (!feof($fichier)) {
    $ligne = fgets($fichier);

    if ($ligne !== false) {
        $nbLignes++;

        // Compter les caractères (sans \n)
        $nbCaracteres += strlen(trim($ligne));

        // Compter les mots
        $mots = preg_split('/\s+/', trim($ligne), -1,
                           PREG_SPLIT_NO_EMPTY);
        $nbMots += count($mots);
    }
}

fclose($fichier);

echo "Nombre de lignes : $nbLignes\n";
echo "Nombre de mots : $nbMots\n";
echo "Nombre de caractères : $nbCaracteres\n";
?>
```

6 Conseils et Astuces Générales

Bonnes Pratiques de Programmation

1. **Nommage des variables :**
   - Utilisez des noms explicites : `prixTotal` plutôt que `pt`
   - Respectez les conventions : camelCase en C/PHP, snake_case acceptable
2. **Commentaires :**
   - Commentez le "pourquoi", pas le "comment"
   - Documentez les algorithmes complexes
3. **Gestion des erreurs :**
   - Vérifiez toujours les ouvertures de fichiers
   - Validez les entrées utilisateur
   - Gérez les cas limites (division par zéro, tableaux vides, etc.)
4. **Optimisation :**
   - Évitez les calculs répétitifs dans les boucles
   - Privilégiez les versions itératives aux récursives pour les grandes données
   - Utilisez des algorithmes efficaces (tri, recherche)

Débogage – Techniques Utiles

1. **Affichage de débogage :**
   - Affichez les valeurs des variables à des points clés
   - Tracez l'exécution des boucles (compteur d'itérations)
2. **Tests progressifs :**
   - Testez chaque fonction individuellement
   - Commencez par des cas simples avant les cas complexes
3. **Erreurs courantes :**
   - Indices de tableau : vérifiez les bornes (0 à n-1)
   - Boucles infinies : vérifiez les conditions d'arrêt
   - Comparaison vs affectation : == vs =
   - Oubli de fermer les fichiers

Complexité Algorithmique – Rappels

Notation Big O – Exemples :

- **O(1)** – Constant : Accès à un élément de tableau
- **O(n)** – Linéaire : Parcours d'un tableau
- **O(n^2)** – Quadratique : Tri à bulles, deux boucles imbriquées
- **O(log n)** – Logarithmique : Recherche dichotomique
- **O(n log n)** – Linéarithmique : Tri fusion, tri rapide

Exemple :

```
// O(n²) - Inefficace pour grandes données
for (i = 0; i < n; i++)
    for (j = 0; j < n; j++)
        // traitement

// O(n) - Plus efficace
for (i = 0; i < n; i++)
    // traitement
```

Fin du Document d'Exercices

Bon courage dans vos révisions & stay strong :)