

Université Mohammed V
Ecole Nationale Supérieure de l'Informatique et de l'Analyse des Systèmes
Département Génie Logiciel

Fondamentaux du langage C

Support de TD/TP

Langage C (M1.1.2)
Semestre 1 – 2026–2027

1

Tome 1

Fondamentaux : écrire et raisonner en C

Pr. Youness BOUTYOUR

youness.boutyour@um5.ac.ma

Table des matières

Objectifs	5
Consignes de travail	5
Partie 1 : Environnement, types et expressions	8
Exercice 1 – Premier exécutable	8
Exercice 2 – Types et tailles	8
Exercice 3 – Celsius et Fahrenheit	8
Exercice 4 – Rectangle	8
Exercice 5 – Division entière	8
Exercice 6 – Échange	8
Exercice 7 – Conversion de secondes	8
Exercice 8 – Moyenne pondérée	9
Exercice 9 – Codes de caractères	9
Exercice 10 – Dernier chiffre	9
Exercice 11 – Somme de durées	9
Exercice 12 – Puissance carrée et cube	9
Exercice 13 – Expressions équivalentes	9
Exercice 14 – Budget TTC	9
Exercice 15 – Formules de base	9
Exercice 16 – Conversions de bases	10
Exercice 17 – Échange par addition	10
Exercice 18 – Couleur RGB compacte	10
Exercice 19 – Prix en centimes	10
Exercice 20 – Moyenne de trois mesures	10
Partie 2 : Conditions, boucles et fonctions	13
Exercice 21 – Signe et parité	13
Exercice 22 – Maximum de trois	13
Exercice 23 – Année bissextile	13
Exercice 24 – Mention	13
Exercice 25 – Menu calculatrice	13
Exercice 26 – Somme de 1 à n	13
Exercice 27 – Table de multiplication	13

Exercice 28 – Factorielle	13
Exercice 29 – PGCD	14
Exercice 30 – Nombre premier	14
Exercice 31 – Fibonacci	14
Exercice 32 – Nombre mystère	14
Exercice 33 – Saisie contrôlée	14
Exercice 34 – Compter un chiffre	14
Exercice 35 – Triangle	14
Exercice 36 – Suite de Collatz	15
Exercice 37 – Décomposition en facteurs premiers	15
Exercice 38 – Racine par Newton	15
Exercice 39 – Palindrome numérique	15
Exercice 40 – Jour suivant	15
Exercice 41 – Lettre, casse et jour	15
Exercice 42 – Équation du second degré	16
Exercice 43 – Caractères jusqu'à l'étoile	16
Exercice 44 – Maximum de cinq nombres	16
Exercice 45 – Nombres parfaits	16
Exercice 46 – Nombres amicaux	16
Exercice 47 – Suites récurrentes	16
Exercice 48 – Séries numériques	17

Partie 3 : Tableaux et chaînes **20**

Exercice 49 – Afficher un tableau	20
Exercice 50 – Somme et moyenne	20
Exercice 51 – Minimum et maximum	20
Exercice 52 – Occurrences	20
Exercice 53 – Inversion	20
Exercice 54 – Décalage à droite	20
Exercice 55 – Recherche linéaire	20
Exercice 56 – Tri par sélection	21
Exercice 57 – Fusion de tableaux	21
Exercice 58 – Matrice identité	21
Exercice 59 – Somme des diagonales	21
Exercice 60 – Longueur d'une chaîne	21

Exercice 61 – Voyelles	21
Exercice 62 – Copie bornée	21
Exercice 63 – Palindrome	22
Exercice 64 – Rotation circulaire	22
Exercice 65 – Fusion de tableaux triés	22
Exercice 66 – Plus longue montée	22
Exercice 67 – Compression RLE	22
Exercice 68 – Fréquence des mots	22
Exercice 69 – Boîte à outils pour tableaux	23
Exercice 70 – Tableau trié et tableau premier	23
Exercice 71 – Couples amicaux dans un tableau	23
Exercice 72 – Chaînes : comparer, rechercher et concaténer	23
Exercice 73 – Occurrences et valeurs distinctes	23

Partie 4 : Pointeurs, structures et mémoire **26**

Exercice 74 – Adresse et déréférencement	26
Exercice 75 – Somme par pointeurs	26
Exercice 76 – Minimum avec indice	26
Exercice 77 – Allocation d'un entier	26
Exercice 78 – Tableau dynamique	26
Exercice 79 – Compteurs avec calloc	26
Exercice 80 – Capacité avec realloc	26
Exercice 81 – Structure Point	27
Exercice 82 – Structure Étudiant	27
Exercice 83 – Pointeur de structure	27
Exercice 84 – Liste : insertion en tête	27
Exercice 85 – Liste : parcours	27
Exercice 86 – Liste : libération	27
Exercice 87 – Détection de fuite	27
Exercice 88 – Const et pointeurs	28
Exercice 89 – Pointeur de fonction	28
Exercice 90 – Matrice dynamique contiguë	28
Exercice 91 – Intervalles et structures	28
Exercice 92 – Liste doublement chaînée	28
Exercice 93 – Répertoire dynamique	28

Exercice 94 – Bibliothèque de fonctions	29
Exercice 95 – Recherche dichotomique et concaténation	29
Exercice 96 – Recréer des fonctions de chaînes	29
Exercice 97 – Conversion et transformations de chaînes	29
Exercice 98 – Bibliothèque de matrices	29

Partie 5 : Fichiers, modularité et mini-projets **32**

Exercice 99 – Écriture d'un fichier texte	32
Exercice 100 – Lecture ligne par ligne	32
Exercice 101 – Nombre de lignes	32
Exercice 102 – Somme depuis un fichier	32
Exercice 103 – Copie binaire	32
Exercice 104 – Module statistiques	32
Exercice 105 – Arguments de la commande	33
Exercice 106 – Conversion d'argument	33
Exercice 107 – Journal de notes	33
Exercice 108 – Tri d'étudiants	33
Exercice 109 – Menu persistant	33
Exercice 110 – Analyseur de texte	33
Exercice 111 – Persistance binaire	34
Exercice 112 – Tests unitaires	34
Exercice 113 – Carnet de contacts	34
Exercice 114 – Filtrage CSV	34
Exercice 115 – Histogramme d'octets	34
Exercice 116 – Recherche en ligne de commande	34
Exercice 117 – Index binaire	35
Exercice 118 – Fusion de journaux triés	35

Objectifs

- Écrire, compiler et tester un programme C11 complet.
- Maîtriser les expressions, conditions, boucles, fonctions, tableaux et chaînes.
- Manipuler des pointeurs, structures et fichiers en contrôlant les erreurs.

Consignes de travail

Les exercices de chaque partie sont rangés par difficulté croissante. Pour chaque exercice, créer un fichier C indépendant, vérifier les entrées indiquées, puis tester un cas ordinaire et un cas limite. Compiler avec `gcc -std=c11 -Wall -Wextra -Wpedantic`. Pour l'exercice de modularité, utiliser les trois fichiers demandés.

```
gcc -std=c11 -Wall -Wextra -Wpedantic exercice.c -lm -o exercice
```

1

PARTIE

Environnement, types et expressions

NOTIONS

types, variables, entrées-sorties, opérateurs et conversions

COMPÉTENCES

écrire, compiler et tester des programmes C simples

PROGRESSION

Initiation → application

Langage C | CHEATSHEET DE LA PARTIE 1

Environnement, types et expressions

Notions de la partie, code C commenté et tests repères.

Notions essentielles

Structure minimale : `int main(void) { ... return 0; }`. Un programme est compilé puis exécuté ; distinguer erreur de compilation, erreur d'exécution et résultat incorrect.

Types usuels : `int` pour les entiers, `float` et `double` pour les réels, `char` pour un caractère. Une variable doit être déclarée avant son utilisation et initialisée avant sa lecture.

Entrées-sorties : `printf` affiche ; `scanf` lit et renvoie le nombre de conversions réussies. Les formats doivent correspondre aux types.

Exemples C commentés

Expressions : `+` `-` `*` `/` `%` ; pour deux entiers, `/` réalise une division entière et `%` fournit le reste.

Conversions : `(double)a / b` force une division réelle. Les parenthèses rendent la priorité explicite ; attention aux dépassements et aux divisions par zéro.

Patron de test : choisir une valeur ordinaire, une valeur nulle ou minimale et une valeur limite ; vérifier les unités, les décimales et le format de sortie.

Code :

```
if (scanf("%lf", &x) == 1) { printf("%.2f", x); }  
// Le retour de scanf confirme que la donnée a été lue.
```

Partie 1 : Environnement, types et expressions



Exercice 1 – Premier exécutable

Créer un fichier `programme.c` qui affiche sur trois lignes le titre du TD, votre nom et la commande de compilation. Résultat attendu : trois lignes lisibles, puis une fin normale du programme.



Exercice 2 – Types et tailles

Déclarer un `int`, un `float`, un `double` et un `char`. Afficher leur valeur et leur taille en octets. Expliquer pourquoi les tailles affichées peuvent dépendre de la machine.



Exercice 3 – Celsius et Fahrenheit

Lire une température réelle en degrés Celsius. Afficher la valeur correspondante en degrés Fahrenheit avec deux décimales. Tester notamment 0 °C et 100 °C.



Exercice 4 – Rectangle

Lire la longueur et la largeur d'un rectangle. Calculer et afficher son périmètre et son aire avec deux décimales. Les deux valeurs saisies sont supposées positives.



Exercice 5 – Division entière

Lire deux entiers, le diviseur étant non nul. Afficher le quotient entier et le reste de leur division.



Exercice 6 – Échange

Lire deux entiers et afficher leurs valeurs avant et après un échange à l'aide d'une variable temporaire. Exemple : 3 et 8 deviennent 8 et 3.



Exercice 7 – Conversion de secondes

Lire un nombre entier de secondes, positif ou nul. L'afficher sous la forme heures, minutes et secondes à l'aide des divisions et des restes.



Exercice 8 – Moyenne pondérée

Lire deux notes et deux coefficients strictement positifs. Afficher la moyenne pondérée avec deux décimales. Les valeurs saisies sont supposées valides.



Exercice 9 – Codes de caractères

Lire deux caractères. Afficher chaque caractère et son code numérique dans le jeu de caractères utilisé par la machine.



Exercice 10 – Dernier chiffre

Lire un entier positif ou nul. Afficher son dernier chiffre à l'aide du reste de la division par 10.



Exercice 11 – Somme de durées

Lire deux durées sous la forme heures minutes secondes. Calculer leur somme en secondes, puis afficher le total en heures, minutes et secondes normalisées.



Exercice 12 – Puissance carrée et cube

Lire un réel x . Calculer et afficher x^2 et x^3 par des multiplications, sans utiliser de boucle ni de fonction mathématique.



Exercice 13 – Expressions équivalentes

Lire deux réels a et b . Calculer séparément $a^2+2ab+b^2$ et $(a+b)^2$, puis afficher les deux résultats.



Exercice 14 – Budget TTC

Lire un prix hors taxe et un taux de TVA. Calculer et afficher le montant de TVA et le prix TTC avec deux décimales. Les valeurs saisies sont supposées positives ou nulles.



Exercice 15 – Formules de base

Lire deux entiers a et b. Afficher leur somme, leur différence, leur produit et, si b est non nul, le quotient réel ; on supposera b non nul pour l'essai.



Exercice 16 – Conversions de bases

Lire un entier naturel compris entre 0 et 255. Afficher sa représentation décimale, hexadécimale et octale avec les formats de printf.



Exercice 17 – Échange par addition

Lire deux entiers et les échanger sans variable temporaire avec trois opérations arithmétiques. Les valeurs sont choisies de façon à éviter un dépassement.



Exercice 18 – Couleur RGB compacte

Lire trois composantes R, G et B comprises entre 0 et 255. *Rappel* : dans 0xRRGGBB, R occupe les bits de poids fort, puis G, puis B. Construire le code par décalages et disjonctions bit à bit, puis afficher le code.



Exercice 19 – Prix en centimes

Lire un montant en euros avec au plus deux décimales. Le convertir en centimes par une multiplication et afficher le montant en euros et centimes.



Exercice 20 – Moyenne de trois mesures

Lire trois mesures réelles. Calculer leur moyenne, la somme des écarts à cette moyenne et la somme des carrés des écarts. Afficher les résultats avec quatre décimales.

2

PARTIE

Conditions, boucles et fonctions

NOTIONS

conditions, répétitions, validation, fonctions et récursivité

COMPÉTENCES

concevoir des algorithmes contrôlés et les découper en fonctions

PROGRESSION

Application → approfondissement

Langage C | CHEATSHEET DE LA PARTIE 2

Conditions, boucles et fonctions

Notions de la partie, code C commenté et tests repères.

Notions essentielles

Choix : `if/else` traite une condition ; `switch` est adapté à un choix discret. Toujours prévoir le cas invalide et les valeurs limites.

Boucles : `for` convient lorsque le nombre de tours est connu ; `while` lorsque la condition d'arrêt est testée avant ; `do...while` garantit au moins une saisie.

Fonction : prototype, paramètres, type de retour et contrat. Une fonction doit réaliser une tâche claire et ne pas dépendre inutilement de variables globales.

Exemples C commentés

Validation :

```
do { lire(); } while (valeur < min || valeur > max);
```

Compteur et accumulateur : initialiser avant la boucle, mettre à jour à chaque tour, puis afficher après la boucle.

Récursivité : identifier un cas de base et réduire l'entrée à un cas plus petit. Pour Newton, le PGCD ou Fibonacci, vérifier l'arrêt et éviter une récursion inutilement coûteuse.

PGCD : `while (b != 0) { int r = a % b; a = b; b = r; }`

// Chaque tour réduit le second opérande.

Partie 2 : Conditions, boucles et fonctions



Exercice 21 – Signe et parité

Lire un entier, indiquer s'il est négatif, nul ou positif, puis pair ou impair. Vérifier le cas zéro.



Exercice 22 – Maximum de trois

Écrire une fonction qui retourne le maximum de deux entiers. Réutiliser cette fonction pour afficher le maximum de trois entiers lus.



Exercice 23 – Année bissextile

Lire une année strictement positive. Appliquer la règle du calendrier grégorien et afficher si elle est bissextile. Tester 1900, 2000 et 2024.



Exercice 24 – Mention

Lire une note entre 0 et 20. Afficher « insuffisant » sous 10, « passable » dès 10, « assez bien » dès 12, « bien » dès 14 et « très bien » dès 16.



Exercice 25 – Menu calculatrice

Lire deux réels et un opérateur parmi +, -, *, et /. Afficher le résultat ou un message d'erreur pour une opération inconnue ou une division par zéro.



Exercice 26 – Somme de 1 à n

Lire un entier n entre 0 et 100000. Calculer $1+2+\dots+n$ à l'aide d'une boucle, puis comparer avec $n(n+1)/2$.



Exercice 27 – Table de multiplication

Lire un entier et afficher les onze produits de sa table, de 0 à 10, chacun sur une ligne.



Exercice 28 – Factorielle

Lire un entier n de 0 à 20. Calculer $n!$ avec une boucle et afficher le résultat. Expliquer le cas 0!.



Exercice 29 – PGCD

Lire deux entiers positifs ou nuls, distincts de (0,0). *Rappel* : le PGCD est le plus grand diviseur commun ; l'algorithme d'Euclide remplace (a, b) par $(b, a \bmod b)$ jusqu'à obtenir un reste nul. Tester (18,12), (7,5) et (0,9).



Exercice 30 – Nombre premier

Lire un entier naturel et afficher s'il est premier. *Rappel* : un nombre premier supérieur ou égal à 2 possède exactement deux diviseurs positifs, 1 et lui-même ; il suffit de tester les diviseurs jusqu'à $\sqrt{n}$. Tester 1, 2, 9 et 17.



Exercice 31 – Fibonacci

Lire n entre 0 et 90. *Rappel* : la suite commence par $F_0 = 0, F_1 = 1$ et vérifie $F_n = F_{n-1} + F_{n-2}$. Afficher les n premiers termes sans récursion.



Exercice 32 – Nombre mystère

Fixer la valeur secrète à 42. Demander des propositions entières jusqu'à la bonne réponse ; après chaque erreur, indiquer « trop petit » ou « trop grand » et compter les tentatives.



Exercice 33 – Saisie contrôlée

Demander une note entière comprise entre 0 et 20. Répéter la saisie tant que la valeur est hors intervalle ou non numérique. Afficher la note valide.



Exercice 34 – Compter un chiffre

Lire un entier positif ou nul et un chiffre entre 0 et 9. Afficher le nombre d'occurrences de ce chiffre dans l'écriture décimale ; traiter spécialement l'entier 0.



Exercice 35 – Triangle

Lire trois longueurs réelles strictement positives. *Rappel* : trois longueurs forment un triangle si chaque longueur est strictement inférieure à la somme des deux autres ; équilatéral signifie trois côtés égaux et isocèle au moins deux côtés égaux. Vérifier puis classer le triangle.



Exercice 36 – Suite de Collatz

Lire un entier naturel strictement positif. *Rappel* : la transformation de Collatz remplace n par $n/2$ si n est pair et par $3n + 1$ s'il est impair. Afficher la suite jusqu'à 1, compter les étapes et arrêter avec un message d'erreur si le compteur dépasse une limite de sécurité.



Exercice 37 – Décomposition en facteurs premiers

Lire un entier n entre 2 et 2 000 000. *Rappel* : la décomposition en facteurs premiers écrit n comme un produit de nombres premiers ; une multiplicité indique combien de fois un facteur apparaît, par exemple $12 = 2^2 \times 3$. Afficher cette décomposition par divisions successives.



Exercice 38 – Racine par Newton

Lire un réel positif ou nul et une précision strictement positive. *Rappel* : pour calculer $\sqrt{a}$, Newton utilise $x_{k+1} = \frac{1}{2}(x_k + a/x_k)$ avec une approximation initiale positive ; arrêter lorsque $|x_{k+1} - x_k|$ est inférieur à la précision.



Exercice 39 – Palindrome numérique

Lire un entier naturel et déterminer, sans convertir en chaîne, s'il est un palindrome. *Rappel* : un palindrome reste identique lorsqu'on lit ses chiffres de gauche à droite ou de droite à gauche ; l'inversion se construit avec le dernier chiffre obtenu par le reste modulo 10.



Exercice 40 – Jour suivant

Lire une date valide au format jour mois année. *Rappel* : une année est bissextile si elle est divisible par 400, ou divisible par 4 mais pas par 100 ; février possède alors 29 jours. Afficher la date du lendemain en tenant compte des mois de 30 ou 31 jours.



Exercice 41 – Lettre, casse et jour

Écrire un programme qui demande une lettre et indique si elle est égale à « a » ou « A ». Indiquer ensuite si elle est une majuscule ou une minuscule. Ajouter un choix entier de 1 à 7 et afficher le jour correspondant.



Exercice 42 – Équation du second degré

Lire trois réels a , b et c et résoudre $ax^2 + bx + c = 0$. *Rappel* : le discriminant vaut $\Delta = b^2 - 4ac$; selon son signe, l'équation possède zéro, une ou deux solutions réelles. Traiter séparément le cas $a = 0$.



Exercice 43 – Caractères jusqu'à l'étoile

Lire des caractères un par un jusqu'à la saisie de "*". Afficher chaque caractère lu et compter les lettres "b" ainsi que les espaces.



Exercice 44 – Maximum de cinq nombres

Lire cinq nombres réels. Afficher le maximum et la position de sa première apparition.



Exercice 45 – Nombres parfaits

Lire un entier N . *Rappel* : un nombre parfait est égal à la somme de ses diviseurs positifs strictement inférieurs à lui, par exemple $6 = 1 + 2 + 3$. Indiquer si N est parfait puis afficher les nombres parfaits strictement inférieurs à N .



Exercice 46 – Nombres amicaux

Lire deux entiers positifs. *Rappel* : deux entiers sont amicaux si la somme des diviseurs propres du premier vaut le second et réciproquement. Vérifier cette propriété.



Exercice 47 – Suites récurrentes

Pour $U_0 = U_1 = 4$ et $U_n = 3U_{n-1} + 4U_{n-2} + 6$, lire n et calculer U_n , puis afficher U_0 à U_{99} . Pour $T_0 = 1$, $T_1 = x$ et $T_n = 2xT_{n-1} - T_{n-2}$, calculer également T_n pour un $x \in [-1, 1]$.



Exercice 48 – Séries numériques

Calculer $K!$, puis la somme $\sum_{k=0}^n x^k / k!$. Améliorer le programme pour approcher e^x à 10^{-6} près, puis approcher π par $\pi = \sqrt{6 \sum_{k=1}^{\infty} 1/k^2}$.

3 PARTIE

Tableaux et chaînes

NOTIONS

tableaux, matrices, chaînes, parcours, recherche et tri

COMPÉTENCES

organiser, parcourir et transformer des données en mémoire

PROGRESSION

Application → défi

Langage C | CHEATSHEET DE LA PARTIE 3

Tableaux et chaînes

Notions de la partie, code C commenté et tests repères.

Notions essentielles

Tableau : une suite contiguë de cases de même type, indexées de 0 à $n-1$. Ne jamais accéder à l'indice n . La taille réelle doit être transmise aux fonctions avec le tableau.

Chaîne C : tableau de caractères terminé par le caractère nul `'\0'`. Une capacité de 100 autorise au plus 99 caractères utiles.

Matrice : deux indices, ligne puis colonne ; les deux boucles doivent respecter les dimensions déclarées.

Exemples C commentés

Parcours :

```
for (size_t i = 0; i < n; ++i) ...
```

Recherche linéaire : comparer chaque case jusqu'à trouver la valeur. **Tri par sélection** : choisir le minimum de la partie non triée puis l'échanger avec sa première case.

Chaînes : préférer `fgets` pour une ligne ; supprimer le saut final si nécessaire. Vérifier la capacité avant toute copie ou concaténation.

```
Recherche : for (size_t i = 0; i < n; ++i)
    if (tab[i] == cible) return (int)i; // première
    position
return -1; // valeur absente
```

Partie 3 : Tableaux et chaînes



Exercice 49 – Afficher un tableau

Lire une taille n entre 1 et 20, puis n entiers. Écrire une fonction d’affichage recevant le tableau et sa taille ; afficher les valeurs dans l’ordre saisi.



Exercice 50 – Somme et moyenne

Lire n entre 1 et 20, puis n entiers. Afficher leur somme et leur moyenne réelle avec deux décimales. Vérifier que n n’est pas nul.



Exercice 51 – Minimum et maximum

Lire un tableau non vide de n entiers, avec $1 \leq n \leq 20$. Écrire une fonction qui calcule simultanément le minimum et le maximum et afficher ces deux valeurs.



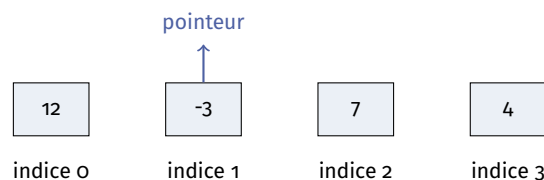
Exercice 52 – Occurrences

Lire n entiers et une valeur recherchée. Afficher combien de fois cette valeur apparaît dans le tableau. Exemple : $[2,3,2]$ et 2 donnent 2.



Exercice 53 – Inversion

Lire un tableau de n entiers, puis inverser ses éléments sur place sans second tableau. Afficher le résultat ; tester $n=1$ et n pair.



Exercice 54 – Décalage à droite

Lire n entiers et une nouvelle valeur. Décaler les éléments d’une position vers la droite, perdre le dernier élément et placer la nouvelle valeur en tête.



Exercice 55 – Recherche linéaire

Lire n entiers et une valeur. Retourner l'indice de sa première occurrence ou -1 si elle est absente ; les indices commencent à zéro.



Exercice 56 – Tri par sélection

Lire n entiers et les trier par ordre croissant au moyen du tri par sélection. Afficher le tableau avant et après le tri.



Exercice 57 – Fusion de tableaux

Lire deux tableaux de tailles n et m avec $n+m \leq 20$. Copier leurs éléments dans un troisième tableau : d'abord ceux du premier, puis ceux du second. Afficher le résultat.



Exercice 58 – Matrice identité

Lire une taille n entre 1 et 10. Construire et afficher la matrice identité $n \times n$ à l'aide de deux boucles imbriquées.



Exercice 59 – Somme des diagonales

Lire une matrice carrée d'entiers de taille n entre 1 et 10. Additionner les deux diagonales sans compter deux fois l'élément central lorsque n est impair.



Exercice 60 – Longueur d'une chaîne

Lire une ligne de 99 caractères au plus avec `fgets`. Calculer sa longueur sans `strlen`, sans compter le saut de ligne final.



Exercice 61 – Voyelles

Lire une ligne et compter les lettres a, e, i, o, u et y en minuscules. Ignorer le saut de ligne de `fgets`.



Exercice 62 – Copie bornée

Écrire une fonction qui copie une chaîne dans un tableau destination de capacité donnée. Elle doit toujours terminer la destination par un caractère nul et indiquer si la copie est complète.



Exercice 63 – Palindrome

Lire un mot de 99 caractères au plus. *Rappel* : un mot est palindrome si le premier caractère est égal au dernier, le deuxième à l'avant-dernier, et ainsi de suite. Comparer les caractères symétriques pour conclure.



Exercice 64 – Rotation circulaire

Lire un tableau d'entiers puis une valeur k . *Rappel* : une rotation gauche de k places envoie l'élément d'indice i vers l'indice $(i - k) \bmod n$; une rotation de n places ne change rien. Effectuer la rotation sans second tableau et en normalisant k .



Exercice 65 – Fusion de tableaux triés

Lire deux tableaux déjà triés dans l'ordre croissant. *Rappel* : la fusion compare les deux premiers éléments encore disponibles, recopie le plus petit, puis avance dans le tableau correspondant. Les doublons sont conservés. Produire un troisième tableau trié sans tri global.



Exercice 66 – Plus longue montée

Lire un tableau d'entiers et rechercher la plus longue sous-suite contiguë strictement croissante. *Rappel* : contiguë signifie que les indices se suivent sans trou ; strictement croissante signifie $a[i] > a[i - 1]$. Afficher son indice de début, sa longueur et ses valeurs.



Exercice 67 – Compression RLE

Lire une ligne de caractères et produire sa compression RLE (Run-Length Encoding). *Rappel* : RLE remplace chaque séquence de caractères identiques par le caractère suivi de sa longueur ; ainsi aaabbbc devient a3b2c1. Prévoir une capacité suffisante et expliquer la limite du format.



Exercice 68 – Fréquence des mots

Lire une phrase de 200 caractères au plus. *Rappel* : un mot est ici une suite de caractères alphanumériques ; deux mots sont égaux après conversion ASCII en minuscules. Compter les occurrences dans deux tableaux de chaînes et conserver l'ordre de première apparition.



Exercice 69 – Boîte à outils pour tableaux

Lire un tableau d'entiers puis afficher son minimum, son maximum, sa moyenne, le nombre d'occurrences de chaque valeur, son inversion et son tri croissant. Découper le programme en fonctions.



Exercice 70 – Tableau trié et tableau premier

Lire un tableau d'entiers et vérifier s'il est trié dans l'ordre croissant. Vérifier également si tous ses éléments sont des nombres premiers.



Exercice 71 – Couples amicaux dans un tableau

Lire un tableau d'entiers positifs et afficher chaque couple d'éléments distincts qui sont amicaux. Ne pas afficher deux fois le même couple.



Exercice 72 – Chaînes : comparer, rechercher et concaténer

Lire deux chaînes. Les comparer, rechercher la seconde comme sous-chaîne de la première, copier la première dans une troisième chaîne, puis concaténer les deux dans une destination suffisamment grande.



Exercice 73 – Occurrences et valeurs distinctes

Lire un tableau et une valeur x. Afficher les positions de x puis supprimer toutes ses occurrences. Refaire l'opération en supposant le tableau trié. Enfin, construire un tableau ne contenant qu'une occurrence de chaque valeur.

4

PARTIE

Pointeurs, structures et mémoire

NOTIONS

adresses, pointeurs, allocation dynamique, structures et listes

COMPÉTENCES

gérer explicitement la mémoire et modéliser des données composées

PROGRESSION

Approfondissement → maîtrise

Langage C | CHEATSHEET DE LA PARTIE 4

Pointeurs, structures et mémoire

Notions de la partie, code C commenté et tests repères.

Notions essentielles

Un pointeur contient une adresse. `*p` déréférence cette adresse et `&x` obtient l'adresse de `x`. Avant `*p`, vérifier que `p != NULL`.

Passage par adresse : une fonction peut modifier une variable en recevant son adresse. Pour un tableau, le paramètre est déjà une adresse vers sa première case.

Structure : utiliser `objet.champ` avec une valeur et `pointeur->champ` avec un pointeur.

Exemples C commentés

Cycle de vie :

```
p = malloc(n * sizeof *p);  
vérifier le résultat, utiliser le bloc dans ses limites,  
puis appeler free(p) exactement une fois. Avec realloc,  
utiliser un pointeur temporaire afin de ne pas perdre  
l'ancien bloc en cas d'échec.
```

Liste : créer un nœud, relier son champ suivant, mettre à jour la tête, parcourir jusqu'à `NULL`, puis libérer chaque nœud avant de quitter le programme.

Échange :

```
void echanger(int *a, int *b) { int t = *a;  
*a = *b; *b = t; }
```

// Un pointeur permet de modifier les variables de l'appelant.

Partie 4 : Pointeurs, structures et mémoire



Exercice 74 – Adresse et déréférencement

Lire un entier, afficher son adresse et sa valeur, puis ajouter 5 à cette valeur au moyen d'un pointeur. Afficher la nouvelle valeur.



Exercice 75 – Somme par pointeurs

Lire n entiers, avec $1 \leq n \leq 20$. Calculer leur somme dans une fonction qui avance un pointeur de début vers la fin du tableau.



Exercice 76 – Minimum avec indice

Lire un tableau non vide. Écrire une fonction qui renvoie son minimum et écrit, via un pointeur, l'indice de sa première occurrence.



Exercice 77 – Allocation d'un entier

Lire un entier, lui réserver dynamiquement une case, afficher la valeur via le pointeur, puis libérer cette case. Traiter un échec d'allocation.



Exercice 78 – Tableau dynamique

Lire n entre 1 et 1000, puis n entiers. Allouer exactement n cases, calculer la moyenne réelle et libérer le tableau, y compris si une saisie échoue.



Exercice 79 – Compteurs avec calloc

Lire n entre 1 et 1000. Allouer n compteurs initialisés à zéro, incrémenter le premier et le dernier, les afficher, puis libérer la mémoire.



Exercice 80 – Capacité avec realloc

Lire des entiers positifs jusqu'à la sentinelle -1. *Rappel* : `realloc` peut déplacer un bloc ; utiliser un pointeur temporaire et ne remplacer l'ancien pointeur qu'après une réussite.

Stocker les valeurs dans un tableau dont la capacité double lorsqu'il est plein, afficher puis libérer.



Exercice 81 – Structure Point

Définir une structure `Point` avec deux coordonnées réelles. Lire un point et afficher le carré de sa distance à l'origine, sans calculer de racine.



Exercice 82 – Structure Étudiant

Définir un étudiant avec nom, identifiant et note. Lire n étudiants, $1 \leq n \leq 10$, puis afficher le nom et la note de ceux qui ont obtenu au moins 10.



Exercice 83 – Pointeur de structure

Lire le nom et la note d'un étudiant, puis un bonus positif. Écrire une fonction qui ajoute le bonus à la note via un pointeur, sans dépasser 20.



Exercice 84 – Liste : insertion en tête

Lire n entiers, avec $0 \leq n \leq 20$. *Rappel* : un nœud contient une valeur et un pointeur vers le nœud suivant ; insérer en tête signifie créer le nouveau nœud puis faire pointer sa liaison vers l'ancienne tête. Afficher la liste et libérer tous ses nœuds.



Exercice 85 – Liste : parcours

Créer une liste de trois entiers, puis écrire une fonction qui parcourt la liste et retourne son nombre de nœuds. Afficher ce nombre et libérer la liste.



Exercice 86 – Liste : libération

Créer une liste de n nœuds. Écrire une fonction qui libère tous les nœuds sans relire un nœud libéré ; la tête doit devenir nulle après l'appel.



Exercice 87 – Détection de fuite

Un programme alloue deux tableaux mais ne libère que le premier. Écrire une version corrigée qui vérifie les allocations, utilise les deux tableaux et libère les deux sur tous les chemins.



Exercice 88 – Const et pointeurs

Écrire un programme qui utilise `const int *p`, `int * const q` et `const int * const r`. Montrer par le code ce qui peut être modifié et ce qui reste constant.



Exercice 89 – Pointeur de fonction

Définir trois opérations arithmétiques et une fonction d’affichage qui reçoit un pointeur de fonction. Lire deux entiers et un choix, puis appeler l’opération sélectionnée sans recopier la logique du menu.



Exercice 90 – Matrice dynamique contiguë

Allouer une matrice d’entiers de dimensions n et m dans un seul bloc contigu. La remplir, afficher ses sommes de lignes et libérer correctement la mémoire, y compris en cas d’échec.



Exercice 91 – Intervalles et structures

Définir une structure représentant un intervalle fermé [début, fin]. Lire deux intervalles valides, afficher leur intersection si elle existe et calculer sa longueur.



Exercice 92 – Liste doublement chaînée

Créer une liste doublement chaînée d’entiers. *Rappel* : chaque nœud possède deux liens, `precedent` et `suivant` ; après une suppression, les deux voisins doivent être reconnectés. Insérer en tête et en queue, supprimer la première occurrence d’une valeur, parcourir dans les deux sens puis libérer la liste.



Exercice 93 – Répertoire dynamique

Gérer un tableau dynamique de structures contenant un nom et un numéro. *Rappel* : la capacité est le nombre de cases réservées et la taille le nombre de contacts présents ; agrandir avec `realloc` sans perdre l'ancien bloc en cas d'échec. Ajouter jusqu'à la sentinelle, rechercher puis agrandir par paliers contrôlés.



Exercice 94 – Bibliothèque de fonctions

Écrire des fonctions pour afficher une mention selon une note, lire deux variables globales, calculer une somme et un produit, retourner le minimum et le maximum de deux entiers, puis remplacer un entier par son carré au moyen d'un pointeur. Tester toutes les fonctions dans `main`.



Exercice 95 – Recherche dichotomique et concaténation

Écrire des fonctions de lecture, affichage, inversion, tri, recherche linéaire et recherche dichotomique dans un tableau trié. Ajouter un second tableau à la fin du premier avec `realloc`. Prévoir un menu de test.



Exercice 96 – Recréer des fonctions de chaînes

Réimplémenter, sans appeler les fonctions homonymes de `string.h`, les opérations de copie, copie bornée, comparaison et concaténation bornée. Tester chaque fonction avec des chaînes suffisamment dimensionnées.



Exercice 97 – Conversion et transformations de chaînes

Écrire une fonction qui convertit une chaîne décimale correcte en entier, une fonction qui inverse une chaîne sur place et une fonction qui remplace toutes les occurrences d'un caractère par un autre. Tester les trois fonctions.



Exercice 98 – Bibliothèque de matrices

Écrire des fonctions pour lire et afficher une matrice, calculer la somme et le produit de deux matrices compatibles, vérifier si une matrice est nulle et rechercher la première position d'une valeur. Proposer un menu de test.

5

PARTIE

Fichiers, modularité et mini-projets

NOTIONS

fichiers texte et binaires, arguments, modularité et tests

COMPÉTENCES

construire des programmes persistants, modulaires et vérifiables

PROGRESSION

Maîtrise → projet

Langage C | CHEATSHEET DE LA PARTIE 5

Fichiers, modularité et mini-projets

Notions de la partie, code C commenté et tests repères.

Notions essentielles

Fichier texte : ouvrir avec `fopen`, vérifier le résultat, lire ou écrire, puis fermer avec `fclose`. Les modes "r", "w" et "a" correspondent respectivement à lecture, écrasement et ajout.

Fichier binaire : `fread` et `fwrite` manipulent des blocs d'octets ou d'enregistrements ; vérifier le nombre d'éléments traité. EOF indique la fin d'une lecture caractère par caractère.

Arguments : `argc` compte les arguments et `argv` contient leurs chaînes.

Exemples C commentés

Modularité : placer les prototypes dans un fichier `.h`, les définitions dans un fichier `.c`, puis compiler tous les modules :

```
gcc -std=c11 -Wall -Wextra -Wpedantic main.c stats.c  
-o programme
```

Robustesse : contrôler chaque ouverture, conversion, lecture, écriture et allocation ; fermer les fichiers même dans les chemins d'erreur.

Tests : tester fichier vide, ligne finale sans saut, argument manquant, entrée invalide et cas nominal.

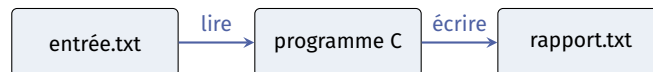
```
Lecture : while (fgets(ligne, sizeof ligne, f) != NULL)  
    fputs(ligne, stdout); // traiter chaque ligne  
// Toujours tester fopen avant d'utiliser le flux.
```

Partie 5 : Fichiers, modularité et mini-projets



Exercice 99 – Écriture d'un fichier texte

Un outil de supervision doit produire un rapport quotidien. Créer `sortie.txt`, y écrire trois lignes de texte, fermer le fichier et signaler toute erreur d'ouverture, d'écriture ou de fermeture.



ouvrir → vérifier → traiter → fermer



Exercice 100 – Lecture ligne par ligne

Un service d'analyse traite un journal d'incidents. Ouvrir `entree.txt` et afficher son contenu ligne par ligne avec un tampon de 256 caractères. Vérifier les erreurs de lecture et fermer le fichier.



Exercice 101 – Nombre de lignes

Ouvrir `entree.txt`, compter ses lignes, y compris une dernière ligne sans saut final, et afficher le résultat. Le fichier vide compte zéro ligne.



Exercice 102 – Somme depuis un fichier

Lire tous les entiers de `entree.txt` séparés par des espaces. *Rappel* : `fscanf` renvoie le nombre de conversions réussies ; EOF signifie la fin du fichier, tandis qu'un résultat nul indique un caractère invalide. Afficher la somme et distinguer ces deux situations.



Exercice 103 – Copie binaire

Copier `source.bin` vers `copie.bin` par blocs d'octets. *Rappel* : en mode binaire, `fread` renvoie le nombre d'octets effectivement lus et `fwrite` le nombre d'éléments écrits ; vérifier ces valeurs ainsi que l'ouverture et la fermeture.



Exercice 104 – Module statistiques

Écrire une fonction `somme` et l'appeler depuis `main`. *Rappel* : un fichier d'en-tête contient les déclarations partagées ; le fichier source contient les définitions et l'édition de liens rassemble les fichiers objets. Déplacer le prototype dans `ex104_stats.h`, la définition dans `ex104_stats.c` et le programme principal dans `ex104_main.c`, puis compiler les deux fichiers C ensemble.



Exercice 105 – Arguments de la commande

Écrire un programme qui affiche le nombre d'arguments reçus et chacun de leurs contenus. *Rappel* : dans `main(int argc, char *argv[])`, `argc` compte les arguments et `argv[0]` contient le nom du programme.



Exercice 106 – Conversion d'argument

Recevoir un entier décimal comme unique argument de la commande. *Rappel* : `strtol` fournit l'adresse du premier caractère non converti via `endptr` ; une conversion est valide seulement si toute la chaîne a été consommée et si aucune erreur de plage n'est signalée. Rejeter un texte invalide ou une valeur hors plage.



Exercice 107 – Journal de notes

Lire dans `notes.txt` des notes entières entre 0 et 20, puis écrire dans `rapport.txt` la moyenne et le nombre de notes au moins égales à 10. Gérer un fichier vide.



Exercice 108 – Tri d'étudiants

Lire `n` étudiants, chacun avec nom et note, puis les trier par note décroissante et, à note égale, par nom croissant. *Rappel* : un tri décroissant place la plus grande note en premier ; pour comparer des chaînes, utiliser `strcmp`, qui renvoie une valeur négative, nulle ou positive. Afficher la liste triée.



Exercice 109 – Menu persistant

Créer un menu qui permet d'ajouter un entier, d'afficher les valeurs stockées ou de supprimer la dernière valeur. Répéter jusqu'au choix zéro ; capacité maximale : 20 valeurs.



Exercice 110 – Analyseur de texte

Ouvrir `entree.txt` et compter ses octets, ses lignes et ses mots. *Rappel* : un mot est une suite de caractères non blancs ; une ligne est comptée dès qu'un saut de ligne est rencontré, et la dernière ligne sans saut final compte également.



Exercice 111 – Persistance binaire

Écrire trois enregistrements (identifiant entier et note réelle) dans `etudiants.bin`, puis les relire et les afficher. *Rappel* : `fwrite` et `fread` travaillent avec une taille d'élément et un nombre d'éléments ; vérifier les valeurs retournées pour détecter une écriture ou une lecture partielle.



Exercice 112 – Tests unitaires

Écrire des tests avec `assert` pour un maximum, un PGCD et une recherche dans un tableau. *Rappel* : `assert(condition)` arrête le programme si la condition est fausse ; couvrir au moins un cas ordinaire et une limite pour chaque fonction.



Exercice 113 – Carnet de contacts

Réaliser un petit carnet de dix contacts au plus. Le menu permet d'ajouter un nom et un téléphone, rechercher par nom, sauvegarder en texte et quitter. Vérifier les limites et les erreurs de fichier.



Exercice 114 – Filtrage CSV

Lire un fichier CSV contenant `nom;note`. *Rappel* : dans ce support, le séparateur CSV est le point-virgule ; une ligne valide contient un nom puis une note entre 0 et 20. Écrire dans un second fichier uniquement les lignes dont la note est au moins égale à 10, en vérifiant chaque conversion et chaque opération de fichier.



Exercice 115 – Histogramme d'octets

Lire un fichier binaire par blocs et calculer la fréquence des 256 valeurs d'octet. *Rappel* : un octet non signé prend une valeur de 0 à 255 ; le tableau des fréquences est donc indexé par la valeur lue. Afficher les cinq valeurs les plus fréquentes avec leur nombre d'occurrences.



Exercice 116 – Recherche en ligne de commande

Écrire un programme qui reçoit un mot et un nom de fichier dans `argv`. *Rappel* : les arguments suivent le nom du programme dans `argv[1]`, `argv[2]`, etc. Afficher les numéros des lignes contenant le mot comme sous-chaîne et retourner un code différent selon qu'une occurrence a été trouvée.



Exercice 117 – Index binaire

Créer un fichier binaire d'enregistrements de longueur fixe contenant un identifiant et une note. *Rappel* : pour atteindre l'enregistrement d'indice i , le déplacement vaut $i \times \text{sizeof}(\text{enregistrement})$ octets ; `fseek` déplace la position courante sans lire les enregistrements précédents.



Exercice 118 – Fusion de journaux triés

Lire deux fichiers texte dont chaque ligne commence par un horodatage entier croissant. *Rappel* : un horodatage plus petit doit être écrit en premier ; lorsqu'un fichier est épuisé, recopier directement le reste de l'autre fichier. Produire un troisième fichier contenant la fusion triée et signaler les lignes mal formées.