



SPÉ NSI · 1ÈRE

✓ EXIGIBLE AU BAC

PRIORITÉ P1

Types de base et structures de contrôle en Python

variables, conditions et boucles

🎯 À LA FIN DE CETTE FICHE, TU SAURAS...

- ✓ Reconnaître et utiliser les quatre types de base de Python : entier, flottant, chaîne de caractères, booléen.
- ✓ Écrire une structure conditionnelle (if / elif / else) pour exécuter des instructions selon une condition.
- ✓ Écrire une boucle for et une boucle while, et choisir la structure adaptée selon que le nombre de répétitions est connu ou non.

TON PARCOURS DE RÉVISION — coche chaque partie une fois révisée !

- | | |
|--|---|
| ☐ 1 · Les mots à connaître | ☐ 8 · La carte mentale |
| ☐ 2 · Les règles essentielles | ☐ 9 · Trouve l'erreur |
| ☐ 3 · Méthode A | ☐ 10 · Exercices type Bac |
| ☐ 4 · Méthode B | ☐ 11 · QCM chrono (5 min) |
| ☐ 5 · Exemple type Bac | ☐ 12 · Ta carte « 30 secondes » |
| ☐ 6 · Les pièges à éviter | ☐ ∞ Corrections (pour les parents) |
| ☐ 7 · Tips pour t'en souvenir | |

👤 LE COIN DES PARENTS — pas besoin de connaître le cours, tout est justifié.

1. Consultez avec votre enfant le calendrier des échéances (devoirs, DS, épreuves de spécialité, Grand Oral) et vérifiez qu'aucune date n'est oubliée.
2. Aidez-le à répartir les révisions de ce chapitre sur plusieurs jours (planning de répétition espacée J+1/J+3/J+7/J+14), sans vous prononcer sur le contenu du programme.
3. Observez les signaux d'alerte (fatigue persistante, stress, décrochage) et parlez-en avec lui, sans vous substituer aux professionnels du lycée.

1

Les mots à connaître *surligne ceux que tu confonds !*

Variable — un nom associé à une valeur en mémoire, dont le contenu peut changer pendant l'exécution du programme.

Type de donnée — la catégorie qui indique la nature d'une valeur (nombre entier, nombre à virgule, texte...) et les opérations qu'on peut lui appliquer.

Entier (int) — le type de donnée pour les nombres sans partie décimale (par exemple 3, -12 ou 0).

Flottant (float) — le type de donnée pour les nombres à virgule (par exemple 3.14 ou -0.5).

Chaîne de caractères (str) — le type de donnée pour du texte, toujours écrit entre guillemets (par exemple "bonjour").

Booléen (bool) — le type de donnée qui ne prend que deux valeurs possibles : True (vrai) ou False (faux).

Opérateur de comparaison — un symbole (==, !=, <, >, <=, >=) qui compare deux valeurs et renvoie toujours un booléen.

Structure de contrôle — le nom général donné aux instructions qui organisent l'ordre d'exécution d'un programme : les structures conditionnelles (if / elif / else) et les boucles (for, while).

Structure conditionnelle (if / elif / else) — une instruction qui exécute un bloc de code seulement si sa condition est vraie, avec d'éventuelles alternatives testées dans l'ordre d'écriture.

Boucle (for et while) — une instruction qui répète un bloc de code plusieurs fois : la boucle for répète un nombre de fois connu à l'avance (boucle bornée), la boucle while répète tant qu'une condition reste vraie (boucle non bornée).

Condition d'arrêt — la condition qui doit devenir fausse pour qu'une boucle while cesse de se répéter.

Affectation (=) — l'instruction qui donne une valeur à une variable ; à ne pas confondre avec l'opérateur de comparaison ==, qui compare deux valeurs sans en modifier aucune.

2

Les règles essentielles *chaque règle avec son exemple*

- 1 **Chaque valeur en Python a un type qui détermine sa nature et les opérations possibles : entier (int), flottant (float), chaîne de caractères (str) ou booléen (bool).**

exemple → 3 est un int, 3.0 est un float, "3" est un str et True est un bool : quatre valeurs différentes malgré une écriture proche.

- 2 **Une structure conditionnelle if / elif / else exécute un bloc de code seulement si sa condition est vraie ; elle passe au bloc suivant sinon.**

exemple → if note >= 10 : le programme affiche "Admis" ; sinon (else), il affiche "Non admis".

- 3 **Une boucle for répète un bloc d'instructions un nombre de fois connu à l'avance, en général en parcourant les éléments d'une séquence (liste, plage de nombres).**

exemple → for i in range(5) : répète le bloc 5 fois, avec i qui prend successivement les valeurs 0, 1, 2, 3, 4.

- 4 **Une boucle while répète un bloc d'instructions tant que sa condition reste vraie, sans nombre de répétitions fixé à l'avance.**

exemple → while compteur < 10 : répète le bloc tant que compteur est inférieur à 10.

- 5 Une boucle while doit toujours contenir une instruction qui rapproche la condition d'arrêt de devenir fausse, sinon elle se répète indéfiniment (boucle infinie).**

exemple → while compteur < 10 : compteur = compteur + 1, incrémente compteur à chaque tour jusqu'à ce que la condition devienne fausse.

- 6 Les opérateurs de comparaison (==, !=, <, >, <=, >=) comparent deux valeurs et renvoient toujours un booléen, utilisé comme condition.**

exemple → 5 > 3 renvoie True ; 5 == 3 renvoie False.

3 Méthode A — Choisir entre une boucle for et une boucle while

Exemple suivi : afficher les 5 premiers nombres, puis attendre qu'une valeur saisie dépasse 20.

- Étape 1** Se demander si le nombre de répétitions est connu AVANT le début de la boucle.
- Étape 2** Si oui (par exemple parcourir une liste ou une plage de nombres fixée) : utiliser une boucle for.
- Étape 3** Si non (le nombre de répétitions dépend d'un résultat calculé pendant l'exécution) : utiliser une boucle while, avec une condition d'arrêt qui doit évoluer à chaque tour.
- Étape 4** Vérifier que la boucle while contient bien une instruction qui modifie la variable testée par la condition, pour éviter une boucle infinie.

4 Méthode B — Écrire une structure conditionnelle à plusieurs cas

Exemple suivi : attribuer une mention à une note sur 20 : Félicitations si note >= 16, Bien si note >= 14, Passable si note >= 10, sinon Refusé.

- Étape 1** Écrire le premier cas avec if, en commençant par la condition la plus stricte (ici note >= 16).
- Étape 2** Ajouter chaque cas suivant avec elif, toujours du plus strict au moins strict : Python teste les conditions dans l'ordre et s'arrête à la première vraie.
- Étape 3** Terminer par else pour le cas restant, sans condition : il s'applique seulement quand tous les elif précédents sont faux.

5 Exemple type Bac — rédigé pas à pas

Écris un programme qui parcourt la liste notes = [8, 15, 12, 17, 9] et compte combien de notes sont supérieures ou égales à 10 (admisses).

Étape 1 — Créer une variable compteur initialisée à 0, qui comptera les notes admises.

Étape 2 — Utiliser une boucle for pour parcourir chaque note de la liste notes, une par une : le nombre de répétitions est connu à l'avance, 5 éléments.

Étape 3 — À l'intérieur de la boucle, utiliser une structure conditionnelle if pour tester si la note est supérieure ou égale à 10.

Étape 4 — Si la condition est vraie, augmenter compteur de 1 : compteur = compteur + 1.

Étape 5 — Dérouler le programme sur les 5 notes : 8 (condition fausse, compteur reste à 0), 15 (condition vraie, compteur passe à 1), 12 (condition vraie, compteur passe à 2), 17 (condition vraie, compteur passe à 3), 9 (condition fausse, compteur reste à 3).

Conclusion : Le programme affiche 3 : trois notes sur cinq sont supérieures ou égales à 10.

6 Les pièges à éviter *les erreurs classiques en DS...*

✗ Oublier les deux-points (:) à la fin de la ligne if / elif / else / for / while, obligatoires en Python.

✗ Confondre = (affectation, qui donne une valeur à une variable) et == (comparaison, qui renvoie un booléen).

✗ Écrire une boucle while dont la condition ne devient jamais fausse, faute d'instruction qui modifie la variable testée : boucle infinie.

✗ Croire qu'une boucle for peut remplacer n'importe quelle boucle while : elle ne convient que si le nombre de répétitions est connu avant le début de la boucle.

7 Tips — pour t'en souvenir

= affecte une valeur (une seule action), == compare deux valeurs (renvoie True ou False) : ne confonds jamais les deux.

Avant d'écrire une boucle, demande-toi : je connais le nombre de répétitions à l'avance ? Oui → for. Non → while, avec une condition d'arrêt qui doit évoluer.

Une boucle while sans instruction qui modifie la variable testée tourne à l'infini : vérifie toujours ce point avant d'exécuter ton code.

8 La carte mentale *dessiner, c'est mémoriser !*

Carte mentale à dessiner : « PYTHON : TYPES ET STRUCTURES DE CONTRÔLE » au centre, 3 branches colorées : TYPES DE BASE (int, float, str, bool), CONDITIONS (if/elif/else, du plus strict au moins strict), BOUCLES (for = bornée, while = non bornée + condition d'arrêt). Un exemple pour chaque branche. Redessine-la de mémoire après avoir lu la fiche.



ta carte mentale, de mémoire...

9 À toi de jouer — trouve l'erreur

Un élève veut afficher les nombres de 1 à 5 et écrit : $i = 1$, puis `while i < 5 : afficher i, puis  $i = i + 1$ .`

→ cherche l'erreur avant de lire la réponse !

L'erreur : La condition $i < 5$ arrête la boucle dès que i vaut 5 : le programme n'affiche que 1, 2, 3 et 4, jamais 5. Pour afficher les nombres jusqu'à 5 inclus, il fallait écrire $i \leq 5$ comme condition d'arrêt. Réflexe : avant de valider une condition, vérifie toujours si la dernière valeur voulue doit être incluse ou exclue de la boucle.

10 Exercices type Bac *corrections en fin de fiche ✂*

Exercice 1. Écris une structure conditionnelle qui affiche "Majeur" si $\text{age} \geq 18$, et "Mineur" sinon.

Exercice 2. Écris une boucle while qui affiche les nombres de 1 à 4, avec une variable compteur initialisée à 1.

11 QCM — 10 questions

⌚ 5 MIN MAX • OBJECTIF 8/10

Une seule bonne réponse par question, sans calculatrice — entraînement épreuve d'automatismes du Bac 2026.

1. Quel est le type de la valeur 3.0 en Python ?

- ☐ A int
- ☐ B str
- ☐ C float

2. Quelle expression renvoie un booléen (True ou False) ?

- ☐ A $5 + 3$
- ☐ B $5 == 3$
- ☐ C "5"

3. Que fait l'instruction `if age >= 18` : afficher "Majeur" ?

- ☐ A Elle affiche "Majeur" uniquement si `age` est supérieur ou égal à 18
- ☐ B Elle affiche "Majeur" à chaque exécution du programme, quelle que soit la valeur de `age`
- ☐ C Elle modifie la valeur de la variable `age`

4. Pour parcourir les 5 éléments d'une liste, nombre de répétitions connu à l'avance, quelle structure choisir ?

- ☐ A une boucle `while`
- ☐ B une structure conditionnelle `if`
- ☐ C une boucle `for`

5. Pourquoi la boucle `while` `compteur < 10` : afficher compteur (sans jamais modifier compteur) ne s'arrête-t-elle jamais ?

- ☐ A Parce que la condition `compteur < 10` ne peut jamais devenir fausse, faute d'instruction qui modifie `compteur`
- ☐ B Parce qu'afficher une valeur est interdit dans une boucle `while`
- ☐ C Parce qu'il manque un `elif` après le `while`

6. Que vaut `i` à la sortie de : `i = 1`, puis `while i < 5 : i = i + 1` ?

- ☐ A 4
- ☐ B 5
- ☐ C 6

7. Dans un `if / elif / else` à plusieurs cas, quel ordre faut-il respecter pour les conditions `elif` ?

- ☐ A Aucun ordre particulier, Python les teste toutes
- ☐ B Du cas le plus strict au moins strict, car Python s'arrête au premier vrai
- ☐ C Du cas le moins strict au plus strict

8. Quelle est la différence entre une boucle `for` et une boucle `while` ?

- ☐ A `for` répète un nombre de fois connu à l'avance (bornée), `while` répète tant qu'une condition reste vraie (non bornée)
- ☐ B `for` et `while` sont deux noms différents pour exactement la même instruction
- ☐ C `for` ne peut être utilisée qu'avec des nombres, `while` qu'avec des chaînes de caractères

9. Quel type de donnée Python attribue-t-il à la valeur "18" (avec les guillemets) ?

- ☐ A `int`
- ☐ B `bool`
- ☐ C `str`

10. Un programme parcourt la liste `notes = [5, 11, 9, 14]` avec une boucle `for` et compte, avec un compteur initialisé à 0, les notes strictement supérieures à 10. Combien vaut compteur à la fin ?

- ☐ A 1
- ☐ B 2
- ☐ C 3



12

En 30 secondes — ta carte de dernière révision

découpe-la et garde-la dans ta trousse — révision de veille de contrôle !

- Quatre types de base : entier (`int`), flottant (`float`), chaîne de caractères (`str`), booléen (`bool`), chacun avec ses propres opérations.
- `if / elif / else` exécute un bloc selon une condition ; les cas s'écrivent du plus strict au moins strict.
- Boucle `for` = nombre de répétitions connu à l'avance ; boucle `while` = tant qu'une condition reste vraie, avec une condition d'arrêt qui doit évoluer.

PLAN DE RÉVISION ESPACÉE — relis la carte et refais le QCM : ☐ J+1 ☐ J+3 ☐ J+7 ☐ J+14



VA PLUS LOIN — TESTE-TOI SUR CETTE NOTION

📱 Scanne le QR code (ou clique dessus) : le QCM chronométré de cette notion, note immédiate, chaque erreur expliquée. Gratuit, sans inscription.

🎯 Pour t'entraîner — QCM en nombre, évaluations, explications détaillées, analyse de tes points faibles — l'abonnement Réussito (sans engagement) : via le QR code ou sur reussito.fr/abo

📖 Débloque toutes les fiches de 1ère : via le QR code ou sur reussito.fr/premiere



L'élève travaille sans les solutions sous les yeux ; vous corrigez ensemble avec ces pages. Chaque réponse est justifiée : pas besoin de connaître le cours.

Correction des exercices

Exercice 1. La structure conditionnelle correcte est : `if age >= 18`, afficher "Majeur" ; sinon (`else`), afficher "Mineur".

Exercice 2. La boucle correcte est : `compteur = 1`, puis `while compteur <= 4` : afficher compteur, puis `compteur = compteur + 1`. La condition `compteur <= 4`, et non `compteur < 4`, est nécessaire pour inclure la valeur 4.

Réponses du QCM — expliquées une par une

- 1 → C** 3.0 contient un point décimal : c'est un flottant (`float`), même si sa partie décimale est nulle. *Piège : `int` correspondrait à 3 sans point décimal, une valeur différente d'écriture ; `str` correspondrait à une valeur entre guillemets comme "3.0", pas au nombre 3.0 lui-même.*
- 2 → B** `==` est un opérateur de comparaison : il compare deux valeurs et renvoie toujours un booléen, ici `False` car 5 est différent de 3. *Piège : `5 + 3` est une addition qui renvoie un entier (8), pas un booléen ; "5" est une chaîne de caractères, pas une comparaison.*
- 3 → A** Une structure conditionnelle `if` n'exécute son bloc que si la condition testée, ici `age >= 18`, est vraie. *Piège : afficher "Majeur" à chaque exécution ignorerait la condition, ce qui contredit le principe même du `if` ; `if` ne modifie jamais la valeur d'une variable, il teste seulement une condition.*
- 4 → C** Le nombre de répétitions est connu avant le début de la boucle (5 éléments) : c'est le cas d'usage de la boucle `for`, boucle bornée. *Piège : la boucle `while` convient quand le nombre de répétitions n'est pas connu à l'avance, ce n'est pas le cas ici ; `if` n'est pas une boucle, elle ne répète rien.*
- 5 → A** Sans instruction qui modifie `compteur` à l'intérieur de la boucle, sa valeur ne change jamais : la condition reste vraie indéfiniment, et la boucle tourne sans fin. *Piège : afficher une valeur est une instruction valide à l'intérieur d'une boucle `while`, elle n'est pas la cause du problème ; `elif` appartient aux structures conditionnelles, il n'a aucun rapport avec une boucle `while`.*
- 6 → B** La boucle s'arrête dès que `i < 5` devient faux, c'est-à-dire dès que `i` atteint 5 : la dernière valeur de `i` à la sortie est donc 5. *Piège : 4 est la dernière valeur pour laquelle la condition était encore vraie, mais la boucle exécute encore un tour avant de ressortir ; 6 supposerait un tour de boucle supplémentaire qui ne se produit pas, car la condition est testée avant chaque tour.*
- 7 → B** Python teste les conditions `if` / `elif` dans l'ordre d'écriture et s'arrête à la première vraie : il faut donc écrire la condition la plus stricte en premier. *Piège : Python ne teste pas systématiquement toutes les conditions, il s'arrête à la première vraie ; commencer par le moins strict ferait que ce cas s'appliquerait toujours en premier, masquant les cas plus stricts.*
- 8 → A** La boucle `for` est bornée : le nombre de répétitions est fixé avant le début. La boucle `while` est non bornée : elle dépend d'une condition évaluée à chaque tour. *Piège : `for` et `while` ont des comportements différents, ce ne sont pas deux noms pour la même instruction ; les deux peuvent être utilisées avec tout type de donnée, cette restriction n'existe pas.*

9 → C

Les guillemets font de "18" une chaîne de caractères (str), même si son contenu ressemble à un nombre : ce n'est pas la valeur numérique 18. *Piège : int correspondrait à 18 sans guillemets, une valeur numérique différente ; bool ne prend que les valeurs True ou False, jamais un texte.*

10 → B

Seules 11 et 14 sont strictement supérieures à 10 : le compteur passe de 0 à 1 pour 11, reste à 1 pour 9, puis passe à 2 pour 14. *Piège : 1 correspondrait à ne compter qu'une seule des deux notes valides, en oubliant 11 ou 14 ; 3 supposerait qu'une note supplémentaire, comme 9, soit comptée à tort alors qu'elle n'est pas strictement supérieure à 10.*