



TECHNOLOGIE · 3E

✓ EXIGIBLE AU BREVET

PRIORITÉ P1

Programmation d'un objet technique

booléens, listes et programmation textuelle

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

- ✓ Utiliser le type booléen et les opérateurs logiques ET, OU, NON dans une condition.
- ✓ Manipuler une liste : lire une valeur à sa position, parcourir tous ses éléments.
- ✓ Découper un programme en sous-programmes (fonctions) pour le rendre plus clair : la modularité.
- ✓ Passer de la programmation par blocs à la programmation textuelle pour écrire un programme simple.

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

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

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

- Faites lire « Les mots à connaître » : un booléen ne vaut que vrai ou faux, une liste range plusieurs valeurs à des positions numérotées à partir de 0.
- Demandez à votre enfant d'expliquer avec ses mots la différence entre ET (les deux conditions vraies) et OU (une seule suffit).
- Corrigez le QCM avec les pages détachables. Moins de 8/10 : refaire la fiche dans 3 jours.

1 Les mots à connaître *surigne ceux que tu confonds !*

Booléen — un type de donnée qui ne peut prendre que deux valeurs : vrai (True) ou faux (False).

Opérateur logique — un mot-clé qui combine ou transforme des valeurs booléennes. ET n'est vrai que si les deux conditions sont vraies. OU est vrai dès qu'une des deux conditions est vraie. NON inverse une valeur vraie en fausse, et inversement.

Liste — une structure de données qui range plusieurs valeurs sous un seul nom, dans un ordre précis. Chaque valeur est repérée par sa position, la première valeur étant à la position 0.

Variable — un espace de mémoire, identifié par un nom, qui contient une valeur pouvant changer pendant l'exécution d'un programme.

Condition — une structure de contrôle qui exécute des instructions différentes selon qu'une expression booléenne est vraie ou fausse (« si... alors... sinon »).

Boucle — une structure de contrôle qui répète une ou plusieurs instructions, par exemple pour parcourir une par une toutes les valeurs d'une liste.

Sous-programme (fonction) — une partie du programme, regroupée sous un nom, qui réalise une tâche précise. On l'appelle aussi fonction : elle peut être appelée (réutilisée) plusieurs fois sans réécrire son code. Une fonction peut retourner une valeur : c'est le résultat qu'elle renvoie à la fin de son exécution. Par exemple, une fonction peut retourner vrai ou faux, si elle teste une condition.

Modularité — le fait de découper un programme en plusieurs sous-programmes indépendants. Chacun est plus facile à écrire, à tester et à corriger séparément.

Programmation textuelle et programmation par blocs — deux façons d'écrire un programme. La programmation textuelle utilise des lignes de texte, comme en Python. Elle remplace, dès la 3e, la programmation par blocs, qui assemble des blocs graphiques, comme en Scratch et qui était utilisée avant.

Compteur — une variable qui sert à compter. Elle démarre à 0, puis augmente de 1 à chaque fois qu'une condition est vraie.

2 Les règles essentielles *chaque règle avec son exemple*

1 Le type booléen ne peut prendre que deux valeurs : vrai ou faux.

exemple → La variable porte_ouverte vaut vrai si la porte est ouverte, faux si elle est fermée.

2 L'opérateur ET n'est vrai que si les deux conditions qu'il combine sont vraies en même temps. L'opérateur OU est vrai dès qu'au moins une des deux conditions qu'il combine est vraie.

exemple → « luminosite_faible ET presence_detectee » n'est vrai que si les deux sont vraies en même temps. « luminosite_faible OU presence_detectee » est vrai si l'une des deux conditions est vraie, même si l'autre est fausse.

3 L'opérateur NON inverse une valeur booléenne : NON transforme vrai en faux, et faux en vrai.

exemple → Si presence_detectee vaut faux, alors NON presence_detectee vaut vrai. Personne n'est détecté, donc l'absence de présence est vraie.

4 Une liste range plusieurs valeurs sous un seul nom, dans un ordre précis. On accède à chaque valeur par sa position, la première étant à la position 0.

exemple → La liste temperatures contient 3 valeurs : 18, 21 et 19, aux positions 0, 1 et 2.

5 Un sous-programme (ou fonction) regroupe des instructions sous un nom, pour les réutiliser sans les réécrire.

exemple → La fonction allumer_lampadaire() peut être appelée plusieurs fois dans le programme, chaque fois qu'il faut allumer le lampadaire.

- 6 Découper un programme en plusieurs sous-programmes, c'est la modularité. Cela facilite l'écriture, le test et la correction du programme.**

exemple → Séparer `mesurer_luminosite()`, `tester_conditions()` et `allumer_lampadaire()` en 3 sous-programmes distincts est plus clair qu'un seul long bloc de code.

- 7 La programmation textuelle (Python) remplace, en 3e, la programmation par blocs (Scratch) utilisée avant.**

exemple → En Python, on écrit `luminosites = [8, 22, 30, 15, 9]` ; en programmation par blocs, les mêmes valeurs sont assemblées dans un bloc « liste ».

3

Méthode — Évaluer une condition avec ET/OU en parcourant une liste

Exemple suivi : le programme d'un lampadaire connecté qui teste, pour chaque relevé, si la luminosité est faible OU si une présence est détectée.

- Étape 1** Identifier les listes de valeurs à parcourir (ici `luminosites` et `presences`), qui doivent avoir la même longueur.
- Étape 2** Repérer l'opérateur logique de la condition : OU (une seule des deux conditions suffit) ou ET (il faut les deux).
- Étape 3** Pour chaque position de la liste, en partant de la position 0, évaluer chaque test simple puis combiner les deux résultats avec l'opérateur logique repéré à l'étape précédente.
- Étape 4** Si le résultat est vrai, augmenter le compteur de 1 : un résultat vrai compte pour 1, un résultat faux compte pour 0. Sinon, laisser le compteur inchangé et passer à la position suivante.
- Étape 5** Une fois toutes les positions parcourues, la valeur finale du compteur est le résultat du programme.

4

Exemple type Brevet — rédigé pas à pas

Un lampadaire de jardin s'allume automatiquement grâce à sa fonction `allumer(luminosite, presence)`. Elle retourne vrai si la luminosité est inférieure à 10 OU si une présence est détectée. Le programme enregistre 5 relevés, rangés dans deux listes de même longueur : `luminosites` contient 8, 22, 30, 15 et 9 ; `presences` contient faux, faux, vrai, faux et faux (aux mêmes positions, de 0 à 4). a) Combien de fois le lampadaire s'allume-t-il sur ces 5 relevés, avec la condition OU ? b) Si l'on remplace OU par ET dans la fonction (le lampadaire ne s'allume que si les deux conditions sont vraies en même temps), combien de fois s'allume-t-il sur les mêmes relevés ?

Étape 1 — Position 0 : `luminosite` vaut 8, `presence` vaut faux. Avec OU : $(8 < 10)$ est vrai, donc le résultat est vrai même sans la présence. Le lampadaire s'allume, le compteur passe à 1.

Étape 2 — Position 1 : `luminosite` vaut 22, `presence` vaut faux. Les deux tests sont faux, donc OU est faux. Le lampadaire ne s'allume pas, le compteur reste à 1.

Étape 3 — Position 2 : `luminosite` vaut 30, `presence` vaut vrai. $(30 < 10)$ est faux, mais la présence est vraie, donc OU est vrai. Le lampadaire s'allume, le compteur passe à 2.

Étape 4 — Position 3 : luminosité vaut 15, présence vaut faux. Les deux tests sont faux, donc OU est faux. Le compteur reste à 2.

Étape 5 — Position 4 : luminosité vaut 9, présence vaut faux. $(9 < 10)$ est vrai, donc OU est vrai. Le lampadaire s'allume, le compteur passe à 3.

Étape 6 — Avec ET, chaque position exige que la luminosité soit inférieure à 10 ET qu'une présence soit détectée, en même temps. Aucune des 5 positions ne combine les deux : ET est faux à chaque position.

Conclusion : a) Avec OU, le lampadaire s'allume 3 fois, aux positions 0, 2 et 4. b) Avec ET, il ne s'allume jamais (0 fois), car aucune position ne combine à la fois une luminosité inférieure à 10 ET une présence détectée.

5 Les pièges à éviter *les erreurs classiques en contrôle...*

✗ Confondre ET (les deux conditions doivent être vraies) et OU (une seule suffit).

✗ Croire qu'une liste est identique à une variable simple, alors qu'elle range plusieurs valeurs à des positions différentes.

✗ Oublier qu'en programmation textuelle, la première position d'une liste est 0, pas 1.

✗ Croire qu'un sous-programme (fonction) s'exécute automatiquement, alors qu'il doit être appelé explicitement dans le programme.

6 Tips — pour t'en souvenir

ET = les deux conditions vraies ; OU = au moins une vraie ; NON = inverse vrai et faux.

Dans une liste, la première position est toujours 0, pas 1.

Une fonction doit être appelée pour s'exécuter : elle ne se lance jamais toute seule.

Programmation textuelle = lignes de code (Python) ; **programmation par blocs = blocs graphiques assemblés** (Scratch), utilisée avant la 3e.

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

« PROGRAMMATION » au centre, 5 branches colorées : BOOLÉEN (vrai/faux), OPÉRATEURS LOGIQUES (ET, OU, NON), LISTE (valeurs à des positions numérotées), SOUS-PROGRAMME / FONCTION (tâche réutilisable, modularité), PROGRAMMATION TEXTUELLE (lignes de code, comme Python, remplace la programmation par blocs). Redessine-la de mémoire après avoir lu la fiche.



ta carte mentale, de mémoire...

8 À toi de jouer — trouve l'erreur

Un élève écrit : « La fonction `ouverture(badge_valide, code_correct)` retourne vrai si `badge_valide` ET `code_correct` sont vrais. Avec `badge_valide` qui vaut vrai et `code_correct` qui vaut faux, le portail s'ouvre quand même, parce que le badge est valide, ça suffit. »

→ cherche l'erreur avant de lire la réponse !

L'erreur : L'élève confond ET et OU : il traite l'opérateur ET comme s'il suffisait qu'une seule des deux conditions soit vraie. Avec l'opérateur ET, les deux conditions doivent être vraies en même temps pour que le résultat soit vrai. Ici, `badge_valide` est vrai mais `code_correct` est faux, donc le résultat est faux : la fonction retourne faux, et le portail reste fermé. Si la fonction utilisait OU, `badge_valide` vrai aurait suffi. Mais elle utilise ET, qui est plus exigeant.

9 Exercices type Brevet *corrections en fin de fiche ✂*

Exercice 1. Un vélo à assistance électrique connecté enregistre 5 mesures de vitesse, en km/h, rangées dans une liste vitesses : 5, 18, 22, 9 et 25. La fonction `coupe_assistance(vitesse)` retourne vrai si vitesse est inférieure à 6 OU supérieure à 20. Combien de fois l'assistance se coupe-t-elle sur ces 5 mesures ?

Exercice 2. Un portail automatique utilise la fonction `ouverture(badge_valide, code_correct)`, qui retourne vrai si `badge_valide` ET `code_correct` sont vrais tous les deux. Un visiteur présente un badge valide (`badge_valide` vaut vrai) mais tape un code faux (`code_correct` vaut faux). Le portail s'ouvre-t-il ? Justifie ta réponse avec l'opérateur logique utilisé. Que vaut NON `code_correct` dans cette situation ?

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

1. Un booléen est une donnée qui peut valoir :

- ☐ A vrai ou faux, et rien d'autre
- ☐ B n'importe quel nombre entier
- ☐ C une liste de plusieurs valeurs

2. L'opérateur logique ET est vrai quand :

- ☐ A au moins une des deux conditions est vraie
- ☐ B les deux conditions sont vraies en même temps
- ☐ C aucune des deux conditions n'est vraie

3. L'opérateur logique OU est vrai quand :

- ☐ A les deux conditions sont vraies en même temps, jamais une seule
- ☐ B aucune des deux conditions n'est vraie
- ☐ C au moins une des deux conditions est vraie

4. Une liste sert à :

- ☐ A ranger plusieurs valeurs sous un seul nom, dans un ordre précis
- ☐ B stocker une seule valeur qui change au cours du programme
- ☐ C répéter une instruction plusieurs fois

5. Dans la liste luminosites, qui contient 8, 22, 30, 15 et 9 dans cet ordre, quelle est la valeur à la position 2 ?

- ☐ A 22
- ☐ B 9
- ☐ C 30

6. Un sous-programme (fonction) sert à :

- ☐ A regrouper des instructions sous un nom, pour les réutiliser sans les réécrire
- ☐ B stocker plusieurs valeurs dans un ordre précis
- ☐ C transformer un programme textuel en programme par blocs

7. La modularité consiste à :

- ☐ A écrire tout le programme dans un seul bloc de code, sans le découper
- ☐ B découper un programme en plusieurs sous-programmes indépendants
- ☐ C remplacer toutes les variables par des listes

8. L'opérateur logique NON appliqué à une condition vraie donne :

- ☐ A une condition toujours vraie, NON ne change rien
- ☐ B une condition fausse : NON inverse la valeur
- ☐ C une erreur, NON ne peut s'appliquer qu'à deux conditions à la fois

9. luminosites contient 8, 22, 30, 15 et 9 ; presences contient faux, faux, vrai, faux et faux, aux mêmes positions. Le lampadaire s'allume si luminosite est inférieure à 10 OU si presence est vraie. Combien de fois s'allume-t-il sur ces 5 relevés ?

- ☐ A 0
- ☐ B 5
- ☐ C 3

10. Un élève dit : « la fonction ouverture(badge_valide, code_correct) retourne vrai si badge_valide ET code_correct sont vrais. Avec badge_valide vrai et code_correct faux, le portail s'ouvre puisque le badge est valide. » Quelle est son erreur ?

- ☐ A il confond ET (qui exige que les deux conditions soient vraies) avec OU (qui se contente d'une seule)
- ☐ B il a oublié que badge_valide était vrai
- ☐ C il a inversé les deux conditions du portail

**11**

En 30 secondes — ta carte de dernière révision

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

- Booléen = vrai ou faux. ET exige les deux vraies, OU se contente d'une seule, NON inverse.
- Liste = plusieurs valeurs sous un seul nom, à des positions numérotées à partir de 0.
- Fonction = sous-programme réutilisable ; la modularité découpe le programme en plusieurs fonctions.

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



QCM en ligne

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 3e : via le QR code ou sur reussito.fr/3e



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. Position 0, vitesse vaut 5 : $(5 < 6)$ est vrai, donc OU est vrai. L'assistance se coupe, le compteur passe à 1. Position 1, vitesse vaut 18 : les deux tests sont faux, OU est faux, le compteur reste à 1. Position 2, vitesse vaut 22 : $(22 > 20)$ est vrai, donc OU est vrai, le compteur passe à 2. Position 3, vitesse vaut 9 : les deux tests sont faux, le compteur reste à 2. Position 4, vitesse vaut 25 : $(25 > 20)$ est vrai, donc OU est vrai, le compteur passe à 3. L'assistance se coupe 3 fois sur les 5 mesures.

Exercice 2. Non, le portail ne s'ouvre pas. L'opérateur ET exige que les deux conditions soient vraies en même temps. Ici badge_valide est vrai mais code_correct est faux : le résultat est donc faux, la fonction retourne faux et le portail reste fermé. NON code_correct vaut vrai, car NON inverse faux en vrai : ici code_correct est faux, donc NON code_correct est vrai.

Réponses du QCM — expliquées une par une

- 1 → A** Un booléen ne peut prendre que deux valeurs : vrai (True) ou faux (False). *Piège : B confond booléen et type numérique, qui peut prendre une infinité de valeurs ; C confond booléen et liste, qui range plusieurs valeurs.*
- 2 → B** ET n'est vrai que si les deux conditions combinées sont vraies en même temps. *Piège : A décrit l'opérateur OU, pas ET ; C décrit une situation où le résultat est toujours faux, ce qui n'est le cas d'aucun des deux opérateurs.*
- 3 → C** OU est vrai dès qu'au moins une des deux conditions combinées est vraie, même si l'autre est fausse. *Piège : A décrit l'opérateur ET, pas OU ; B décrit une situation où le résultat est toujours faux.*
- 4 → A** Une liste range plusieurs valeurs sous un seul nom, dans un ordre précis, chaque valeur étant repérée par sa position. *Piège : B décrit une variable simple, pas une liste ; C décrit une boucle, pas une liste.*
- 5 → C** La première position d'une liste est 0 : position 0 vaut 8, position 1 vaut 22, position 2 vaut 30. *Piège : A correspond à la position 1, pas 2, à cause d'un décalage d'une position ; B correspond à la dernière position (4), pas à la position 2.*
- 6 → A** Un sous-programme (fonction) regroupe des instructions sous un nom, pour les réutiliser sans les réécrire. *Piège : B décrit une liste, pas une fonction ; C décrit une transformation qui n'existe pas dans le programme.*
- 7 → B** La modularité consiste à découper un programme en plusieurs sous-programmes indépendants, plus faciles à écrire, à tester et à corriger. *Piège : A décrit l'inverse de la modularité ; C confond modularité et remplacement de variables par des listes, deux choses différentes.*
- 8 → B** NON inverse une valeur booléenne : appliqué à une condition vraie, il donne faux ; appliqué à une condition fausse, il donne vrai. *Piège : A ignore le rôle de NON, qui inverse toujours la valeur de la condition ; C invente une contrainte inexistante : contrairement à ET et OU, NON s'applique justement à une seule condition.*
- 9 → C** Le lampadaire s'allume aux positions 0 (luminosité faible), 2 (présence détectée) et 4 (luminosité faible), soit 3 fois. *Piège : A correspondrait au résultat obtenu avec l'opérateur ET sur ces mêmes relevés, pas avec OU ; B compte les 5 relevés comme si le lampadaire s'allumait à chaque fois, ce qui ignore la condition.*
- 10 → A** Il confond ET et OU : avec ET, les deux conditions doivent être vraies en même temps, une seule ne suffit pas. *Piège : B ne correspond pas à l'erreur, badge_valide vrai est bien pris en compte ; C invente une confusion de conditions absente de l'énoncé.*