MATHS 4E FICHE DE COURS

Algorithmique et programmation — cours de maths 4e

Un programme est une suite d'instructions qu'une machine exécute sans rien deviner : elle fait exactement ce qui est écrit, ni plus ni moins. Trois idées suffisent pour tout ce qu'on demande en 4e — une variable qui retient, une condition qui décide, une répétition qui recommence. Et la plus grosse difficulté n'est pas d'écrire le programme : c'est de traduire l'objectif en condition juste.

LE MOT CLÉ	LE GESTE	LA RÈGLE D'OR
La condition : vraie ou fausse	**Suivre la variable, ligne à ligne**	**« Au moins 10 » s'écrit score ≥ 10**

À quoi ça sert : Algorithmique et programmation

La confusion entre « plus de » et « au moins » n'est pas une subtilité d'exercice : c'est l'un des bugs les plus courants du métier, et il coûte cher. Un site qui offre la livraison « à partir de 50 € » et qui teste « total > 50 » refuse la livraison gratuite à qui commande exactement 50 € — et personne ne s'en aperçoit tant qu'aucun client ne tombe pile dessus. Les mêmes trois briques font tourner tout ce qui vous entoure : le distributeur qui compare votre code, la machine à laver qui répète un cycle, l'appareil photo du téléphone qui décide d'allumer le flash. Et le réflexe qu'enseigne cette fiche — essayer la valeur exacte du seuil — est celui que les développeurs appliquent tous les jours sous le nom de « test aux limites ».

Un peu d'histoire : Algorithmique et programmation

Le mot « algorithme » est le nom d'un homme : al-Khwarizmi, savant de Bagdad du IXe siècle, dont le nom latinisé en « Algoritmi » a fini par désigner les méthodes de calcul qu'il décrivait — le même auteur, d'ailleurs, dont le livre a donné le mot « algèbre ». Quant au premier programme de l'histoire, il a été écrit en 1843 par la mathématicienne britannique Ada Lovelace pour une machine qui n'a jamais été construite, la machine analytique de Charles Babbage. Elle y avait vu quelque chose que Babbage lui-même n'avait pas vu : que la machine pourrait manipuler autre chose que des nombres.

Définition : Algorithmique et programmation

Un algorithme est une suite d'instructions qui mène à un résultat ; un programme en est l'écriture dans

un langage qu'une machine comprend. Trois briques suffisent ici. La VARIABLE est une case nommée qui retient une valeur, et cette valeur peut changer. La CONDITION est une affirmation que la machine teste : elle vaut vrai ou faux. L'INSTRUCTION CONDITIONNELLE, « si … alors … sinon », choisit ce qu'on exécute selon la réponse.

⚠️ La machine ne devine rien : elle applique la condition écrite, pas celle qu'on avait en tête.

score part de 0, gagne 4 trois fois, puis on décide

Le programme se lit de haut en bas. Après les trois tours de boucle, score vaut $3 \times 4 = 12$. Comme $12 \geq 10$, la condition est vraie : le lutin dit « Gagné », et la branche « sinon » n'est jamais exécutée.

Propriétés : Algorithmique et programmation

Vraie ou fausse, rien d'autre

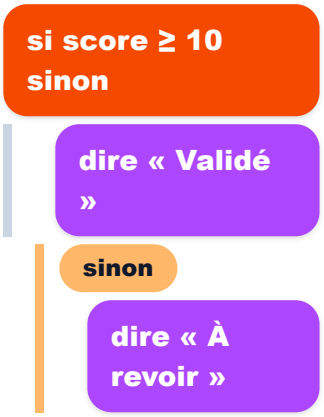
Une condition est un test dont la réponse est oui ou non. ⚠️ Et le cas limite compte : si score vaut exactement 10, alors « score > 10 » est FAUSSE, tandis que « score ≥ 10 » est vraie.

Données	si score vaut	score > 10
	9	faux
	10	faux
	11	vrai

une seule ligne diffère : celle du seuil exact

Une branche, jamais deux

« Si … alors … sinon » choisit. Quand la condition est vraie, seules les instructions du haut s'exécutent ; quand elle est fausse, seules celles du « sinon ». Il n'y a jamais les deux.



La variable garde la dernière valeur

« Mettre score à 0 » écrase ce qu'il y avait. « Ajouter 4 à score » ne met pas 4 : il remplace l'ancienne valeur par l'ancienne plus 4. Trois fois de suite depuis 0, on arrive à 12.

Données	l'instruction
	mettre score à 0
	ajouter 4 à score
	ajouter 4 à score
	ajouter 4 à score

la variable garde la dernière valeur écrite

Modifier, c'est changer une seule

chose

Voici le même programme avec 2 au lieu de 4. Le score final devient $3 \times 2 = 6$, la condition est fausse, et le lutin dit « Continue ». Un nombre a changé, la sortie a basculé.

La formule : Algorithmique et programmation

LES QUATRE BLOCS DE LA 4e

mettre … à … · ajouter … à … · répéter … fois · si … alors … sinon

Quatre blocs suffisent à écrire tous les programmes de cette fiche : un pour créer une variable, un pour la faire évoluer, un pour répéter, un pour décider. Tout le reste est du vocabulaire.

Méthode : Algorithmique et programmation

1. Traduire l'objectif

On part de la phrase et on choisit le symbole. « Plus de 10 » exclut 10, « au moins 10 » l'inclut, « exactement 10 » ne garde que lui. C'est l'étape où se jouent la plupart des erreurs.

Données	l'objectif dit
	plus de 10
	au moins 10
	exactement 10

« au moins » inclut la valeur, « plus de » ne l'inclut pas

2. Suivre la variable

On note la valeur de la variable après chaque instruction, dans un tableau. C'est long la première fois, et c'est la seule façon d'être sûr — un programme ne se lit pas, il se déroule.

Données	l'instruction
	mettre score à 0
	ajouter 4 à score
	ajouter 4 à score
	ajouter 4 à score

la variable garde la dernière valeur écrite

3. Tester aux bornes

On essaie la valeur juste en dessous du seuil, le seuil exact, et celle juste au-dessus. Une erreur de « > » ou « ≥ » ne se voit QUE sur le seuil exact : tester au hasard ne la trouvera pas.

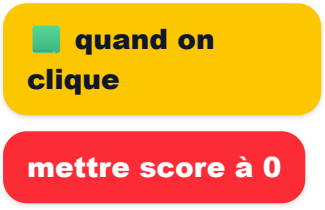
Données	si score vaut	score > 10
	9	faux
	10	faux
	11	vrai

une seule ligne diffère : celle du seuil exact

Selon ce que l'on cherche : Algorithmique et programmation

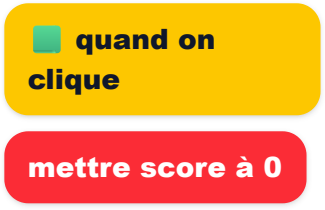
Un jeu qui compte les points

Une variable retient le score, une boucle le fait monter, une condition décide du message final. C'est le squelette de presque tous les petits jeux.



Changer la difficulté

Baisser le bonus de 4 à 2 rend le jeu plus dur sans toucher au reste : le score final passe de 12 à 6, et le message change. Modifier, c'est agir sur un seul nombre — puis vérifier.



Programmer une formule

Une expression littérale se met telle quelle dans un bloc : $y = 3x + 2$ devient « mettre y à 3 × x + 2 ». La lettre du mathématicien et la variable du programmeur sont la même idée.

répéter 3 fois
ajouter 4 à score
si score ≥ 10 sinon
dire « Gagné »
sinon
dire « Continue »

répéter 3 fois
ajouter 2 à score
si score ≥ 10 sinon
dire « Gagné »
sinon
dire « Continue »

mettre y à 3 × x + 2
dire y

Exemples corrigés : Algorithmique et programmation

Que dit le lutin ?

Le programme met score à 0, répète 3 fois « ajouter 4 à score », puis teste « score ≥ 10 ».

Quel message s'affiche ?

On déroule. Après « mettre score à 0 », score vaut 0. La boucle tourne trois fois : 0 → 4 → 8 → 12. La condition « $12 \geq 10$ » est vraie, donc le lutin dit « Gagné ». ⚠️ La branche « sinon » n'est PAS

Le bug du « au moins »

Un programme doit dire « Validé » si la note est AU MOINS 10. Il utilise la condition « note > 10 ».

Où est le problème, et comment le trouver ?

Données	si score vaut	score > 10	score ≥ 10
	9	faux	faux
	10	faux	VRAI
	11	vrai	vrai

une seule ligne diffère : celle du seuil exact

Pour une note de 9, le programme refuse — c'est correct. Pour 11, il valide — correct aussi. Mais pour exactement 10, « $10 > 10$ » est FAUSSE : l'élève est refusé alors qu'il devrait passer. ⭐ Le programme exclut le cas où la note est égale au seuil, et c'est le SEUL essai qui le révèle. La correction est d'écrire « note ≥ 10 ». Et la leçon vaut au-delà : on ne teste pas au hasard, on teste aux bornes.

exécutée : une instruction conditionnelle ne choisit jamais les deux.

Modifier sans tout casser

On veut rendre le jeu plus difficile : le bonus passe de 4 à 2.

Que devient le message affiché ?

Seul le bloc « ajouter … à score » change. La boucle donne maintenant 0 → 2 → 4 → 6, donc score vaut 6. La condition « $6 \geq 10$ » est fausse, et le lutin dit « Continue ». ⭐ On a modifié UNE valeur, et on a vérifié en déroulant : c'est la méthode complète. Modifier sans dérouler ensuite, c'est espérer plutôt que savoir.

Pièges à éviter : Algorithmique et programmation

Confondre « > » et « ≥ » : « au moins 10 » s'écrit score ≥ 10. Avec score > 10, un élève qui a exactement 10 est refusé — et le bug ne se voit sur aucun essai sauf celui-là.

Croire qu'un « si … alors … sinon » exécute les deux branches : il n'en exécute JAMAIS qu'une seule. Si la condition est fausse, seules les instructions du « sinon » tournent.

Oublier que « ajouter 4 à score » REMPLACE la valeur : score ne vaut pas 4, il vaut son ancienne valeur plus 4. Trois fois de suite à partir de 0, cela fait 12 et non 4.

À retenir : Algorithmique et programmation

Une condition est une affirmation que le programme teste : elle est VRAIE ou FAUSSE, jamais autre chose. « si … alors » n'exécute ses instructions que si elle est vraie.

Une variable est une case qui garde une valeur, et cette valeur peut changer. « mettre à » écrase, « ajouter à » modifie à partir de l'ancienne valeur.

Un programme se vérifie en le faisant tourner sur des valeurs choisies — surtout celles qui tombent exactement sur le seuil, là où les erreurs se cachent.

Exercices corrigés : Algorithmique et programmation

1. La variable score vaut 10. La condition « score > 10 » est-elle vraie ?

Correction :

Non. Le symbole « > » signifie STRICTEMENT supérieur : 10 n'est pas strictement supérieur à 10. Pour inclure le cas d'égalité, il faut écrire « score ≥ 10 ».

2. Dans un bloc « si … alors … sinon », que se passe-t-il quand la condition est fausse ?

Correction :

Seules les instructions du « sinon » sont exécutées. Celles du « alors » sont ignorées entièrement. Une instruction conditionnelle choisit toujours une branche et une seule.

3. score vaut 0. On exécute trois fois « ajouter 4 à score ». Quelle est la valeur finale ?

Correction :

12. Chaque exécution remplace l'ancienne valeur par l'ancienne plus 4 : 0 → 4 → 8 → 12. ⚠️ La réponse 4 confond « ajouter 4 » avec « mettre à 4 ».

4. On veut afficher « Gagné » si le score est au moins 10. Quelle condition choisir ?

Correction :

« score ≥ 10 ». « Au moins 10 » inclut la valeur 10 ; « score > 10 » l'exclurait et refuserait un joueur qui

a exactement 10 points.

5. Un programme teste « score > 5 ». On veut maintenant qu'il réagisse seulement si le score dépasse 8. Quelle est la nouvelle condition ?

Correction :

« score > 8 ». On ne change que le nombre du seuil : le symbole reste « > », puisque « dépasse » veut bien dire strictement supérieur. Puis on vérifie en essayant 8, qui doit être refusé.

6. Que fait le bloc « mettre score à 0 » ?

Correction :

Il donne la valeur 0 à la variable score, en effaçant ce qu'elle contenait. ⚠️ Ce n'est pas « ajouter 0 » : « mettre à » écrase, « ajouter à » part de l'ancienne valeur.