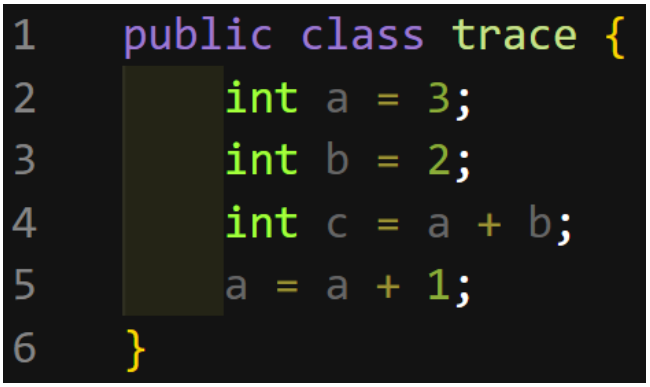
Fiche - Traces d'exécution

Qu'est-ce qu'une trace d'exécution ?

C'est une méthode pour **suivre pas à pas** ce que fait un programme ou un algorithme.
On note **les valeurs des variables après chaque ligne exécutée** → cela aide à comprendre et à trouver les erreurs.

Exemple simple



```
1   public class trace {
2       int a = 3;
3       int b = 2;
4       int c = a + b;
5       a = a + 1;
6   }
```

	Variables		
# Ligne exécutée	**a**	**b**	**c**
2	3	-	-
3	3	2	-
4	3	2	5
5	4	2	5

Fiche - Traces d'exécution

Erreurs fréquentes

Forme et rigueur de la trace

- Oublier une ligne pour chaque instruction exécutée.
- Remplir la trace avec les instructions au lieu des valeurs des variables.
- Sauter directement au résultat final sans détailler les étapes.
- Mal organiser le tableau (valeurs placées dans la mauvaise colonne).

Variables et affectations

- Oublier de mettre à jour la valeur d'une variable après une affectation (x = x + 1;).
- Confondre déclaration et initialisation (int a; int a = 0;).

Conditions (if / else / switch)

- Croire que toutes les branches s'exécutent au lieu d'une seule.
- Mal évaluer une condition (x == 3 vs x = 3).
- Oublier que la condition doit donner un booléen (true ou false).

Boucles (for, while, do...while)

- Ne pas mettre à jour la variable de boucle (boucle infinie).
- Confondre pré-incrément (++i) et post-incrément (i++).

Méthodes et portée

- Oublier qu'une variable locale disparaît en fin de méthode.
- Ne pas tracer la valeur de retour d'une méthode (int y = f(x);).
- Omettre l'effet des paramètres passés

Astuces

- Utilise un **tableau clair** avec une colonne par variable.
- Indique toujours la **valeur après exécution de la ligne**.
- Combine cette méthode avec le **débogueur** ou des ***System.out.println()*** pour vérifier.