

Analyse :

Algorithmique :

- Lectures correctes.
- Boucle `for (i = 0; i < n; i++)` : Correcte.
- `scanf("%d", &x)` dans la boucle : Correct.
- Logique : `if (a > x)` (Absent, `sum` non incrémenté ? Ah si `sum = sum + 1`).
- `else if (x >= a)` (Présent, `sum = sum` inutile).
- Utilise `sum` pour compter les absents. Pas de compteur pour présents affiché distinctement.
- Tests d'arrêt dans la boucle : `if (i == n || n == s)` -> Condition `n == s` est constante et ne dépend pas de l'avancement. Devait être `sum == s`.
- Pas de `break` explicite demandé mais condition d'arrêt mal formulée (affiche juste "stop in" sans arrêter).

Notation :

Critère	Points	Commentaire
Lecture N, A, S	3 / 3	Correct.
Initialisation	3 / 3	<code>sum</code> initialisé.
Condition boucle	2 / 4	Boucle sur N ok, mais l'arrêt sur seuil n'est pas effectif (juste un print).
Logique prés./abs.	4 / 4	Correcte.
Compteurs	2 / 3	Compte les absents (<code>sum</code>), mais oublie de gérer/afficher les présents distinctement.
Affichages inter.	1 / 2	Affiche le cumul d'absents.
Affichage final	1 / 1	Logique <code>a > s</code> ? Non, devrait comparer <code>sum</code> et <code>s</code> . Confus.

NOTE FINALE : 16 / 20

Feedback :

- **Points forts** : Code fonctionnel sur la partie itérative de base.
- **Points faibles** : Confusion entre les variables (`a > s` à la fin au lieu de `sum > s`). L'arrêt sur seuil n'arrête pas vraiment la boucle (`break` interdit, il fallait mettre la condition dans le `for`).

- **Appréciation globale : Bon.** L'étudiant a compris le principe général.
-