

Input reading: 3pts | Initialization: 3 pts | Loop condition: 4 pts | Counters logic: 4 pts | Stop conditions: 3 | Final output: 3

```

#include <stdio.h>
int main() {
    int n, a, s, i = 0, x; sum = 0;
    printf("enter the total number registered students :")
    scanf("%d", &n);
    printf("enter the minimum attendance required :")
    scanf("%d", &a);
    printf("enter the absence three sds")
    scanf("%d", &s);
    for(i = 0; i <= n; i++) {
        scanf("%d", &x);
        if(a > n) {
            sum = sum;
            printf("absent students : d%", sum);
        }
        if else (x >= a) {
            sum += sum;
            printf("present students : d%", );
        }
        if else (n == x || n == s) {
            printf("stop n");
        }
        if else (a > s) {
            printf("session cancelled");
        }
        else (s > a) {
            printf("session valid");
        }
    }
    return 0;
}

```

## Copy 9

---

```
#include <stdio.h>
int main()
{
    int n, a, s, i = 0, x, sum = 0;
    printf("enter the total number registered students");
    scanf("%d", &n);
    printf("enter the minimum attendance required");
    scanf("%d", &a);
    printf("enter the absence threshold");
    scanf("%d", &s);
    for (i = 0; i <n; i++)
    {
        scanf("%d", &x);
        if (a> x)
        {
            sum = sum + 1;
            printf("absent students: %d", sum);
        }

        else if (x>= a)
        {
            sum = sum;
            printf("present students: %d", sum);
        }

        if (i == n || n == s)
        {
            printf("stop in");
        }

        if (a> s)
        {
            printf("Session Cancelled");
        }

        else if (s> a)
        {
            printf("Session Valid");
        }

    }

    return 0;
}
```

## 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 &gt; 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.
-