Notice

Dans tous les sujets :

- Les limites sont disponibles sur la page "Overview" de la plateforme.
- Vous pouvez télécharger des archives sur la plateforme, qui contiennent les évaluateurs d'exemple, les implémentations d'exemple, les tests d'exemple, et les scripts de compilation et exécution.
- Vous pouvez faire jusqu'à 50 soumissions par sujet, et vous devez fournir exactement un fichier par soumission.
- En testant vos programmes avec l'évaluateur d'exemple, l'entrée fournie doit correspondre à la description faite dans chaque sujet, autrement le comportement n'est pas spécifié.
- Dans les entrées d'un évaluateur d'exemple, deux tokens consécutifs sur la même ligne doivent être séparés par une unique espace, sauf si spécifié autrement.
- Lorsque vous testez votre code localement sur votre ordinateur, nous vous recommandons d'utiliser les scripts fournis dans les archives. Notez que nous utilisons l'option `-std=gnu++17` lors de la compilation.
- Si vous ne pouvez pas soumettre sur CMS, vous pouvez à la place utiliser l'outil `ioisubmit` pour sauvegarder votre code afin qu'il soit exécuté à la fin de l'épreuve.
    - Exécutez `ioisubmit <task_shortname> <source_file>` depuis le dossier contenant `<source_file>`.
    - Demandez à un membre du comité de prendre une photo de la sortie de `ioisubmit`. Votre soumission ne sera pas considérée sans que cette étape soit effectuée.
        - Si vous participez en ligne, demandez à votre surveillant de prendre une photo de la sortie de `ioisubmit` et de l'envoyer aux organisateurs.

Conventions

Les énoncés des sujets spécifient les signatures des fonctions en utilisant des types génériques `void`, `int`, `int64`, `int[]` (tableau) et `union(bool,int[])`.

En C++, les évaluateurs utilisent des types ou implémentations suivants :

`void`	`bool`	`int`	`int[]`
`void`	`bool`	`int`	`std::vector<int>`

union(bool, int[])	longueur du tableau a
std::variant<bool, std::vector<int>>	a.size()

En C++, `std::variant` est défini dans le header `<variant>`. Une méthode avec un type de retour `std::variant<bool, std::vector<int>>` peut renvoyer soit un `bool`, soit un `std::vector<int>`. Le code d'exemple suivant montre trois exemples de fonctions qui renvoient `std::variant`.

```
std::variant<bool, std::vector<int>> foo(int N) {
  return N % 2 == 0;
}
std::variant<bool, std::vector<int>> goo(int N) {
  return std::vector<int>(N, 0);
}
std::variant<bool, std::vector<int>> hoo(int N) {
  if (N % 2 == 0) {
    return false;
  }
  return std::vector<int>(N, 0);
}
```