Allgemeine Informationen

- Die Limits sind auf der "Overview"-Seite im Contest-System zu finden.
- Du kannst im Contest-System ein Attachment-Paket herunterladen. Es enthält Sample-Grader, Beispielimplementationen, Beispieltestfälle, sowie Compile- und Run-Scripts.
- Du darfst bei jeder Aufgabe bis zu 50 Mal einsenden, und du musst bei jeder Einsendung genau eine Datei einschicken.
- Wenn du mit dem Sample-Grader testest, muss deine Eingabe dem Format und den Beschränkungen (Constraints) aus der Aufgabenstellung entsprechen; sonst kann unspezifiertes Verhalten auftreten.
- In den Eingaben für den Sample-Grader sind aufeinanderfolgende Tokens in einer Zeile von einzelnen Leerzeichen getrennt, es sei denn, ein anderes Format ist explizit angegeben.
- Um auf deiner lokalen Maschine zu testen, empfehlen wir die Scripts in den Attachment-Paketen. Bitte beachte, dass wir die Compileroption `-std=gnu++17` verwenden.
- Wenn du dein Programm nicht im CMS einschicken kannst, kannst du das Tool `ioisubmit` benutzen, um dein Programm abzuspeichern, damit es nach dem Wettbewerb ausgewertet werden kann.
    - Führe `ioisubmit <task_kurzname> <source_file>` in dem Verzeichnis mit `<source_file>` aus.
    - Bitte ein Kommitteemitglied, ein Foto von der Ausgabe von `ioisubmit` zu machen. Die Einsendung wird nicht beachtet, wenn dieser Schritt nicht gemacht wurde.
        - Falls du online teilnimmst, bitte deine Aufsichtsperson, ein Foto von der Ausgabe von `ioisubmit` zu machen und an die Organisatoren zu schicken.

Konventionen

Die Aufgabenstellungen geben die Funktionssignaturen mit sprachunabhängigen Typnamen `void`, `bool`, `int`, `int[]` (Array), und `union(bool, int[])`. In C++ verwenden die Grader passende Datentypen oder Implementationen, wie in folgender Tabelle angegeben:

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

union(bool, int[])	**Länge des Arrays a**
`std::variant<bool, std::vector<int>>`	`a.size()`

In C++ ist `std::variant` im Header `<variant>` definiert. Eine Funktion mit dem Rückgabetyp `std::variant<bool, std::vector<int>>` kann entweder ein `bool` oder ein `std::vector<int>` zurückgeben. Das folgende Beispiel enthält drei funktionierende Funktionen mit Rückgabetyp `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);
}
```