Poznámky

Pro všechny úlohy:

- Omezení lze nalézt na stránce Přehled (Overview) v CMS.
- Ke každé z úloh je ze soutěžního systému možné stáhnout archiv obsahující ukázkový grader, kostru řešení, ukázkové vstupy a kompilační a spouštěcí skripty.
- Každou úlohu můžete odevzdat nejvýše 50krát, v rámci každého odevzdání musíte zaslat právě jeden soubor.
- Při testování pomocí ukázkových graderů váš vstup musí přesně odpovídat formátu a podmínkám popsaným v zadání, jinak se může stát cokoliv.
- Není-li v zadání řečeno jinak, ve vstupech ukázkových graderů jsou po sobě následující položky na každé řádce vstupu oddělené mezerou.
- Pokud testujete váš program lokálně na svém počítači, doporučujeme použít skripty dostupné v archivech v CMS. Upozorňujeme, že používáme option `-std=gnu++17`.
- Pokud se vám nedaří odevzdat v CMS, můžete pomocí příkazu `ioisubmit` odeslat váš program k vyhodnocení po skončení soutěže.
  - Spusťte `ioisubmit <označení_úlohy> <soubor_s_řešením>` v adresáři, v níž se `<soubor_s_řešením>` nachází.
  - Zavolejte organizátory, aby zajistili, ať pověřená osoba vyfotí výstup příkazu `ioisubmit`. Pokud toto neuděláte, odevzdaný soubor nebude brán v potaz.
    - Soutěžíte-li online, poproste vás dozor, ať vyfotí výstup `ioisubmit` a foto odešle organizátorům.

Konvence

V zadáních jsou funkce popsány pomocí obecných typů `void`, `bool`, `int`, `int[]` (pole) a `union(bool, int[])`.

Gradery v C++ používají nasledující konkrétní typy či implementace

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

`union(bool, int[])`	**délka pole `a`**
`std::variant<bool, std::vector<int>>`	`a.size()`

V C++ je `std::variant` definovaná v hlavičkovém souboru `<variant>`. Metoda s návratovou hodnotou `std::variant<bool, std::vector<int>>` může vrátit jak `bool`, tak `std::vector<int>`. Níže jsou tři příklady funkcí s návratovou hodnotou typu `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);
}
```