Notice (nieľad)

Pre každú úlohu:

- Na webe testovača na stránke "Overview" sú časové a pamäťové limity.
- Z webu testovača si môžeš stiahnuť balíček, v ktorom je ukážkový grader, ukážkové implementácie, príklady vstupu zo zadania a skripty na kompiláciu a spúšťanie.
- Ku každej úlohe môžeš spraviť nanajvýš 50 submitov. Zakaždým odovzdávaš jeden súbor.
- Pri práci s ukážkovým graderom dodržuj formát vstupu zo zadania. Ak tak nespravíš, môže ukážkový grader robiť, čo len chce.
- Ak nie je povedané ináč, vo vstupoch pre ukážkový grader sú údaje oddeľované jednou medzerou.
- Pri lokálnom testovaní riešení odporúčame použiť kompilačný skript v balíčku. Všimni si, že používame prepínač kompilátora -std=gnu++17.
- Ak nevieš odovzdávať do CMS, môžeš použiť nástroj ioisubmit , ktorý lokálne uloží tvoj submit tak, aby ho po súťaži mohli organizátori otestovať.
    - Spusti `ioisubmit <task_shortname> <source_file>` v adresári obsahujúcom odovzdávaný `<source_file>`.
    - Popros dozor, aby odfotil výstup ioisubmit a poslal ho organizátorom. Tento krok je nutný, bez neho tvoj submit nebude otestovaný. (Ak si online, tvoj dozor potom musí poslať fotku organizátorom.)

Typy premenných

V zadaniach sa z historických dôvodov používajú generické typy a operácie s nimi: `void`, `bool`, `int`, `int[]` (pole) a `union(bool, int[])` (premenná, ktorá môže byť buď boolean alebo pole).

V C++ ide v skutočnosti o nasledovné typy a operácie:

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

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

V C++ je `std::variant` definovaný v hlavičkovom súbore `<variant>`. Funkcia s návratovým typom `std::variant<bool, std::vector<int>>` môže vrátiť buď `bool` alebo

`std::vector<int>`. Nižšie uvedený ukážkový kód ukazuje tri príklady funkcií, ktoré vracajú nejaký `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);
}
```