Tudnivalók

Minden feladat esetén:

- A korlátok a CMS "Overview" oldalán találhatók.
- A CMS-ből letölthető egy csomag, ami a mintaértékelőt, egy mintamegvalósítást, példateszteket, valamint fordító és futtató szkripteket tartalmaz.
- Feladatonként legfeljebb 50-szer küldhetsz be, minden esetben egy fájlt.
- A mintaértékelővel való teszteléskor a bemeneted formátuma a leírásnak megfelelő legyen, ellenkező esetben bármi előfordulhat.
- A mintaértékelő bemenetein két egymást követő értéket egyetlen szóköz válassza el, hacsak más formátumot nem ír elő a feladat!
- Lokális teszteléskor javasoljuk, hogy a csomagban adott szkripteket használd. Megjegyezzük, hogy fordításnál a `-std=gnu++17` kapcsolót használjuk.
- Ha nem tudod a CMS-be beküldeni a feladatodat, akor használda az `ioisubmit` parancsot, amely eltárolja a kódodat a verseny utáni végrehajtásra.
    - Futtasd a `ioisubmit <task_shortname> <source_file>` parancsot a `<source_file>` könyvtárban.
    - Kérj meg egy bizottsági tagot, hogy készítsen fotót az `ioisubmit` kimenetéről. Csak akkor értékeljük az így elmentett megoldásokat, ha ezt megtetted.
        - Ha online versenyzel, kérd meg a proktorodat, hogy ő készítse el a fotót az `ioisubmit` kimenetéről, és küldje el a szervezőknek.

Jelölések

A feladatleírásban általános típusnevek szerepelnek: `void`, `bool`, `int`, `int[]` (tömb), és `union(bool, int[])`.

C++-ban az értékelő az alábbi megfeleltetés szerinti típusokat használja:

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

union(bool, int[])	**az a tömb mérete**
`std::variant<bool, std::vector<int>>`	`a.size()`

C++-ban az `std::variant` a `<variant>` header-ben van definiálva. Egy `std::variant<bool, std::vector<int>>` típussal visszatérő függvény vagy `bool` értéket, vagy `std::vector<int>` értéket adhat vissza. Az alábbi példakód három működő példát mutat, amelyben a visszatérési típus `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);
}
```