Nota

Para todas as tarefas:

- Os limites estão disponíveis na página "Overview" no sistema de competição.
- Há um pacote anexo que você pode baixar do sistema de competição, contendo corretores exemplo, exemplos de implementações, exemplos de casos de teste e scripts de compilação e execução.
- Você pode fazer até 50 submissões para cada tarefa e você deve submeter exatamente um arquivo em cada submissão.
- Ao testar seus programas com o corretor exemplo, sua entrada deve corresponder ao formato e restrições definidas no enunciado da tarefa, caso contrário, podem ocorrer comportamentos indefinidos.
- Nas entradas do corretor exemplo, cada dois tokens consecutivos em uma linha são separados por um único espaço em branco, a menos que outro formato seja especificado explicitamente.
- Ao testar seu código na sua máquina local, recomendamos que use os scripts fornecidos no pacote anexo. Observe que utilizamos a opção `-std=gnu++17` para compilar.
- Se você não conseguir submeter usando o CMS, você pode usar a ferramenta `ioisubmit` para armazenar seu código para ser avaliado após a competição.
    - Execute `ioisubmit <task_shortname> <source_file>` no diretório com `<source_file>`.
    - Peça a um membro do comitê para tirar uma foto da saída do `ioisubmit`. Sua submissão não será considerada a menos que este passo tenha sido realizado.
        - Se você estiver competindo on-line, peça a seu proctor para tirar uma fotografia da saída do `ioisubmit` e enviá-la aos organizadores.

Convenção

Os enunciados das tarefas especificam assinaturas usando nomes genéricos de tipos `void`, `bool`, `int`, `int[]` (vetor), e `union(bool, int[])`.

Em C++, os corretores usam tipos de dados ou implementações apropriadas, como listado abaixo

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

union(bool, int[])	tamanho do vetor a
std::variant<bool, std::vector<int>>	a.size()

Em C++, `std::variant` é definido no cabeçalho `<variant>`. Um método com o tipo de retorno `std::variant<bool, std::vector<int>>` pode retornar ou um `bool` ou um `std::vector<int>`. O exemplo de código abaixo mostra três exemplos corretos de funções que retornam `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);
}
```