Nota de Implementación

Si no puedes subir una solución a través del CMS, por favor haz lo siguiente:

- Pon los archivos de tu envío en una carpeta llamada `submit_<task name>` en el escritorio (la carpeta ya debería existir) **antes de que termine la competencia**.
- Para problemas de "solo-salida", tus archivos deben llamarse `01.out`, `02.out`, ..., `10.out`.
- Para otro tipo de problemas, debe haber **exactamente un** archivo en esa carpeta, que contenga el código fuente.
- Solicita al líder de tu equipo que envíe una apelación.

Si crees que deberías recibir tiempo adicional, por favor haz lo siguiente:

- Envía una solicitud de aclaración (a través de CMS o en papel) tan pronto como sea posible.
- No abandones tu escritorio ni hables con otros competidores cuando termine la competencia.

Cada tarea tiene un paquete adjunto que puedes descargar del sistema.

Hay algunas tareas de "solo-salida", para las cuales:

- El paquete adjunto contiene casos de prueba de entrada y casos de prueba de ejemplo. Cada caso de prueba es una subtarea separada.
- Puedes enviar múltiples archivos de salida como un solo archivo zip. Para este propósito, tus archivos de salida deben llamarse `??.out`, donde `??` es el número del caso de prueba (Por ejemplo, `03.out`). Puedes comprimir múltiples archivos usando el siguiente comando: `zip output.zip *.out`
- Puedes realizar a lo más 100 envíos para tareas de solo-salida. En cada uno, puedes enviar los archivos de salida para cualquier subconjunto de los casos de prueba.

Para otras tareas:

- El paquete adjunto contiene evaluadores de ejemplo, implementaciones de ejemplo, casos de prueba de ejemplo, y scripts de compilación.
- Puedes realizar a lo más 50 envíos para cada tarea, y en cada uno debes enviar exactamente un archivo.
- El nombre del archivo que debes enviar es dado en la cabecera del enunciado de la tarea. El archivo debe implementar las funciones descritas en el enunciado

usando los nombres de las funciones, el tipo de retorno y los parámetros dados en las implementaciones de ejemplo.
- Eres libre de implementar otras funciones.
- Los programas enviados no deben leer desde entrada estándar, escribir a salida estándar, ni interactuar con otros archivos. Sin embargo, sí pueden escribir a la salida de error estándar.
- Tus programas enviados no deben llamar **`exit()` o `System.exit()`**.
- Al probar tus programas con el evaluador de ejemplo, la entrada debe cumplir con el formato y restricciones descritos en el enunciado de la tarea, de lo contrario el comportamiento del programa puede ser inesperado.
- En los datos de entrada del evaluador de ejemplo, cada dos datos consecutivos en la misma línea habrá exactamente un espacio de separación entre ellos, salvo que otro formato haya sido explícitamente especificado en el enunciado.
- Para probar tu código en tu ordenador te recomendamos que utilices los scripts que facilitamos en el paquete adjunto. Si no los utilizas, asegúrate de añadir la opción `-std=gnu++14` al compilar, especialmente en C++.

Convenciones

Las tareas especifican declaraciones de funciones empleando los tipos genéricos `int`, `int64`, `int[]` (arreglo).

En todos los lenguajes de programación soportados, los evaluadores usan los siguientes tipos de datos o implementaciones:

Lenguaje	**`int`**	**`int64`**	**`int[]`**	**tamaño de un arreglo `a`**
C++	`int`	`long long`	`std::vector<int>`	`a.size()`
Java	`int`	`long`	`int[]`	`a.length`

Límites

Tarea	**Límite de tiempo**	**Límite de memoria**
line	solo-salida	solo-salida
vision	1 seg	1024 MB
walk	4 seg	1024 MB