**fstream:** Flujos hacia/desde ficheros. Permite la manipulación de archivos desde el programar, tanto leer como escribir en ellos.

**iosfwd:** Contiene declaraciones adelantadas de todas las plantillas de flujos y sus typedefs estándar. Por ejemplo ostream.

**iostream:** Parte del a STL que contiene los algoritmos estándar, es quizá la más usada e importante (aunque no indispensable).

**List:** Parte de la STL relativa a contenedores tipo list; listas doblemente enlazadas

**math:** Contiene los prototipos de las funciones y otras definiciones para el uso y manipulación de funciones matemáticas.

**memory:** Utilidades relativas a la gestión de memoria, incluyendo asigna dores y punteros inteligentes (auto\_ptr).

**"auto\_ptr"** es una clase que conforma la librería memory y permite un fácil manejo de punteros y su destrucción automáticamente.

**new:** Manejo de memoria dinámica

**numeric:** Parte de la librería numérica de la STL relativa a operaciones numéricas.

**ostream:** Algoritmos estándar para los flujos de salida.

**queue:** Parte de la STL relativa a contenedores tipo queue (colas de objetos).

**stdio:** Contiene los prototipos de las funciones, macros, y tipos para manipular datos de entrada y salida.

**stdlib:** Contiene los prototipos de las funciones, macros, y tipos para utilidades de uso general.

**string:** Parte de la STL relativa a contenedores tipo string; una generalización de las cadenas alfanuméricas para albergar cadenas de objetos. Muy útil para el fácil uso de las cadenas de caracteres, pues elimina muchas de las dificultades que generan los char

**typeinfo:** Mecanismo de identificación de tipos en tiempo de ejecución

**vector:** Parte de la STL relativa a los contenedores tipo vector; una generalización de las matrices unidimensionales C/C++

**forward\_list:** Esta librería es útil para implementar con gran facilidad listas enlazadas simples.

**List:** Permite implementar listas doblemente enlazadas (listas enlazadas dobles) fácilmente.

**Iterator:** Proporciona un conjunto de clases para iterar elementos.

**Regex:** Proporciona fácil acceso al uso de expresiones regulares para la comparación de patrones.

**Thread:** Útil para trabajar programación multi-hilos y crear múltiples hilos en nuestra aplicación.

**Exception:** Provee varios tipos y funciones relacionadas al manejo de excepciones, incluyendo std::exception, la clase base para todas las excepciones arrojadas por la biblioteca estándar.

Fuentes de consulta

<https://es.wikipedia.org/wiki/Biblioteca_est%C3%A1ndar_de_C%2B%2B>