Aspectos a Recordar

1. Entregar un documento formateado con Doxygen que contenga la especificación y la implementación de la práctica (ver ejemplo de práctica en assig/pro2/sessio10).
2. El programa debe contener varias clases (o módulos). No debe contener un único módulo que contenga todo. Tampoco es recomendable que exista un solo módulo que contenga a todos los demás.
3. Se debe mantener la independencia entre especificación e implementación. La especificación de una operación pública no debe hacer referencia a los detalles de su implementación (por ejemplo, a los atributos privados de su clase o de otras, u operaciones privadas de su clase o de otras clases). En general los atributos se declaran como privados.
4. Las operaciones públicas sólo pueden tener como resultados y como parámetros objetos de las clases definidas por vosotros o tipos de datos simples (`int, double, cha, bool` o `string` están permitidos). Esto es, no pueden tener parámetros ni devolver resultados de las las clases `BinTree, vector, list, queue, stack, map, set` o `iterator`.
5. Pensar en la descomposición modular:
   a) las clases en las que se descompone el programa;
   b) el diagrama modular (generado por Doxygen) debe reflejar las relaciones de "uso" entre las clases y no los "include" necesarios para que el programa compile;
   c) asignación racional de las operaciones a las distintas clases;
   d) distinción entre operaciones públicas y privadas (las operaciones que sólo se usan dentro de una clase no deben aparecer en el apartado público);
   e) especificación correcta de operaciones públicas y privadas, esto incluye precondición, postcondición y cabecera;
   f) identificad las opeaciones static, si usáis alguna;
   g) prestad atención al paso de parámetros (por referencia constante, referencia o valor) de manera que se eviten copias de datos innecesarias;
   h) distinguid entre operaciones consultoras (i.e. `const` detrás de la lista de parámetros) y modificadoras.
6. La implementación de los tipos (atributos) y de las operaciones debe ser eficiente. Es decir, no debe ser redundante en el uso del espacio de memoria y debe minimizar el coste de tiempo asintótico de las operaciones.
7. La especificación de las operaciones recursivas debe ser correcta. Prestad atención a la precondición, postcondición, cabecera (paso de parámetros y calificadores `const` o `static`) si son necesarios.