1. If you’re not sure which container to use, write your code so that it uses only operations common to both vectors and lists: Use iterators, not subscripts, and avoid random access to elements. That way it will be easy to use either a vector or a list as necessary.
2. Default ctor, if **array** is created by default ctor, then the elements are default-initialized (**not value-initialized**).
3. Assignment related operations **invalidate iterators, references, and pointers into the left-hand container**. They remain valid after a swap, and (except arrays and strings) the container to which they refer are swapped.
4. Excepting **array**, swap does not copy, delete, or insert any elements and is guaranteed to run in constant time
5. Adding elements to a vector, string, or deque potentially invalidates all existing iterators, references, and pointers int to the container.
6. The members that access elements in a container return reference.
7. Removing elements **anywhere but the beginning or end** of a **deque** invalidates all iterators, references, and pointers. Iterators, references, and pointers to elements after the erasure point in a **vector** or **string** are invalidated.
8. The members that remove elements do not check their arguments. The programmer must ensure the elements(s) exist before removing them.
9. If resize shrinks the container, then iterators, references, and pointers to the deleted elements are invalidated; resize on a vector, string or deque potentially invalidates all iterators , pointers and references
10. Because code that add or removes elements to a container can invalidate iterators, you need to ensure that the iterator is repositioned, as appropriate, after each operation that change the container.
11. Don’t cache the iterator returned from end() in loops that insert or delete elements in a deque, string, or vector
12. A call to **reserve** will never reduce the amount of space that the container uses. Similarly, the **resize** members change only the number of elements in the container, **not its capacity.**
13. The **stod** function reads the **string** it is given until it finds a character that cannot be part of a number.