1.Iterator

Iterator 是一个迭代器接口，专门用来迭代各种 Collection 集合，包括 Set 集合和 List 集合。

Iterator 接口定义的方法：

boolean	**hasNext**() 如果仍有元素可以迭代，则返回 true。
E	**next**() 返回迭代的下一个元素。
void	**remove**() 从迭代器指向的 collection 中移除迭代器返回的最后一个元素（可选操作）。

2.ListIterator

public interface **ListIterator<E>**extends Iterator<E>

列表迭代器，允许按任一方向遍历列表、迭代期间修改列表，并获得迭代器在列表中的当前位置。ListIterator 没有当前元素；它的光标位置始终位于调用 previous()所返回的元素和调用 next()所返回的元素之间。

注意，remove() 和 set(Object) 方法*不是* 根据光标位置定义的；它们是根据对调用 next() 或 previous() 所返回的最后一个元素的操作定义的。

void	**add**(E e) 将指定的元素插入列表（可选操作）。
boolean	**hasNext**() 以正向遍历列表时，如果列表迭代器有多个元素，则返回 true（换句话说，如果 next 返回一个元素而不是抛出异常，则返回 true）。
boolean	**hasPrevious**() 如果以逆向遍历列表，列表迭代器有多个元素，则返回 true。
E	**next**() 返回列表中的下一个元素。
int	**nextIndex**() 返回对 next 的后续调用所返回元素的索引。
E	**previous**() 返回列表中的前一个元素。
int	**previousIndex**() 返回对 previous 的后续调用所返回元素的索引。
void	**remove**() 从列表中移除由 next 或 previous 返回的最后一个元素（可选操作）。
void	**set**(E e) 用指定元素替换 next 或 previous 返回的最后一个元素（可选操作）。