Interface RandomAccess

docs.oracle.com/javase/8/docs/api/java/util/RandomAccess.html

public interface RandomAccess

Marker interface used by `List` implementations to indicate that they support fast (generally constant time) random access. The primary purpose of this interface is to allow generic algorithms to alter their behavior to provide good performance when applied to either random or sequential access lists.
The best algorithms for manipulating random access lists (such as `ArrayList`) can produce quadratic behavior when applied to sequential access lists (such as `LinkedList`). Generic list algorithms are encouraged to check whether the given list is an `instanceof` this interface before applying an algorithm that would provide poor performance if it were applied to a sequential access list, and to alter their behavior if necessary to guarantee acceptable performance.

It is recognized that the distinction between random and sequential access is often fuzzy. For example, some `List` implementations provide asymptotically linear access times if they get huge, but constant access times in practice. Such a `List` implementation should generally implement this interface. As a rule of thumb, a `List` implementation should implement this interface if, for typical instances of the class, this loop:

```
for (int i=0, n=list.size(); i < n; i++)
    list.get(i);
```

runs faster than this loop:

```
for (Iterator i=list.iterator(); i.hasNext(); )
    i.next();
```

This interface is a member of the Java Collections Framework.

Since:
1.4