KEY POINTS

List ADT

- General support for adding and removing.
- Allows for arbitrary positions.

Java: java.util.List

Array List

- Space used is $O(n)$
- Accessing an index takes $O(1)$ time.
- Add and Remove: $O(n)$
- push will make space:
  - Constant increase: $\Omega(n^2)$
  - Double capacity: $O(1)$ (amortized time.)
- Amortized time is the average time i.e. $\frac{T(n)}{n}$

NAME/DATE/SUBJECT

ADT with general support - List

NOTES

Java List methods:

- size()
- isEmpty()
- get(i) — $O(1)$
- set(i, e) — index, element
- add(i, e) — push(e) $\Rightarrow \Omega(n^2)$, $O(1)$
- remove(i, e)

} will allow for operations within the List - will have to shift elements either way. $O(n)$

Array Lists

- When inserting into an -L, you must make room for the new element. You can either allocate just enough room (some constant c) or you can double the current capacity
  - Each push operation has amortized time (average time) $T(n)/n$.
  - Allocating a constant has a.t. $\Omega(n^2)$

$$c + 2c + 3c + \dots + kc = c(1 + 2 + 3 + \dots k) = \frac{ck(k+1)}{2}$$; c constant

  - Doubling capacity has a.t. $O(1)$ $\quad \frac{O(n)}{n} = 1$

SUMMARY

KEY POINTS

NAME/DATE/SUBJECT

Positional List | Iterator

NOTES

use this if you will modify the head or tail often.

Basically a doubly linked list - however, we create a variable (of object type) to track the position. This allows us to modify items around the position.

* the position is only modified if its item in the list is deleted!

Iterators

Java provides an interface Iterable; this allows you to create multiple iterators for a single list - one common instance is the ArrayList.

SUMMARY