



Sequence Types

Sequence: collection of elements organized in a specified order • allows random access by rank or position Stack: sequence that can be accessed in LIFO fashion Queue: sequence that can be accessed in FIFO fashion Deque: sequence that can be accessed in FIFO fashion Deque: sequence accessed by added to or removing from either end Vector: sequence with random access by rank List: sequence with random access by position

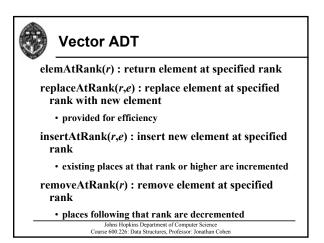
Places in a Sequence Rank
• Place specified by number of places before the place in question

· Abstraction of the concept of array index

Position

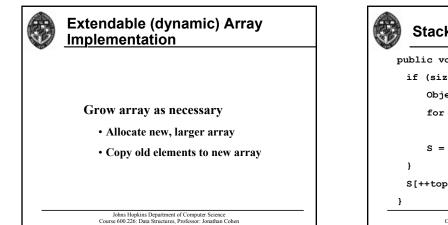
- Place specified by which place precedes and which place follows the place in question
- Abstraction of the concept of (doubly) linked list node

Johns Hopkins Department of Computer Science Course 600.226: Data Structures, Professor: Jonathan Coher

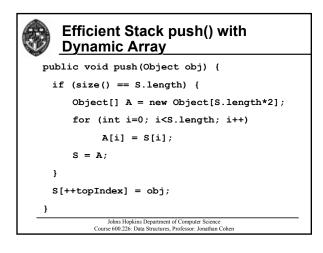


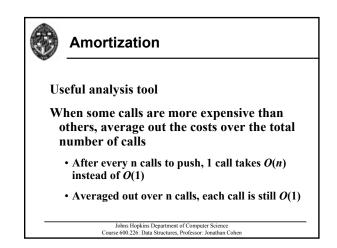
Array-based Vector Implementation
<pre>insertAtRank(r, e) { if (size == A.length)</pre>
throw new VectorFull();
<pre>for (int i=size-1; i>=r; i)</pre>
A[i+1] = A[i];
$\mathbf{A}[\mathbf{r}] = \mathbf{e};$
size++;
}
removeAtRank(r) {
Object e = A[r];
<pre>for (int i=r; i<size-1; <="" i++)="" pre=""></size-1;></pre>
A[i] = A[i+1]; size;
Size;
 J Johns Hopkins Department of Computer Science
Course 600.226: Data Structures, Professor: Jonathan Cohen

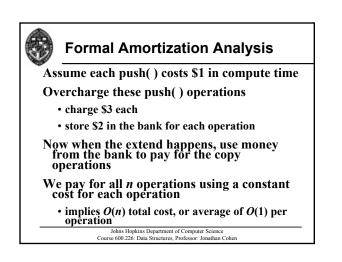
Analysis					
elemAtRank:	<i>O</i> (1)				
replaceAtRank:	<i>O</i> (1)				
insertAtRank:	<i>O</i> (<i>n</i>)				
removeAtRank:	<i>O</i> (<i>n</i>)				
Note: insert and re	Note: insert and remove are O (1) at end				
• may make <i>O</i> (1) a as in queue	at start with index wrapping,				
	partment of Computer Science tructures, Professor: Jonathan Cohen				

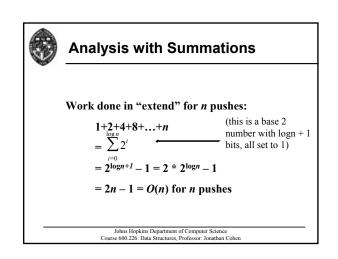


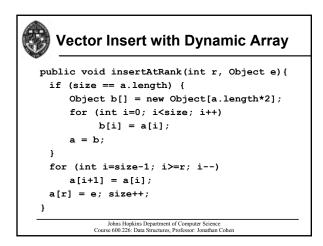
Stack push() with Dynamic Array public void push(Object obj) { if (size() == S.length) { Object[] A = new Object[S.length+1]; for (int i=0; i<S.length; i++) A[i] = S[i]; S = A; } S[++topIndex] = obj; } </pre>

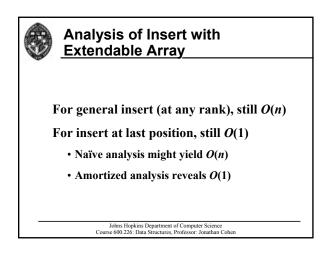


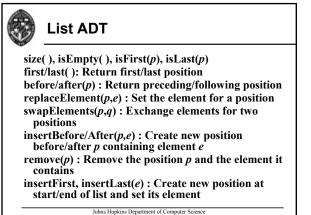




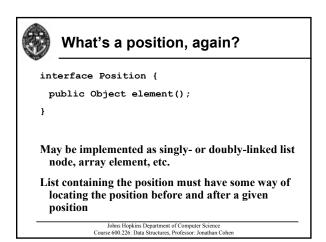


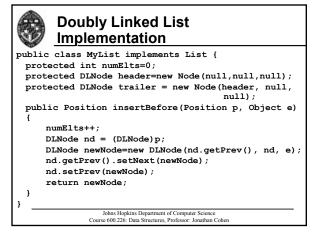


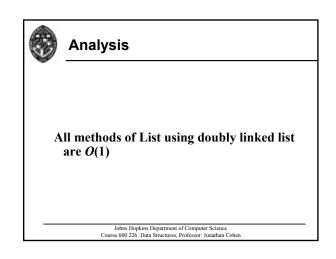


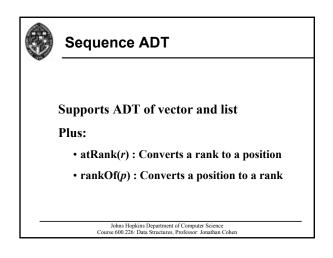


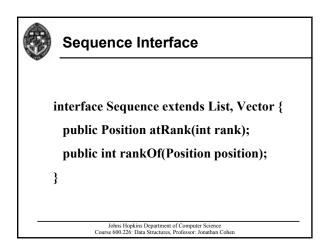
Course 600.226: Data Structures, Professor: Jonathan Cohen













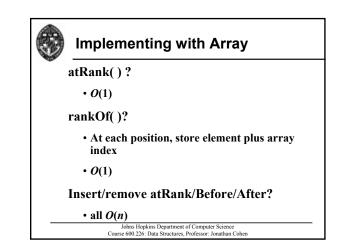
atRank() and rankOf()?

• Both *O*(*n*)

Perform Vector methods by first finding Position at proper rank, then doing insert, delete, etc.

• Finding position is O(n), though the actual insert/delete is only O(1)

Johns Hopkins Department of Computer Science Course 600.226: Data Structures, Professor: Jonathan Cohen



	Comparison			
C	Operation	Array		List
S	ize, isEmpty	O (1)		O (1)
fi	irst, last, before, after	O (1)		O (1)
iı	nsertFirst, insertLast	O (1)		O (1)
r	eplaceElement,swapElement	O (1)		O (1)
iı	nsertAfter, insertBefore	O(n)	>	O (1)
r	emove	O(n)	>	O (1)
a	tRank, rankOf, elemAtRank	O (1)	<	O(n)
r	eplaceAtRank	O (1)	<	O(n)
iı	nsertAtRank, remvAtRank	O (n)		O (n)
	Johns Hopkins Department of Compute Course 600.226: Data Structures, Professor: J			