

Computer Science 136

Data Structures

Lecture #15 (October 20, 2021)

Notes:

1. Announcements.
 - (a) Exams back. Mean: 78%. Hold questions until Friday.
 - (b) Lab 4 out. Lab 2 back today. Lab 3 on the way.
 - (c) Questions?
 2. Linear structures: Sometimes advanced structures come through interface simplification.
 - (a) Supports structures with three basic operations: **add**, **remove**, and **get**.
 - (b) Does *not* support the notion of node *indices*. You cannot remove the second value.
 - (c) Does not determine *where* **add** places a value or **remove** extracts a value.
 - (d) Does not require an **add** and **remove** to “undo” each other (though they may).
 3. **Linear** interface and **AbstractLinear** abstract class. One never directly constructs linear objects.
 4. The **Stack** interface: a last-in, first-out structure. Adds **push**, **pop**, and **peek**.
 - (a) **AbstractStack** identifies **push** with **add**, etc.
 - (b) **StackList** uses a list as the internal implementation.
 - (c) **StackVector** a vector-based implementation.
 - (d) **StackArray** a very fast implementation, but size-limited.
 5. The **Queue**: a first-in, first-out structure.
 - (a) **AbstractQueue** identifies **enqueue** with **add**, etc.
 - (b) **QueueList** uses a (different) list as the internal implementation.
 - (c) **QueueVector** a vector-based implementation.
 - (d) **QueueArray** a very fast implementation, but size-limited.
 6. Solving Mazes:
 - (a) Stacks lead to *depth-first search* and possible fast termination.
 - (b) Queues lead to *breadth-first search* and will find shortest solution.
-