CSCI 241

Scott Wehrwein

Interface vs Implementation
Abstract Data Types

Goals

- Understand the distinction between interface and implementation in the context of operations on data structures.
- Know the meaning of abstract data type.

Last time: Big-Deal CS Concept #1:

Asymptotic Runtime Analysis

Today: Big-Deal CS Concept #2:

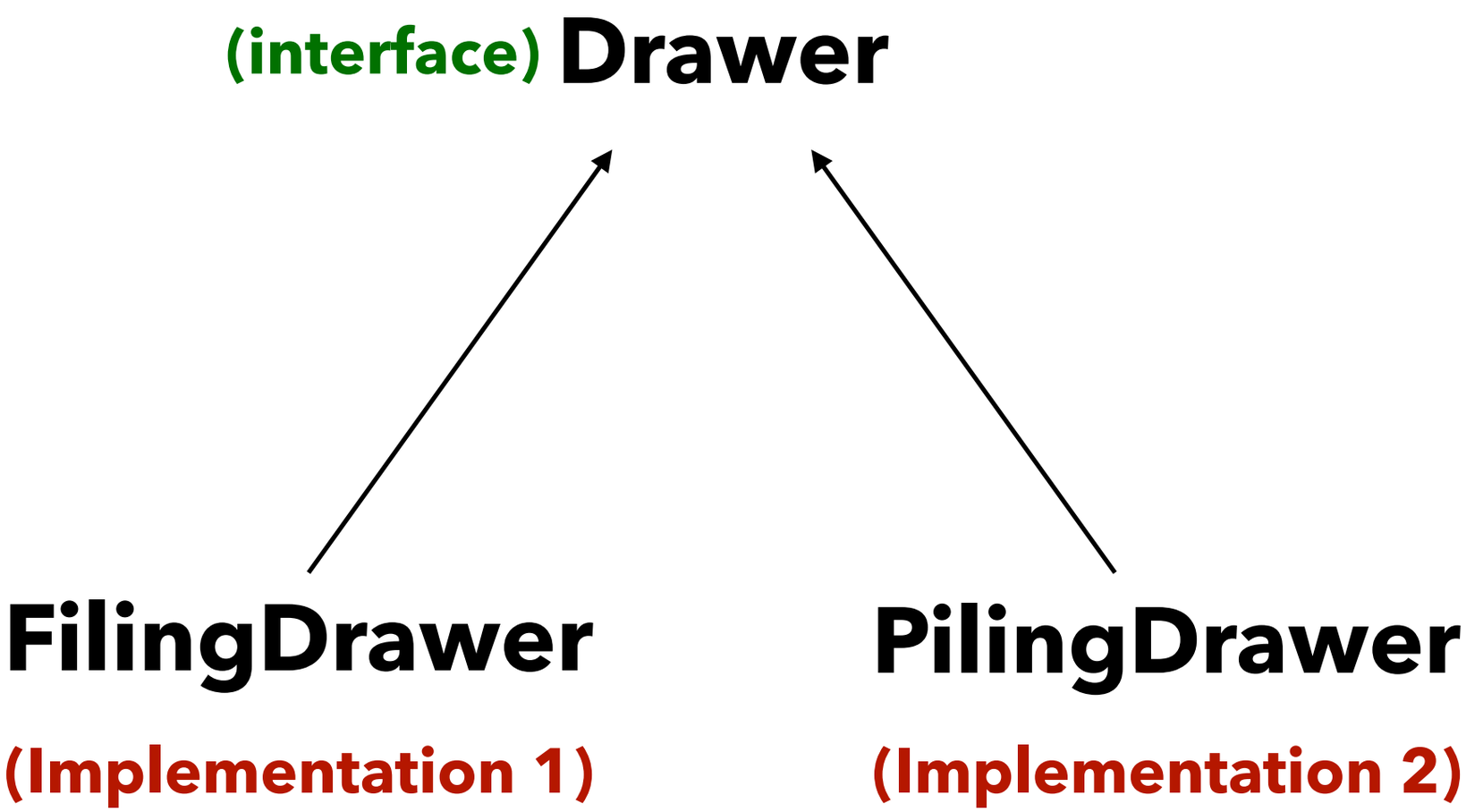
Interface vs Implementation and Abstract Data Types

What the operations do

An abstract data type specifies only interface, not implementation

How they are accomplished

Interface vs Implementation: Example



Interface vs Implementation: Example

abstract data type

Drawer:

Interface

- Contains(item) - returns true iff item is in the drawer (short for "if and only if")
- Add(item) - adds item to the drawer
- Remove(item) - removes item from the drawer if it exists

FilingDrawer implements Drawer:

Implementation

Contains(item):

```
binary search drawer for item
```

Comparing Implementations

class FilingDrawer:

- Contains(item):

  `binary search drawer for item` O(log(N))

class PilingDrawer:

- Contains(item):

  `linear search drawer for item` O(N)

Comparing Implementations

class FilingDrawer: O(N)

- Add(item):

```
insert item in sorted position in drawer
```

class PilingDrawer: O(1)

- Add(item):

```
insert item into the drawer wherever
it's easiest
```

Which is better?

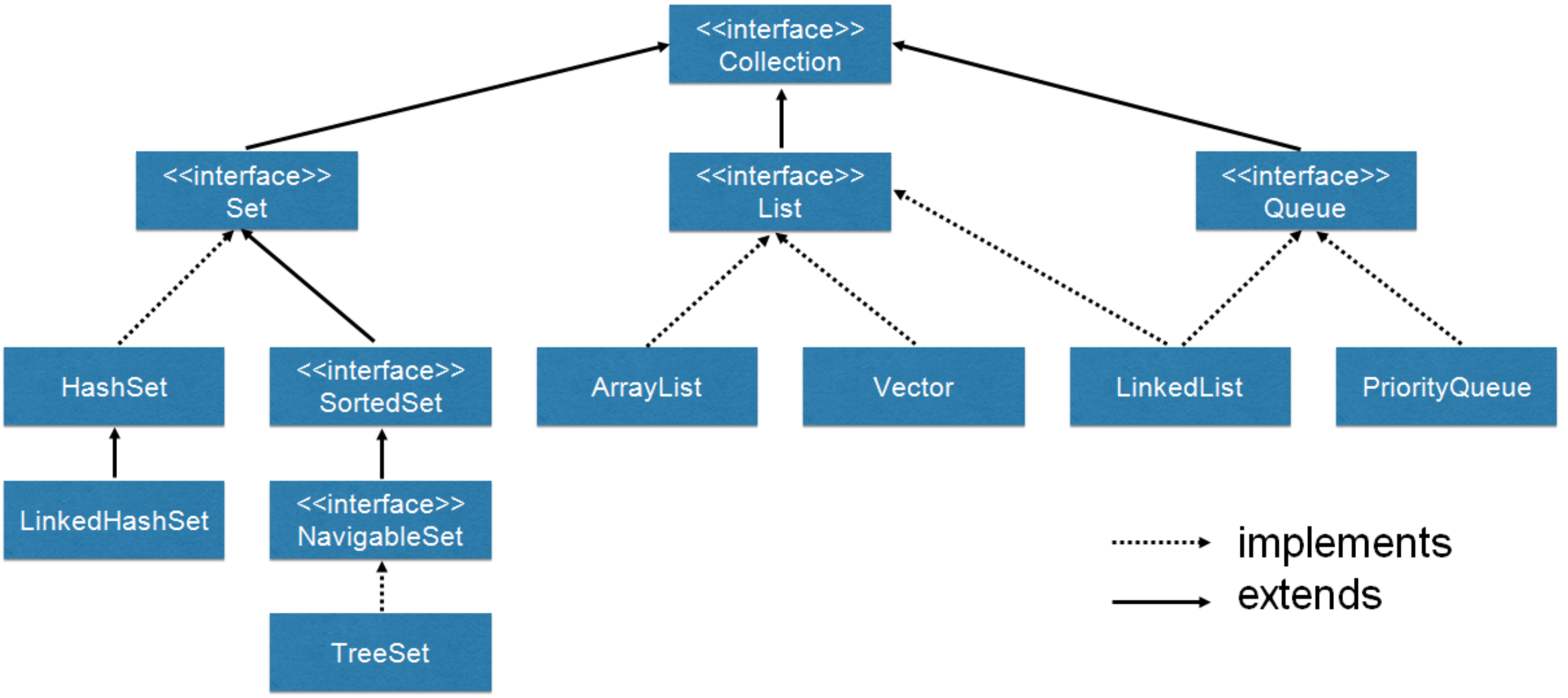
	PilingDrawer	FilingDrawer
Add	O(1)	O(N)
Find	O(N)	O(log N)

The best choice of **implementation** for an **abstract data type** usually depends on **how** you'll be using it.

Abstract Data Types: Examples

ADT	Notable Implementations	When is it covered?
List, Queue, Stack	Array, Nodes	(CSCI 145)
Set	Array, Nodes Tree, Hash table	(Weeks 4,5,7)
Tree	Array, Nodes	(Weeks 4-6; A2)
Priority Queue	Heap	(Week 6; A3)
Map	Tree, Hash table	(Week 7; A3)
Graph	Array, Nodes	(Weeks 8-9; A4)

The Java Collections Hierarchy



Philosophical Question:

Is an array an ADT?