CS 351: Data Structures and Algorithms

—

Week 3

Interfaces

- Specification of behaviors that an ADT may implement and use
- All methods are abstract
- Some methods may have default implementations
- All fields are public static final
- Cannot be instantiated
- Achieve higher level of abstraction
- Decrease coupling
    - Coupling is the interdependence between classes/modules; good software has low coupling

Abstract Classes

- Specification of behaviors that an ADT may implement and use
- Some or all methods are already fully implemented
- Can have fields that are not public static final
- Cannot be instantiated
- Achieve higher level of abstraction
- Decrease coupling
- Methods don’t have to be abstract

Generics

- Parameterization of types
- Declared in class header:
  - public class ArrayList<E>
- Must provide the type when instantiating an object:
  - ArrayList<String> a;
- Compiler warning if the type is not defined in the angle brackets
- Cannot instantiate a generic object, check if an object is a generic type, or create an array of generics

Iterators vs Cursors

Iterators

- Separate object implementing an interface
- Behaviors:
  - hasNext
  - next
    - NoSuchElementException
  - remove*
- Can have many iterators
- Traversing the collection does not change the state of the ADT
- If collection changes without using the iterator, the iterators “go stale”
  - How can we keep track of this?
  - ConcurrentModificationException
  - Synchronization

Cursors

- Part of the ADT
- Can only have one
- Using the cursor changes the state of the ADT
- Simplicity can be beneficial to a client; less likely to make mistakes
  - More complicated for the implementer

Abstract Collection

- public abstract Iterator<E> iterator()
- public abstract int size()
- public boolean isEmpty()
- public boolean contains(Object o)
- public Object[] toArray()
- public <T> T[] toArray(T[] a)
- public boolean add(E e)
  - UnsupportedOperationException
- public boolean remove(Object o)
  - UnsupportedOperationException
- public boolean containsAll(Collection<?> c)
- public boolean addAll(Collection<? extends E> c)
  - UnsupportedOperationException
- public boolean removeAll(Collection<?> c)
  - UnsupportedOperationException
- public boolean retainAll(Collection<?> c)
  - UnsupportedOperationException
- public void clear()
  - UnsupportedOperationException*
- public String toString()

When should you override methods?

It depends on the specification!

- The method does not have a functional implementation
- The provided implementation is inefficient
- The provided implementation differs from the specification requirements