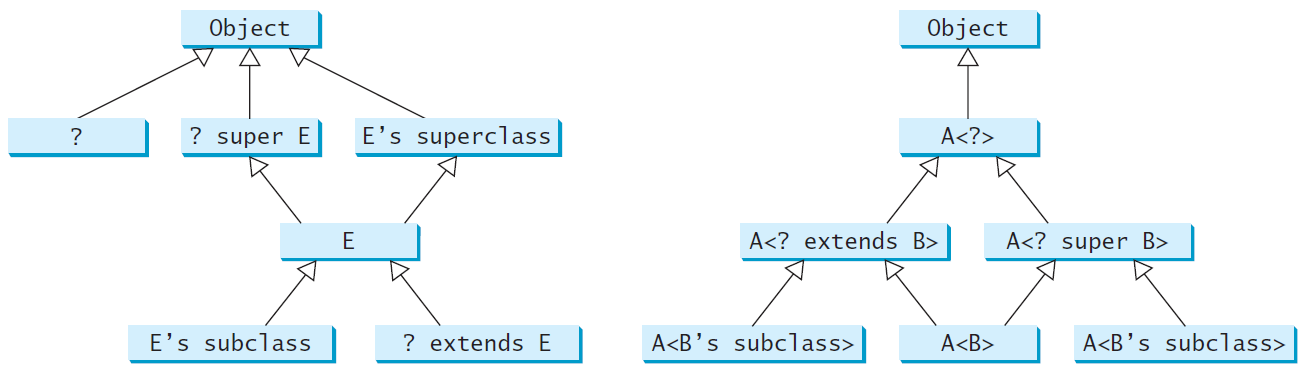
CS 150 Topics List

Chapter 19

1. Motivations for Generics (19.1, 19.2)
   1. Parameterization of types.
   2. Formal Generic Type (<E> or <T>) vs. Actual Concrete Type.
   3. Generic instantiation.
   4. Reduction in run-time errors (changing them into compile-time errors).
   5. Reduction in the need to typecast objects.
2. Defining Generic Classes and Interfaces (19.3)
   1. Constructor definition header does not contain angle (diamond) brackets.
   2. Multiple generic parameters.
   3. Generic types can only be references. Primitives cannot be used as concrete generic types.
      1. Autoboxing and autounboxing can help here.
   4. Generic types can be applied to classes and/or interfaces.
3. Generic Methods (19.4)
   1. Static methods can be defined using generic types (page 743).
   2. Bounded generic types: <E extends GeometricObject>
4. Raw types and backwards compatibility (19.6)
   1. Using no concrete type causes the generic concrete type to be Object.
   2. This allows for backward compatibility with older code that predates Generics.
5. Wildcard Generic Types (19.7)
   1. Unbounded wildcard <?>
   2. Bounded wildcard <? extends T>
   3. Lower-bounded wildcard <? super T>



1. Type Erasure (19.8)
   1. Concrete generic types are erased at compile time and replaced with Object.
2. Restrictions on Generics (19.8)
   1. Cannot use new E
   2. Cannot use new E[]
   3. A generic type parameter for a class cannot be used in a static context.
   4. Exception classes cannot be generic