CSC 243 - Java Programming

Abstract Classes and Interfaces

Abstract Classes

- An abstract class contains methods with no implementation called abstract methods
- The abstract methods are intended to be implemented in concrete subclasses
- An abstract class cannot be used to create objects

```
public abstract class C {
  public int regularMethod() {
    return 0;
  }
  public abstract int abstractMethod();
}
```

Abstract Classes Continued

- An abstract method cannot be contained in a nonabstract class
- An abstract class can not be instantiated with the `new` operator, but can be used as a declared type
- An abstract class can contain a constructor, which would be called in the subclass constructor
- An abstract class does not need to contain an abstract method
- A subclass can be abstract even if the superclass is concrete

Interfaces

- An interface is a class-like construct that contains only constants and abstract methods
- Intended to specify common behavior for objects of related or unrelated classes
- Interface inheritance is the relationship between the class and the interface

```
public interface I {
  public abstract int abstractMethod();
}

public class C implements I {
  /* definition of abstractMethod */
}
```

Interfaces Continued

- Classes are nouns, interfaces can be adjectives or nouns
- A Java class can `extend` only one class, but can `implement` multiple interfaces
- A Java interface can inherit from other interfaces
- In general, interfaces are preferred over abstract classes because an interface can define a common supertype for unrelated classes