Test 1 – during your lab period on the week of Feb 10

About ½ labtest, about ½ written

Test material up to and including Wednesday February 5

Last time we looked at special methods inherited from Object

* equals()
* toString()
* hashCode()
* clone()
* getClass()

Implementing an interface – you can declare that your class implements an “interface” – all this means is that your class contains certain defined methods

Comparable<T> interface

* Comparable<BankAccount>

If class BankAccount implements the Comparable<BankAccount> interface, then it must include whatever methods the interface requires

Comparable<T> requires you to include the method

public int compareTo(T x)

in this case T = BankAccount

public int compareTo(BankAccount b)

Comparable API says: compareTo returns an integer, which is:

-1 if this > b

+1 if this < b

0 if this == b (consistent with equals())

a.compareTo(b): this == a

You have to declare that your class implements the interface

public class BankAccount

implements Comparable<BankAccount>

{ … }

Coming back to mixed static-nonstatic features …

Singleton class – a singleton class is a class that only allows one object of its type to be created

In a singleton, the constructor must be private – this is to prevent the client from creating as many objects of this type as s/he wants

Instead there is a static method, usually called getInstance, that returns the one and only instance of the singleton – getInstance has to be static because you need to run the method even if no objects of singleton type have been created yet