Method Overloading

In Java, it is legal to have two (or more) methods with the same name, taking different parameters

This is called “method overloading” (or just “overloading”).

Overloaded methods are distinguished from each other using their “method signature”.

Method signature is the declaration of the method, minus the return type (and everything before).

e.g.

public double withdraw(double amount)

Method signature:

withdraw(double amount)

The return type is NOT INCLUDED in the method signature, therefore, overloaded methods CANNOT be distinguished by return type.

Beware of parameters that may be converted – e.g. it is possible to declare overloaded methods with types “double” and “float” – be aware of which one is called – to be explicit, use casting

method(double x)

method(float x)

Remember constructors can be overloaded as well – same rules apply

Pitfalls in pass-by-reference

Remember that objects are referred to by their address in memory

BankAccount b;

b contains an address in memory where a BankAccount object is stored; b is not itself the data at that location

* e.g. copying:

BankAccount c = b;

// not a copy – it is a pointer

// to the same object

* testing equality

c == b

// not a good way to test equals

c.equals(b)

// the right way

Primitive types (int, double, float, char …) are NOT objects – stored by value (i.e. variable name refers to the actual value stored).