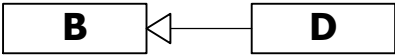

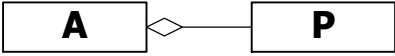
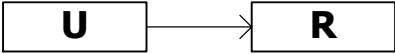


Relationships between Classes

Relationship	Diagram	Code	Explanation
Inheritance D "is-a" B		<pre>public class D : B { ... }</pre>	Derived class D is a specialization of the Base class B. D inherits all the members of B except constructors
Composition Ownership, P is "part-of" C		<pre>public class C { ... private double p = 3.142; }</pre>	Composite class C owns, or contains, a part class P. P is created and destroyed with C. The interface of P is visible only to C, not its clients. Example: P is a value type.
Aggregation Ownership, P is "part-of" A		<pre>public class A { ... private P p = new P(); }</pre>	The Aggregator class A owns a part class P. P is created by a member function of A, and so its lifetime is strictly less than that of A. Example: P is a reference type.
Using Referral: U uses R through a reference		<pre>public class U { ... public void register(R r) // use r } }</pre>	A class U uses instance of class R, to which it holds a reference. R is created by some other entity and a reference to it is passed to some member function of class U.