Shadowing and Overriding

	Static	**Instance**
Variable	shadow	shadow
Method	shadow	override

.

Question: When should you make a method **static**?

One possible answer: When you do not want it to be overridden in a subclass.

Another possible answer: When it only deals with static variables (because then you might not want it overridden in a subclass, since the variable will shadow).

Another possible answer: When you want to be able to use a method without defining an instance of an object.

For example, we call `println()` by including the code: "`System.out.println()`". We do NOT include "`System s = new System();`" and then call `s.printon()` anywhere in the code.

Question: When should you make a variable **static**?

One possible answer: When you want to be able to access its value without going through an instance of its class.

For example, in the `Student` class, we want to know what is the total tstudentCount without relying on a specific student to keep track of it and without allowing different students to have different values of `studentCount`.