CSE143X Lecture Questions for Wednesday, 10/21/20

Time (e.g., 12:45)	Question	Answer
35:00	**Take off your headphones - loud noise warning.** *(This message is sponsored by the American Hearing Association)*	
11:00	Are we going to be able to use toString() so we don't have to call the method? (Thanks! - oh, just got there in the lecture.)	Yes. You don't have to call toString() in most cases.
	A bit of a random but out of personal curiosity, what is the course website programmed in? I saw that CSE143's course site uses Jekyll. I'm assuming HTML/CSS/JS but does that mean Stuart cracks open the source code every week to add links and stuff? Re: Ohh interesting. Thanks for the response! I guess it's like a DIY Content Management System. Very cool :D	I access a directory where files are stored for the website. I just modify specific files like calendar.shtml to add new content. Marty Stepp developed most of the JS that we use on the site.
25:02	Screen blacks out.	Thanks...still working on this.
~36	Ever since loud sound (lightning?), mic is picking up static (even 10 minutes after) [on the video it seems like he struck the mic with his hand. Lol] [[it looked to me like he tried to cover the mic to shield it from lightning sound but maybe]]	Not sure why that would be...I'll check it out. Very simple...gesturing and accidentally hit the microphone.
16:54	So just to be clear, if you write your own toString() method but don't call it, java will override it's own toString() method and call the one you've written? And the same would happen if we did System.out.println(p1) without the concatenation right?	Yes, Java generally calls your toString method in most cases (string concatenation, println, etc).

17	What would happen if your toString method did not return a String? Would it still work the same?	You aren't allowed to change the return type of an inherited method like toString, so it wouldn't compile if you gave it a void return type.
23:33	You mentioned that the default/empty constructor "sets the fields to be the [zero equivalent values]" Does that imply that if you have a non-empty constructor, and you don't initialize one of the fields, that that field won't be initialized? Re: Makes sense!	Java always auto-initializes all fields of an object before the constructor code is executed.
25	Are we allowed to have 2 methods of the same name and parameters but different return types?	No. The return type is not part of the signature.
48-ish	Do we tend to write static class methods ever in 143x? Or is the approach to use instance methods always. Re: cool, thanks!	We tend to have you practice static methods in the 142 part and then switch to instance methods for the 143 part. Because Java is an object-oriented language, it tends to have more instance methods. The exceptions tend to be things like the Math class where you don't have any state information or constants that can be static.
51:40	Will we ever lose style points for including unnecessary "this." notation for the sake of clarity?	No, that's a matter of personal choice, although we encourage you to go for consistency.
47	When would the 0 argument constructor ever call one with parameters, or rather how would it know what values to pass?	This is Java's equivalent to default values. For a point, it makes sense that the default x/y would be the origin. Often there is an obvious default.