Quiz5

ID	Score 10	JUnit Syntax(4)	test Add(2)	test Drop(2)	test credits(2)	test difficulty(2)	Remarks
5410545044	**4.5**	3	1	0	2	0	should use assertEquals instead of assertTrue(a == b)
5410545052	**4.5**	3	1	0	1	1	
5410546334	**0**						Testing a LinkedList, not CourseList. No useful tests.
5410546393	**0**						No files
5410547594	**0**						No files
5610545013	**0**						No JUnit test
5610545048	**6**	4	1	0	2	1	
5610545668	**5.5**	3	2	0	1	1	should use assertEquals(double, double, tolerance) instead of assertSame
5610545676	**8**	4	2	0	2	2	testAdd only testing size of courselist is poor logic
5610545684	**8.5**	3	2	2	1	2	should use assertEquals(double, double, tolerance) instead of assertSame
5610545692	**10**	4	2	2	2	2	good
5610545706	**0**						No files
5610545714	**9.5**	3	2	2	2	2	should use assertEquals(double, double, tolerance) instead of assertSame
5610545722	**10**	4	2	2	2	2	good
5610545731	**6.5**	3	1	2	1	1	should use assertEquals(double, double, tolerance) instead of assertSame. expected and actual args are reversed.
5610545749	**10**	4	2	2	2	2	good
5610545757	**10**	4	2	2	2	2	good
5610545765	**10**	4	2	2	2	2	good
5610545781	**0**						No files
5610545803	**4**	2	1	0	1	1	Don't use static CourseList object, use a new CourseList for each test. Use asertEquals(int,int) not assertEquals(Object, Integer)
5610545811	**10**	4	2	2	2	2	good
5610546222	**10**	4	2	2	2	2	good
5610546231	**9.5**	3	2	2	2	2	use assertEquals(int,int) not assertSame(Integer,Integer). For f.p. use assertEquals(dble,dble, tolerance)
5610546257	**0**						No files
5610546281	**9**	4	2	1	2	2	Tolerance in assertEquals should be > 0.
5610546290	**4.5**	3	1	0	1	1	Course c = new Course(...); assertNotNull(c); is useless.
5610546681	**9.5**	3	2	2	2	2	should use assertEquals(double, double, tolerance) instead of assertSame
5610546699	**6.5**	3	2	0	2	1	In assertEquals(expected, actual) args are reversed, no tolerance parameter for f.p. comparison.
5610546702	**10**	4	2	2	2	2	Tolerance in assertEquals should be > 0.

Quiz5

ID	Score 10	JUnit Syntax(4)	test Add(2)	test Drop(2)	test credits(2)	test difficulty(2)	Remarks
5610546711	**9**	4	2	2	1	2	In assertEquals(expected, actual) args are reversed.
5610546729	**0**						File contains no tests.
5610546745	**10**	4	2	2	2	2	good.
5610546753	**10**	4	2	2	2	2	good.
5610546761	**10**	4	2	2	2	2	good.
5610546770	**8**	4	2	2	1	1	
5610546788	**8.5**	3	2	2	1	2	Incorrect usage of tolerance parameter. Poor code in testAdd. In assertEquals(expected, actual) args are reversed.
5610546800	**0**						No useful tests.

Scoring:
For testAdd, testDrop, testGetCredits, testGetDifficulty: 1 point for each useful test (up to 2 pts each method under test).
You should test things that should fail as well as those that succeed.
The tests need to be somewhat conclusive. For example, this test is not conclusive:

```
Course c = makeCourse("219244", 3, 5.5);
assertTrue( courselist.add(c) );
```

You don't know if the course was *really* added or not. Maybe the method just returns "true".
A few people wrote very thorough tests, with descriptive comments. Well done!

For JUnit Syntax (4 x 0.5pt), I looked for correct use of JUnit. Some common errors are:
1) assertTrue instead of assertEquals:
assertTrue(expected == actual); should be: assertEquals(expected, actual);
If the assertTrue test fails, you won't know what the expected/actual values were.
2) assertSame for testing primitives.
I didn't deduct for this, but when assertSame(expected, actual) is for testing object references.
for primitives (like int) use assertEquals.
3) no tolerance on floating point comparisons. For example:

```
double expected = 7.5; // suppose this is the correct value
assertEquals( courselist.getDifficulty(), expected, 0.0);
```

You should have a small, non-zero tolerance.