What do specs and ScalaTest support?

They support behavior-driven development, emphasizing human-readable specifications of behavior of code with accompanying test cases.

Enable/disable `assert()` and `ensuring()`.

JVM's -ea and -da flags.

List the two forms of `assert()`.

- `assert(Boolean)`

- `assert(Boolean, String)` (explanation to be included in `AssertionError`.)

Describe the test directory hierarchy.

It typically mirrors the source tree's directory hierarchy.

How can you get more informative failure reporting than `assert()`?

ScalaTest's

```
assert(this === that)
expect(this) {
  that
}
```

Verify exception thrown, using ScalaTest.

```
intercept[IllegalArgumentException] {
  //something that throws it
}
```

Give a more Scala-esque way to use ScalaTest.

extend `org.scalatest.FunSuite`

```
test("my cool test") {
  //use assert() or exceptions
}
```

Compare specs and `org.scalatest.Spec`.

Specs provides matches to be used in ScalaTest or JUnit.

ScalaTest's Spec uses

```
describe("some object") {
  it("should do A") {
    //test for A
  }
}
```

What does ScalaCheck support?

Property-based testing.

You give properties with implication operator ==>, and it checks the properties on the values it generates for you.

Concisely check the conditions of method's result.

Use `ensuring(op: Any => Boolean)`

Raises `AssertionError` if predicate fails, and returns the argument correctly cast to result type of function otherwise.

```
ensuring(_.dollars >= 0)
```

Result of function will be inserted and returned if the predicate holds.