Fraction class

Generate a Kotlin class which implements fractional arithmetic. If you've forgotten what they taught you in school about fractions, check this out.

In this exercise we practice TDD (Test Driven Development). Your implementation should pass the following JUnit -tests:

```
internal class FractionMutableTest {

    @Test
    fun testCons() {
        val a = FractionMutable(2,4,-1)
        assert(a.toString() == "-1/2")
    }

    @Test
    fun testToString() {
        val a = FractionMutable(1,2,-1)
        assert(a.toString() == "-1/2")
    }

    @Test
    fun negate() {
        val a = FractionMutable(1,2, -1)
        a.negate()
        assert(a.toString() == "1/2")
    }

    @Test
    fun addPos1() {
        val a = FractionMutable(1,2)
        a.add(FractionMutable(1,3))
        assert(a.toString() == "5/6")
    }

    @Test
    fun addPosNeg1() {
        val a = FractionMutable(1,2)
        a.add(FractionMutable(1,3, -1))
        assert(a.toString() == "1/6")
    }

    @Test
    fun multPos() {
```

```
        val a = FractionMutable(1,2)
        a.mult(FractionMutable(1,3))
        assert(a.toString() == "1/6")
    }

    @Test
    fun multPosNeg1() {
        val a = FractionMutable(1,2)
        a.mult(FractionMutable(1,3, -1))
        assert(a.toString() == "-1/6")
    }

    @Test
    fun div() {
        val a = FractionMutable(8,3)
        a.div(FractionMutable(4,6))
        assert(a.toString() == "4/1")
    }

    @Test
    fun intPart() {
        val a = FractionMutable(8,3)
        assert(a.intPart() == 2)
    }
}
```

Phase 1

From the JUnit tests you are able to find out the methods needed and their parameters. You can first try your implementation with this simple main-function.

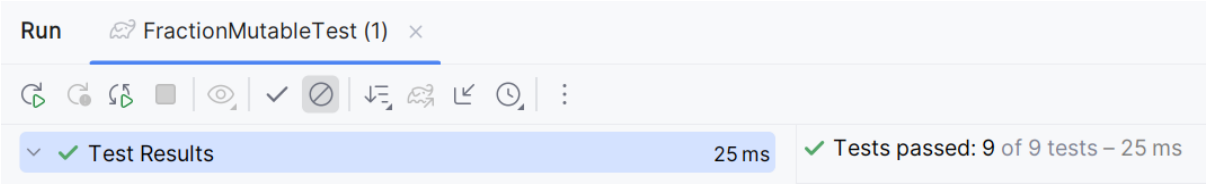
```
fun main() {
    val a = FractionMutable(1,2,-1)
    a.add(FractionMutable(1,3))
    println(a)
    a.mult(FractionMutable(5,2, -1))
    println(a)
    a.div(FractionMutable(2,1))
    println(a)
}
```

The output should be something like this:
-1/6
5/12
5/24

Process finished with exit code 0

Phase 2

Then should run the given JUnit tests, and improve the implementation until you get all tests passed. How to use JUnit with Kotlin, look at this.



Phase 3

Return source code and screenshot from the results of JUnit test.