@Author("Adrian Więch")

{ JUnit 5 CHEAT SHEET }

{ SAMPLE UNIT TEST }

```java
@Test
void should_RetTrue_When_DietRecom() {

  // given
  double wght = 90.0;
  double hght = 1.92;

  // when
  boolean recommended =
    BMICalc.isDietRecommended(wght, hght);

  // then
  assertTrue(recommended);
}
```

{ TEST TYPES }

```java
// basic test
@Test

// repeat test 10x
@RepeatedTest(10)

// parameterize single value
@ParameterizedTest
@ValuesSource(doubles = {70.0, 80.0})
void testName(Double param) { … }

// parameterize multiple values
@ParameterizedTest(name = "w={0}, h={1}")
@CsvSource(value = {
 "70.0, 1.82", "80.0, 1.72"
})
void testName(Double par1, Double par2)

// params from csv file, ignore header
@ParameterizedTest
@CsvFileSource(
  resources = "/diet-params.csv",
  numLinesToSkip = 1
)
void testName(Double par1, Double par2)
```

{ OTHER }

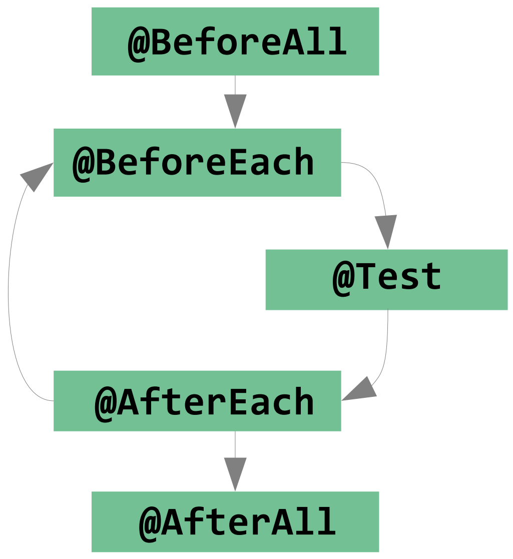
```java
// nested class
@Nested
class InnerClass { … }

// display name
@DisplayName("Custom name")

// skip test
@Disabled

// skip test under condition
assumeTrue(env.equals("prod"))
```

{ TEST LIFECYCLE }



{ ASSERTION TYPES }

```java
// check if x is true/false
assertTrue(x);
assertFalse(x);

// check if object is null
assertNull(object);

// check if expected equals actual
// (primitive types only)
assertEquals(expected, actual);

// check if array1 and array2
// contain the same elements
assertArrayEquals(array1, array2);

// check if doSth() throws
// SampleExeception
Executable executable = () ->  doSth();
assertThrows(
  SampleException.class,
  executable
);

// check multiple assertions
assertAll(
 () → assertEquals(expected1, actual1),
 () → assertEquals(expected2, actual2)
);

// set maximal execution time
Executable executable = () ->  doSth();
assertTimeout(
  Duration.ofMillis(500),
  executable
);
```