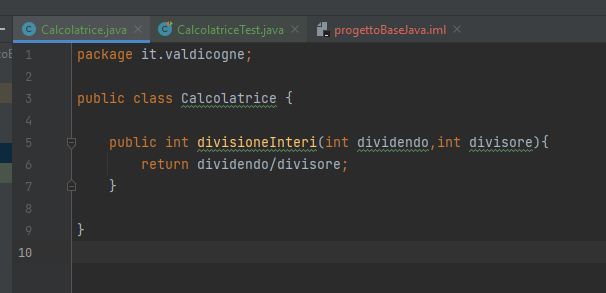
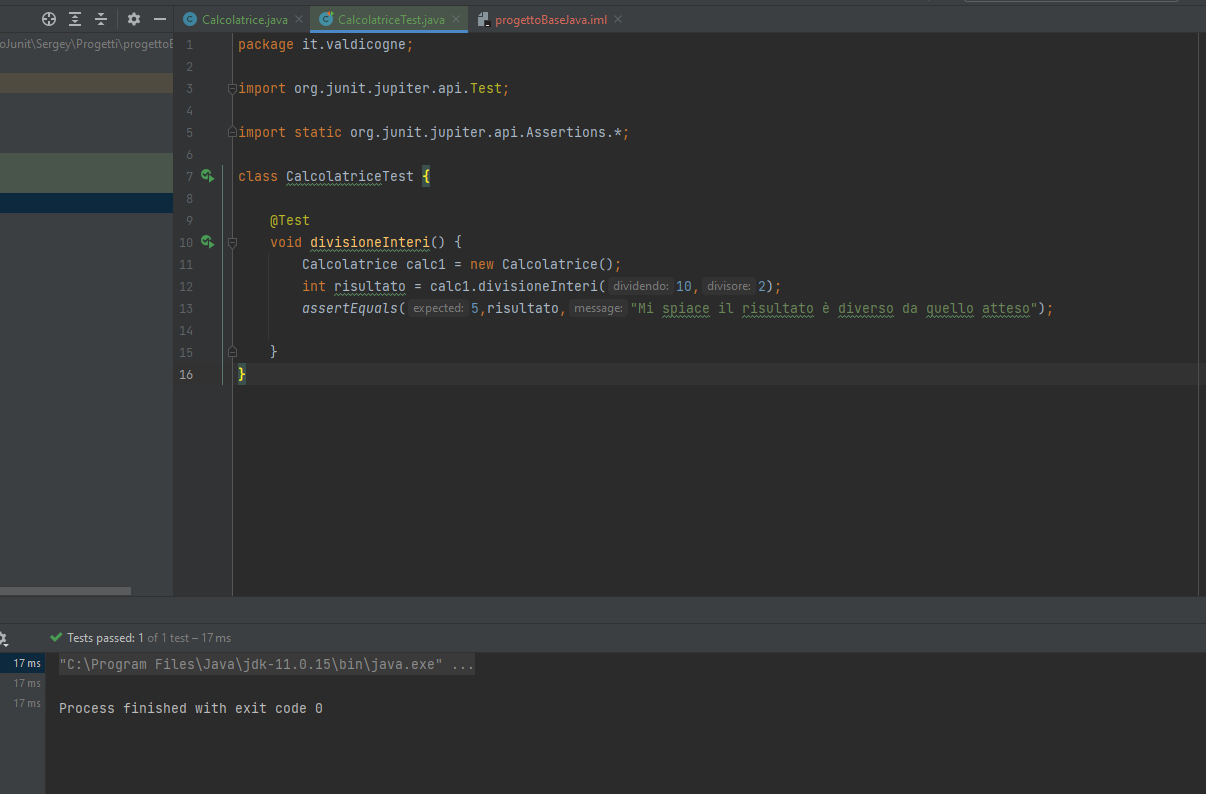
**ASSERTION**

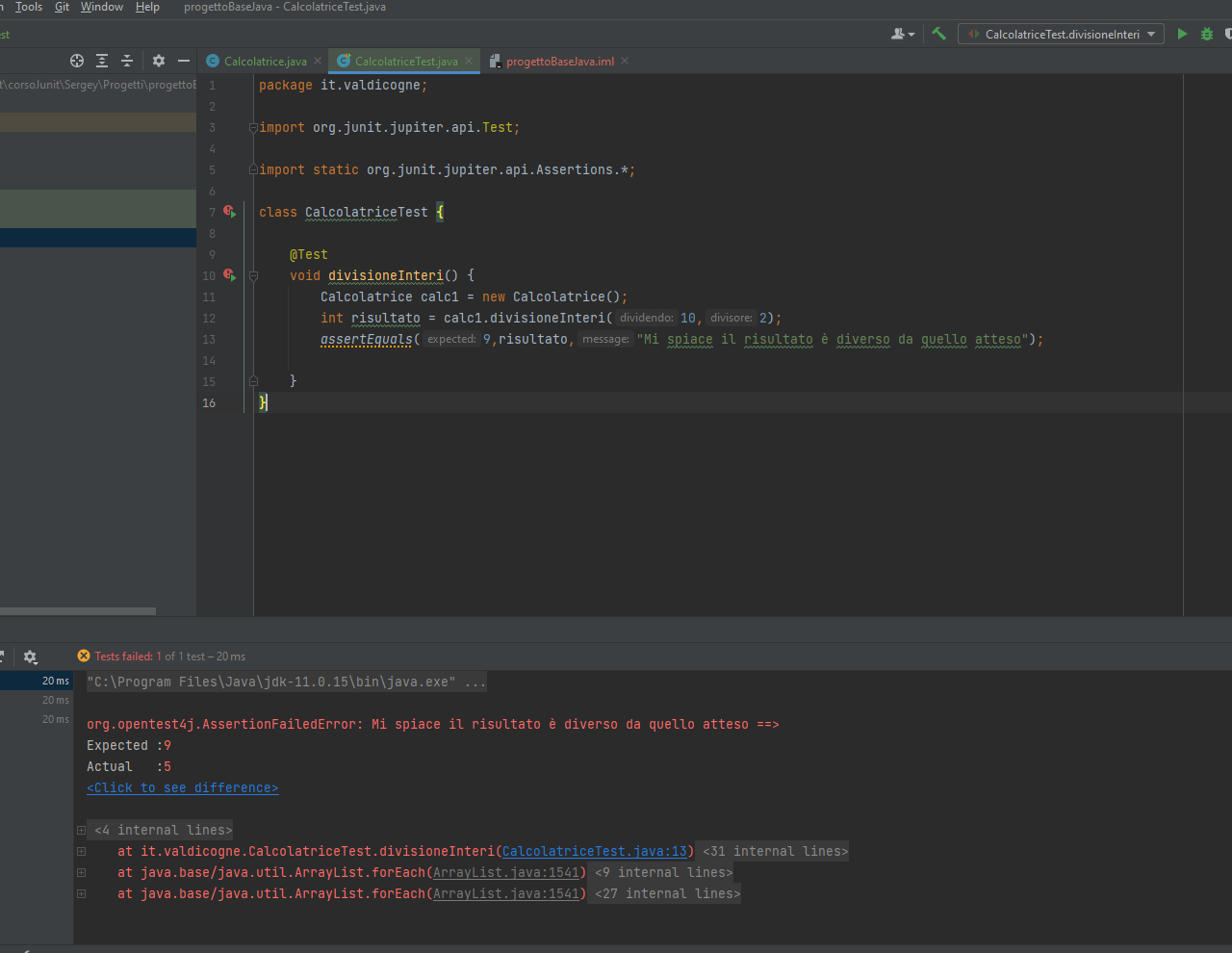
**Calcolatrice.java (src) **

**Calcolatrice.java (test)**

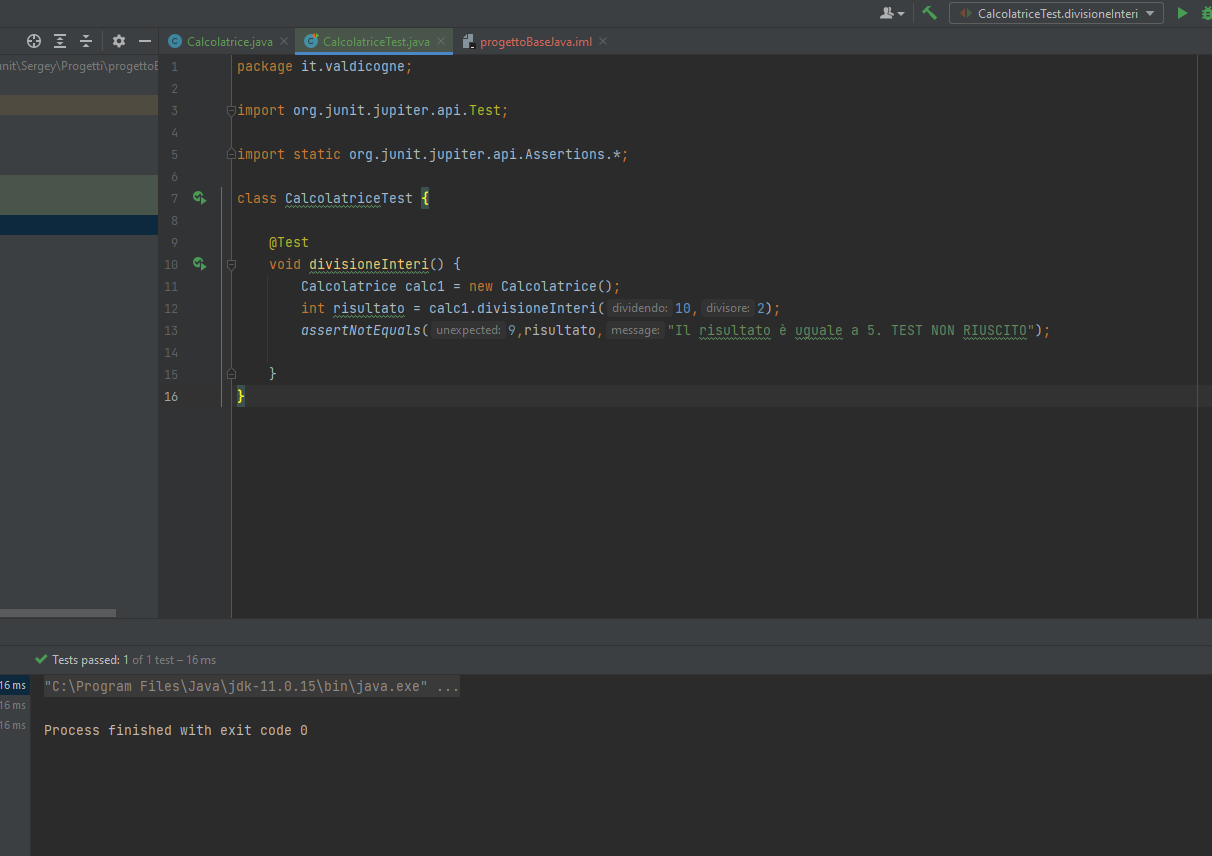
****

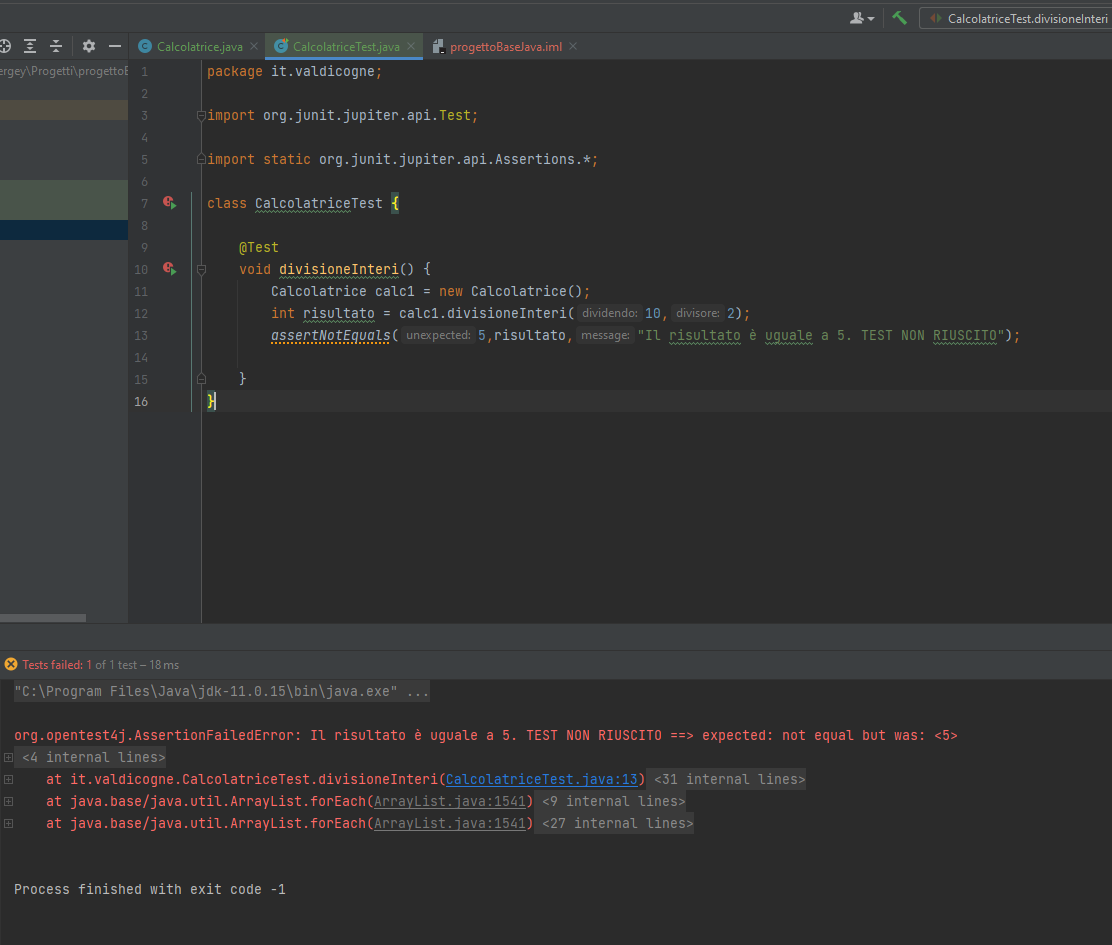
**assertEquals**

****

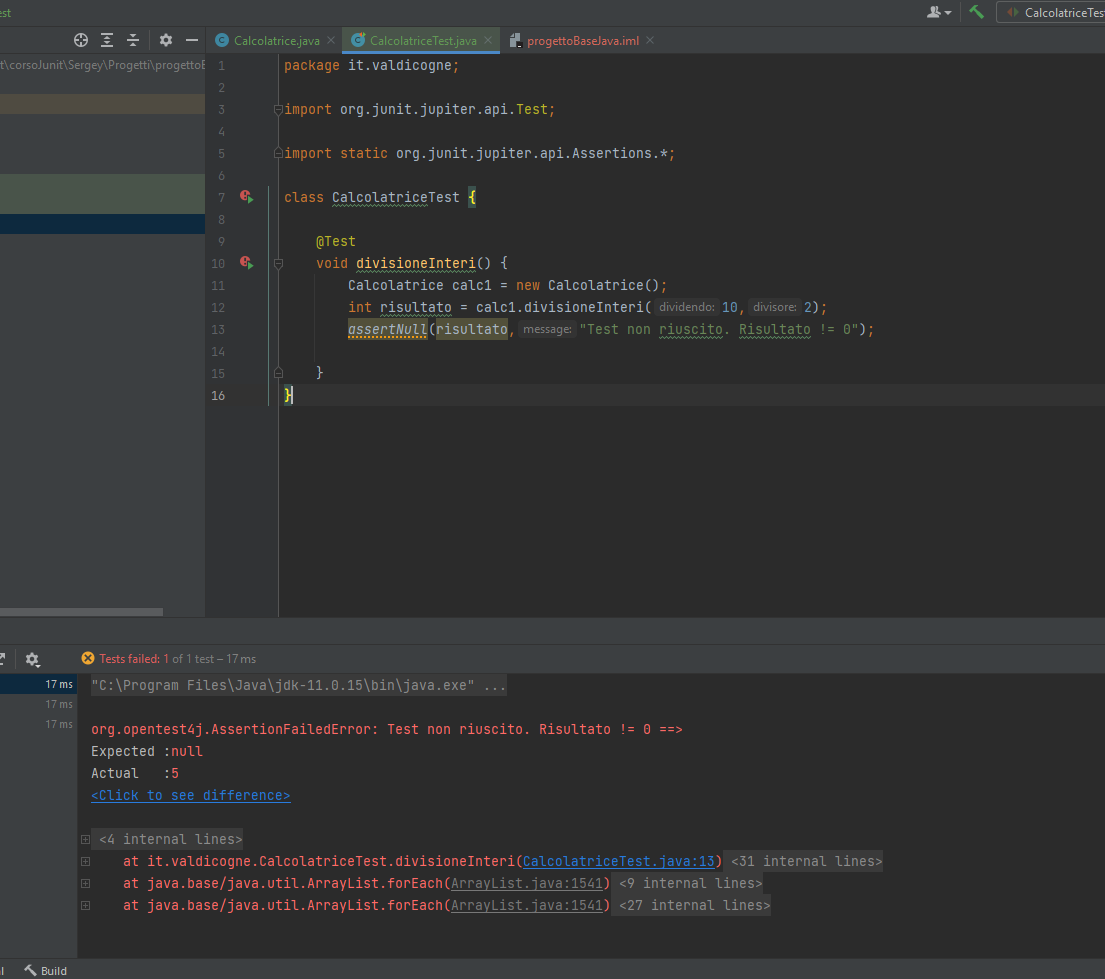
****

**assertNotEquals**

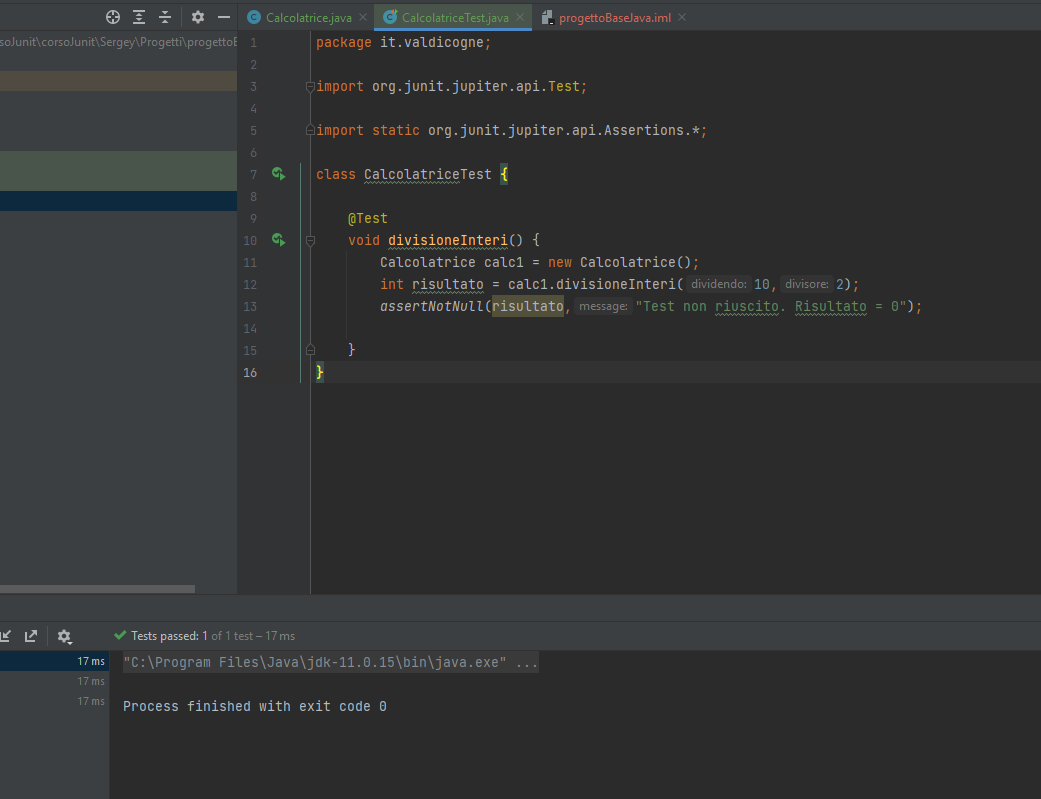
****

****

**assertNull**

****

**assertNotNull**

****

**assertEquals con lambda expression**

**Java src**

public int addizione(int n1,int n2){  
 return n1 + n2;  
}

**Java test**

@Test  
void addizione() {  
 Calcolatrice calc1 = new Calcolatrice();  
 int n1 = 9;  
 int n2 = 6;  
 int expectedResult= 10;  
 int risultato = calc1.addizione(n1,n2);  
 *assertEquals*(expectedResult,risultato,()->n1 + " + " + n2 + " è diverso da 10. Il test è fallito");  
  
}

**Il nome addizione per il metodo presente nalla classe calcolatrice.java in test non è utilizzato.**

**Il metodo si scrive in questo modo:**

@Test  
void testAddizione\_whenNineIsAddedToOne\_shouldReturnTen() {  
 Calcolatrice calc1 = new Calcolatrice();  
 int n1 = 9;  
 int n2 = 1;  
 int expectedResult= 10;  
 int risultato = calc1.addizione(n1,n2);  
 *assertEquals*(expectedResult,risultato,()->n1 + " + " + n2 + " è diverso da 10. Il test è fallito");  
  
}

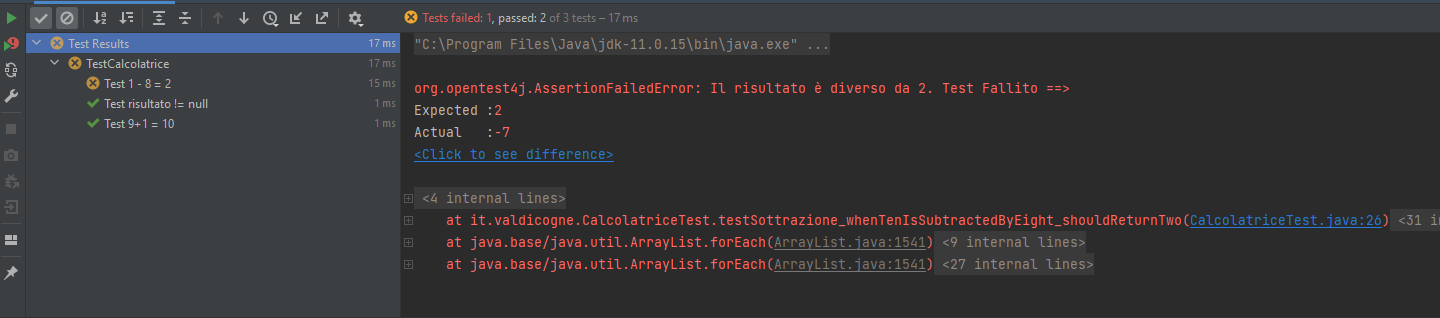
**@DisplayName**

**Calcolatrice.java (src)**

package it.valdicogne;  
  
public class Calcolatrice {  
  
 public int divisioneInteri(int dividendo,int divisore){  
 return dividendo/divisore;  
 }  
  
 public int sottrazione(int n1,int n2){  
 return n1 - n2;  
 }  
  
 public int addizione(int n1,int n2){  
 return n1 + n2;  
 }  
  
}

**Calcolatrice.java (test)**

package it.valdicogne;  
  
import com.sun.source.tree.CaseTree;  
import org.junit.jupiter.api.DisplayName;  
import org.junit.jupiter.api.Test;  
  
import static org.junit.jupiter.api.Assertions.\*;  
  
@DisplayName("TestCalcolatrice")  
class CalcolatriceTest {  
  
 @DisplayName("Test risultato != null")  
 @Test  
 void divisioneInteri() {  
 Calcolatrice calc1 = new Calcolatrice();  
 int risultato = calc1.divisioneInteri(10,2);  
 *assertNotNull*(risultato,"Test non riuscito. Risultato = 0");  
  
 }  
  
 @DisplayName("Test 1 - 8 = 2")  
 @Test  
 void testSottrazione\_whenTenIsSubtractedByEight\_shouldReturnTwo() {  
 Calcolatrice calc1 = new Calcolatrice();  
 int risultato = calc1.sottrazione(1,8);  
 *assertEquals*(2,risultato,"Il risultato è diverso da 2. Test Fallito");  
  
 }  
  
 @DisplayName("Test 9+1 = 10")  
 @Test  
 void testAddizione\_whenNineIsAddedToOne\_shouldReturnTen() {  
 Calcolatrice calc1 = new Calcolatrice();  
 int n1 = 9;  
 int n2 = 1;  
 int expectedResult= 10;  
 int risultato = calc1.addizione(n1,n2);  
 *assertEquals*(expectedResult,risultato,()->n1 + " + " + n2 + " è diverso da 10. Il test è fallito");  
 *assertEquals*(expectedResult,risultato,()->n1 + " + " + n2 + " è diverso da 10. Il test è fallito");  
  
 }  
}

****

**PATTERN AAA (Arrange, Act, Assert)**

**Finora abbiamo scritto I nostri metodi in modo abbastanza confusionario.**

**Il codice si divide in 3 parti:**

1. **Arrange**
2. **Act**
3. **Assert**
4. **Arrange**

**Sotto arrange si crea l’istanza della classe originale e le variabil.**

1. **Act**

**Sotto act si creano le variabili risultato**

1. **Assert**

**Sotto assert si utilizzano i metodi assert (assertEquals, assertNotEquals, fail etc)**

**Ricostruiamo quindi il nostro calcolatrice.java nel nostro test.**

package it.valdicogne;  
  
import com.sun.source.tree.CaseTree;  
import org.junit.jupiter.api.DisplayName;  
import org.junit.jupiter.api.Test;  
  
import static org.junit.jupiter.api.Assertions.\*;  
  
@DisplayName("TestCalcolatrice")  
class CalcolatriceTest {  
  
  
 @DisplayName("test numero1 + numero2 = risultatoAttesoGiustoSomma")  
 @Test  
 void testSomma\_whenTenSixtySummedByThirty\_shouldReturnNinety() {  
 // Arrange  
  
 Calcolatrice calc = new Calcolatrice();  
  
 int numero1 = 60;  
 int numero2 = 30;  
 int risultatoAttesoGiustoSomma = 90;  
 int risultatoAttesoErratoSomma = 23;  
  
  
 // Act  
  
 int somma = calc.addizione(numero1,numero2);  
  
  
 // Assert  
 *assertEquals*(risultatoAttesoErratoSomma , somma, "TestFailed");  
  
 }  
  
  
  
}