Toegepaste Informatica

1TX

2022-2023

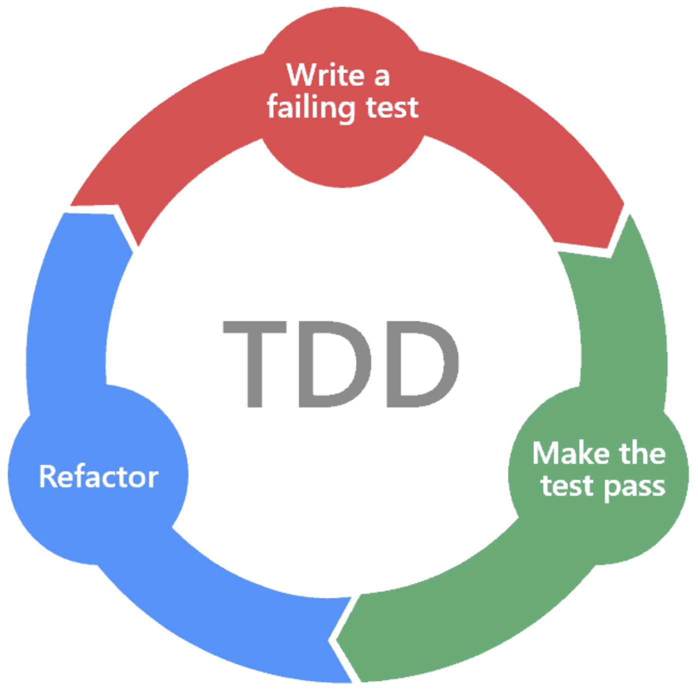
Back-End Development

Test Driven Development

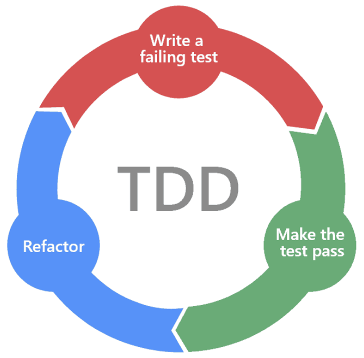
G. Jongen, J. Pieck, E. Steegmans, B. Van Impe

Test Driven Development (TDD)

- is a software development process relying on software requirements being converted to test cases before software is fully developed
    1. Write a failing test
    2. Make the test pass
    3. Refactor



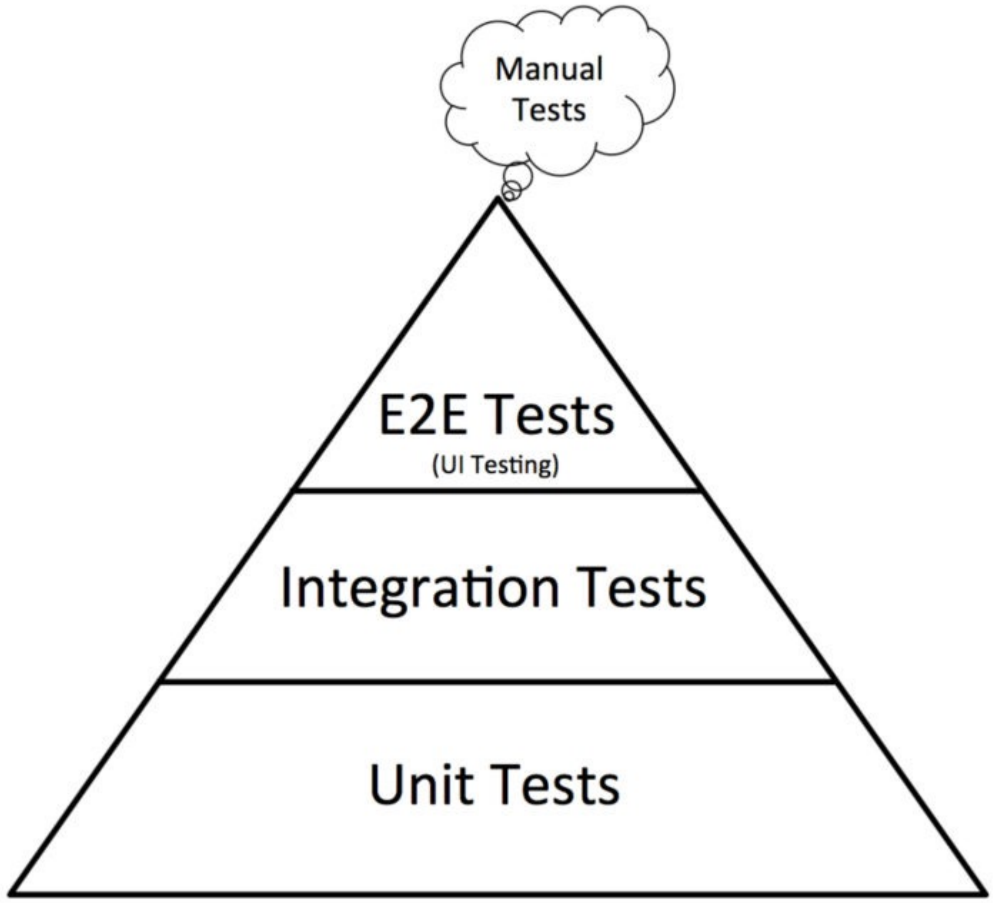
TDD@UCLL



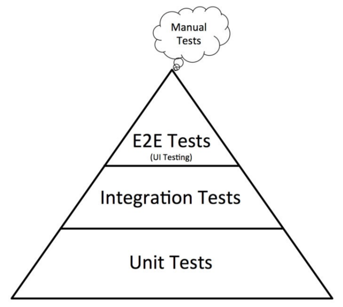
- PHASE 1
  - Students are given the test class and test methods
  - Students are writing code to make the test class and test methods pass

Testing Pyramid

- Unit tests
- Integration tests
- E2E tests
- Manual tests



Unit Testing



- is the process of testing small isolated portions of a software application called units
- you only focus on that little part or unit in your unit test
  - unit is e.g. a class, a method, a happy case, an unhappy case, ...
  - should be very fast
  - should only test 1 functionality/scenario

Unit test

- Test method
    - Tests a method of a class
        - Happy or unhappy case
    - 3 parts
        - Given: the context or input
        - When: the action (or method under test)
        - Then: the expected outcome
    - Name of test method should be human readable
        - given..._when..._then...

JUnit 5 for Java

- pom.xml

```xml
<dependency>
  <groupId>org.junit.jupiter</groupId>
  <artifactId>junit-jupiter-engine</artifactId>
  <version>5.6.2</version>
  <scope>test</scope>
</dependency>
```

STEP 0 – Read the test method
Constructor – Happy case

```
//given
private String validNameElke = "Elke";
private int validAgeElke = 44;

//constructor
//happy case
@Test
void givenValidValues_whenCreatingUser_thenUserIsCreatedWithTheseValues() {
  //when
  User elke = new User(validNameElke, validAgeElke);

  //then
  assertNotNull(elke);
  assertEquals(validNameElke, elke.getName());
  assertEquals(validAgeElke, elke.getAge());
  assertEquals(0, elke.countYearsOfMembership());
}
```

@Test

- Indicates that it is a test method
- It is a method that can be executed
  - It is like a little main method that you can run/execute ...

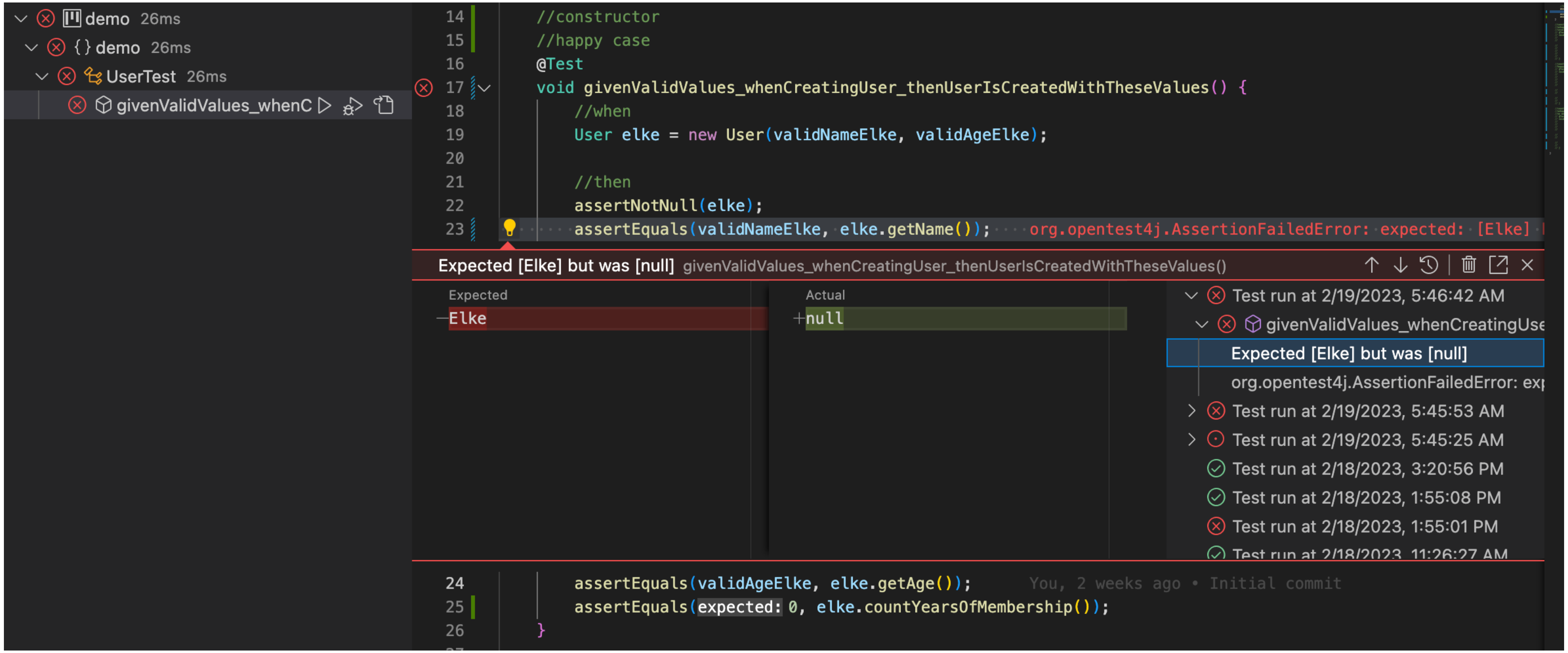
```
@Test
void givenValidValues_whenCreatingUser_thenUserIsCreatedWithTheseValues() {
```

assert methods

- Methods to test the state of objects
  - boolean assertEquals(expected, actual)
    - assertEquals(44, elke.getAge())
      - true if the instance variable age of the elke object has the value 44, false otherwise
  - boolean assertNotNull(object)
    - assertNotNull(elke)
      - true if the object with name elke is created with the default values in the instance variables, false otherwhise

```
assertNotNull(elke);
assertEquals(validAgeElke, elke.getAge());
```

STEP 1 – RUN THE TESTMETHOD



demo 26ms
{ } demo 26ms
UserTest 26ms
givenValidValues_whenC

```java
    //constructor
    //happy case
    @Test
    void givenValidValues_whenCreatingUser_thenUserIsCreatedWithTheseValues() {
        //when
        User elke = new User(validNameElke, validAgeElke);

        //then
        assertNotNull(elke);
        assertEquals(validNameElke, elke.getName());    org.opentest4j.AssertionFailedError: expected: [Elke]
```

Expected [Elke] but was [null] givenValidValues_whenCreatingUser_thenUserIsCreatedWithTheseValues()

Expected
−Elke

Actual
+null

Test run at 2/19/2023, 5:46:42 AM
givenValidValues_whenCreatingUse
Expected [Elke] but was [null]
org.opentest4j.AssertionFailedError: exp
Test run at 2/19/2023, 5:45:53 AM
Test run at 2/19/2023, 5:45:25 AM
Test run at 2/18/2023, 3:20:56 PM
Test run at 2/18/2023, 1:55:08 PM
Test run at 2/18/2023, 1:55:01 PM
Test run at 2/18/2023, 11:26:27 AM

```java
        assertEquals(validAgeElke, elke.getAge());      You, 2 weeks ago • Initial commit
        assertEquals(expected: 0, elke.countYearsOfMembership());
    }
```

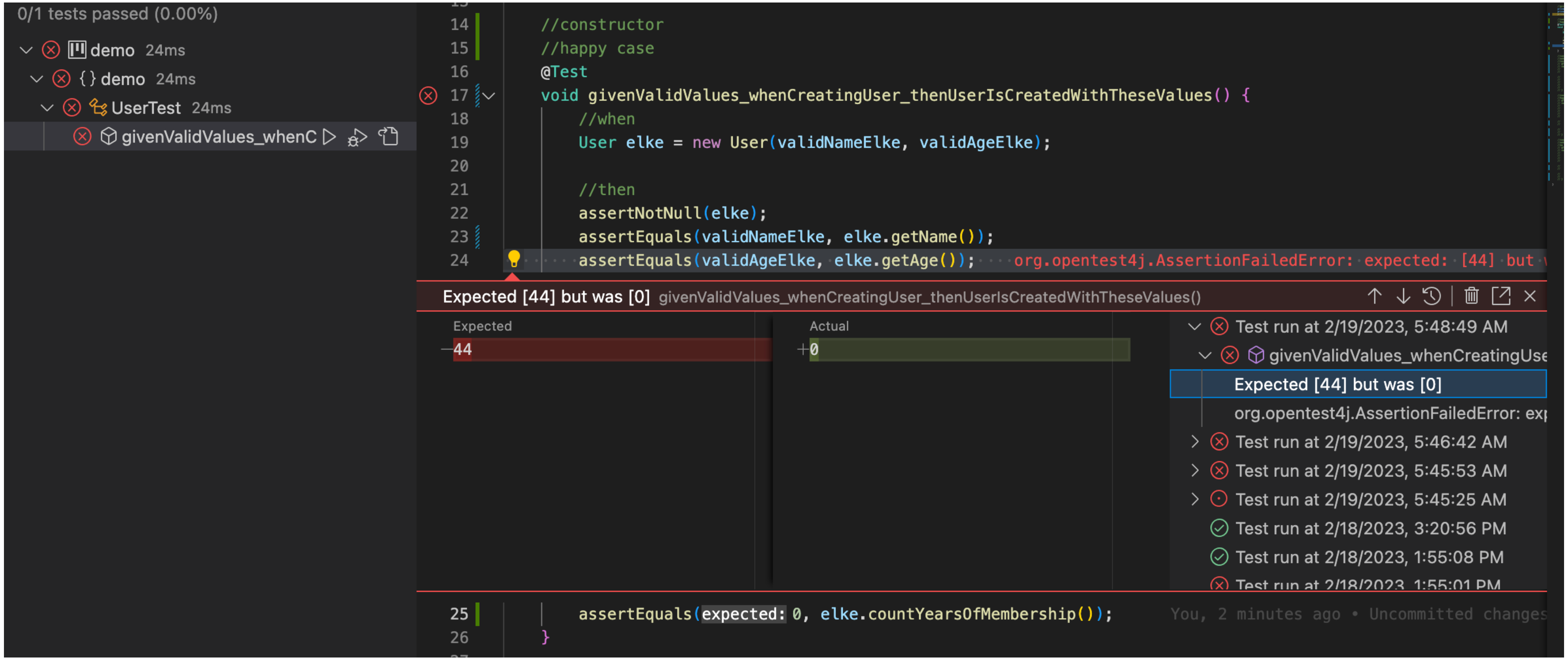
STEP 2 – WRITE CODE FOR THE METHOD UNDER TEST

```
public class User {

  private String name;
  private int age;
  private List<Integer> membershipYears;


  public User(String name, int age) {
    this.name = name;
  }
```

STEP 1 – RE-RUN THE TESTMETHOD



0/1 tests passed (0.00%)

- demo 24ms
  - { } demo 24ms
    - UserTest 24ms
      - givenValidValues_whenC

```java
    //constructor
    //happy case
    @Test
    void givenValidValues_whenCreatingUser_thenUserIsCreatedWithTheseValues() {
        //when
        User elke = new User(validNameElke, validAgeElke);

        //then
        assertNotNull(elke);
        assertEquals(validNameElke, elke.getName());
        assertEquals(validAgeElke, elke.getAge());    org.opentest4j.AssertionFailedError: expected: [44] but
```

Expected [44] but was [0] givenValidValues_whenCreatingUser_thenUserIsCreatedWithTheseValues()

Expected	Actual
−44	+0

- Test run at 2/19/2023, 5:48:49 AM
  - givenValidValues_whenCreatingUse
    - Expected [44] but was [0]
    - org.opentest4j.AssertionFailedError: exp
- Test run at 2/19/2023, 5:46:42 AM
- Test run at 2/19/2023, 5:45:53 AM
- Test run at 2/19/2023, 5:45:25 AM
- Test run at 2/18/2023, 3:20:56 PM
- Test run at 2/18/2023, 1:55:08 PM
- Test run at 2/18/2023, 1:55:01 PM

```java
        assertEquals(expected: 0, elke.countYearsOfMembership());    You, 2 minutes ago • Uncommitted changes
    }
```

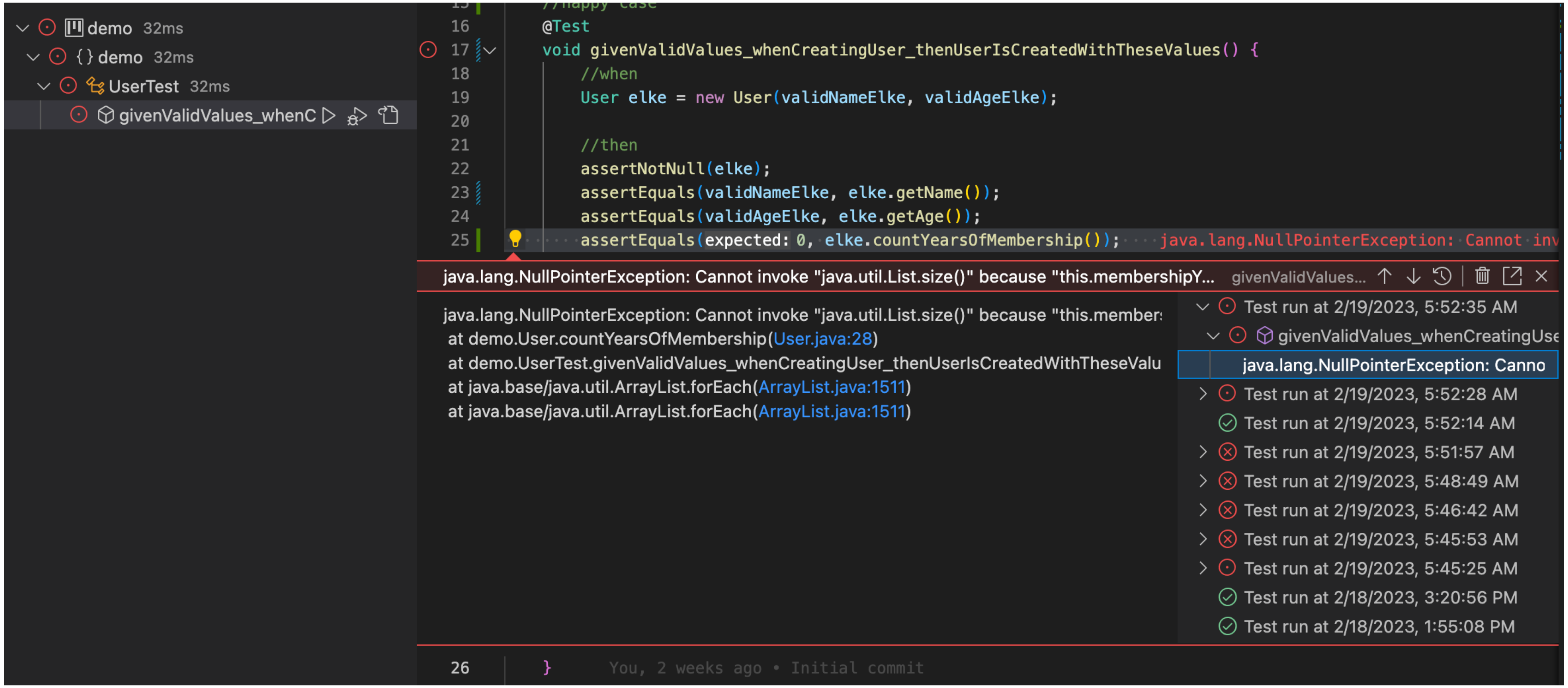
STEP 2 – WRITE CODE FOR THE METHOD UNDER TEST UNTILL TEST PASSES

```java
public class User {

  private String name;
  private int age;
  private List<Integer> membershipYears;


  public User(String name, int age) {
    this.name = name;
    this.age = age;
  }
```

STEP 1 – RE-RUN THE TESTMETHOD



demo 32ms
{} demo 32ms
UserTest 32ms
givenValidValues_whenC

```java
    //happy case
    @Test
    void givenValidValues_whenCreatingUser_thenUserIsCreatedWithTheseValues() {
        //when
        User elke = new User(validNameElke, validAgeElke);

        //then
        assertNotNull(elke);
        assertEquals(validNameElke, elke.getName());
        assertEquals(validAgeElke, elke.getAge());
        assertEquals(expected: 0, elke.countYearsOfMembership());    java.lang.NullPointerException: Cannot inv
```

java.lang.NullPointerException: Cannot invoke "java.util.List.size()" because "this.membershipY... givenValidValues...

```
java.lang.NullPointerException: Cannot invoke "java.util.List.size()" because "this.member:
 at demo.User.countYearsOfMembership(User.java:28)
 at demo.UserTest.givenValidValues_whenCreatingUser_thenUserIsCreatedWithTheseValu
 at java.base/java.util.ArrayList.forEach(ArrayList.java:1511)
 at java.base/java.util.ArrayList.forEach(ArrayList.java:1511)
```

Test run at 2/19/2023, 5:52:35 AM
givenValidValues_whenCreatingUse
java.lang.NullPointerException: Canno
Test run at 2/19/2023, 5:52:28 AM
Test run at 2/19/2023, 5:52:14 AM
Test run at 2/19/2023, 5:51:57 AM
Test run at 2/19/2023, 5:48:49 AM
Test run at 2/19/2023, 5:46:42 AM
Test run at 2/19/2023, 5:45:53 AM
Test run at 2/19/2023, 5:45:25 AM
Test run at 2/18/2023, 3:20:56 PM
Test run at 2/18/2023, 1:55:08 PM

```java
    }       You, 2 weeks ago • Initial commit
```

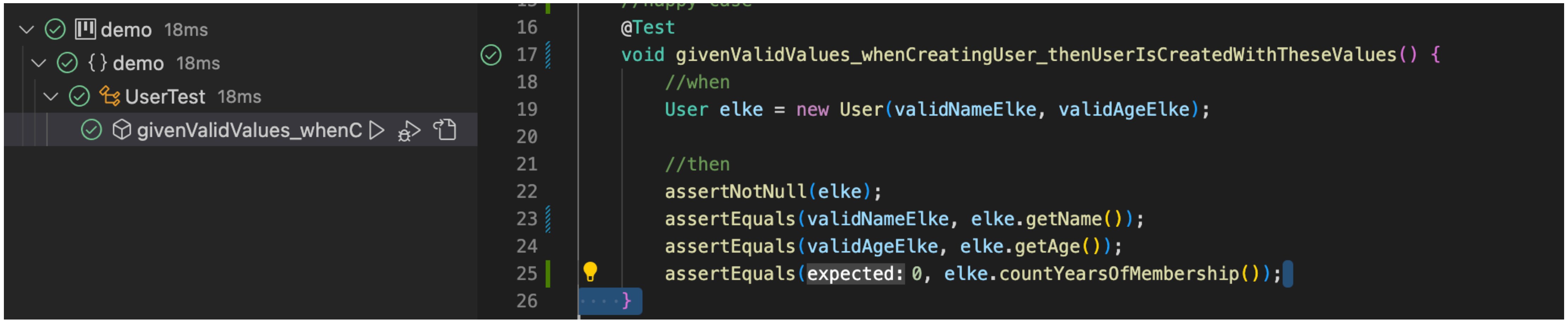
STEP 2 – WRITE CODE FOR THE METHOD UNDER TEST UNTILL TEST PASSES

```java
public class User {

  private String name;
  private int age;
  private List<Integer> membershipYears = new ArrayList<Integer>();


  public User(String name, int age) {
    this.name = name;
    this.age = age;
  }
```

STEP 1 – RE-RUN THE TESTMETHOD



```
    //happy case
    @Test
    void givenValidValues_whenCreatingUser_thenUserIsCreatedWithTheseValues() {
        //when
        User elke = new User(validNameElke, validAgeElke);

        //then
        assertNotNull(elke);
        assertEquals(validNameElke, elke.getName());
        assertEquals(validAgeElke, elke.getAge());
        assertEquals(expected: 0, elke.countYearsOfMembership());
    }
```

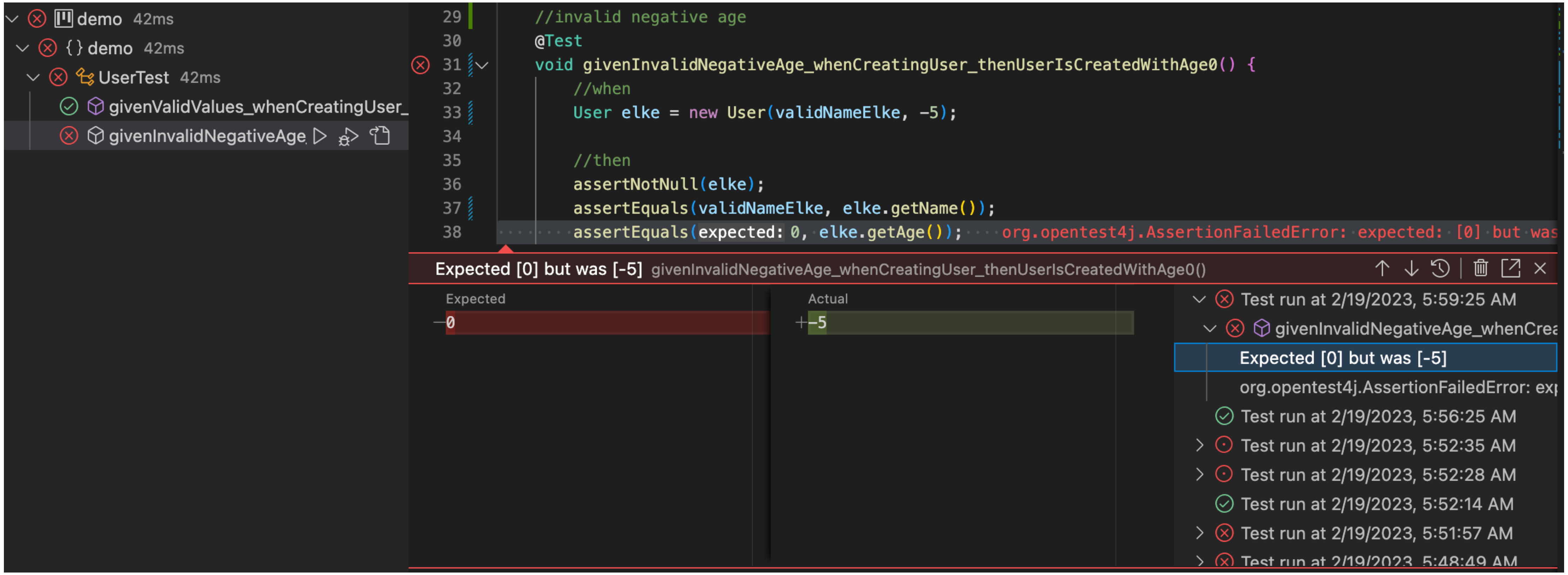
STEP 0 – Read the test method Constructor – Unhappy case

```
//given
private String validNameElke = "Elke";

//constructor
//unhappy case
//invalid negative age
@Test
void givenInvalidNegativeAge_whenCreatingUser_thenUserIsCreatedWithAge0() {
  //when
  User elke = new User(validNameElke, -5);

  //then
  assertNotNull(elke);
  assertEquals(validNameElke, elke.getName());
  assertEquals(0, elke.getAge());
  assertEquals(0, elke.countYearsOfMembership());
}
```

STEP 1 – RUN THE TESTMETHOD



demo 42ms
{ } demo 42ms
UserTest 42ms
givenValidValues_whenCreatingUser_
givenInvalidNegativeAge

```java
//invalid negative age
@Test
void givenInvalidNegativeAge_whenCreatingUser_thenUserIsCreatedWithAge0() {
    //when
    User elke = new User(validNameElke, -5);

    //then
    assertNotNull(elke);
    assertEquals(validNameElke, elke.getName());
    assertEquals(expected: 0, elke.getAge());    org.opentest4j.AssertionFailedError: expected: [0] but was
```

Expected [0] but was [-5] givenInvalidNegativeAge_whenCreatingUser_thenUserIsCreatedWithAge0()

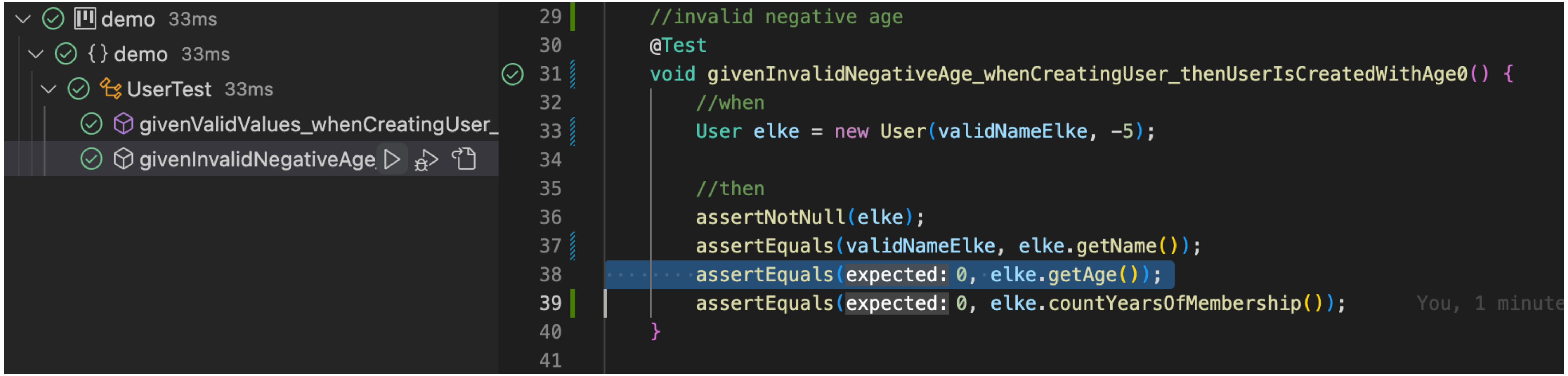
Expected	Actual
−0	+-5

Test run at 2/19/2023, 5:59:25 AM
givenInvalidNegativeAge_whenCrea
Expected [0] but was [-5]
org.opentest4j.AssertionFailedError: ex
Test run at 2/19/2023, 5:56:25 AM
Test run at 2/19/2023, 5:52:35 AM
Test run at 2/19/2023, 5:52:28 AM
Test run at 2/19/2023, 5:52:14 AM
Test run at 2/19/2023, 5:51:57 AM
Test run at 2/19/2023, 5:48:49 AM

STEP 2 – WRITE CODE FOR THE METHOD UNDER TEST

```
public User(String name, int age) {
  this.name = name;
  if (age >= 0)
    this.age = age;
}
```

STEP 3 – RE-RUN THE TESTMETHOD



```
demo 33ms
  {} demo 33ms
    UserTest 33ms
      givenValidValues_whenCreatingUser_
      givenInvalidNegativeAge
```

```
    //invalid negative age
    @Test
    void givenInvalidNegativeAge_whenCreatingUser_thenUserIsCreatedWithAge0() {
        //when
        User elke = new User(validNameElke, -5);

        //then
        assertNotNull(elke);
        assertEquals(validNameElke, elke.getName());
        assertEquals(expected: 0, elke.getAge());
        assertEquals(expected: 0, elke.countYearsOfMembership());      You, 1 minute
    }
```

IMPORTANT

- Always re-run your green coloured test methods
  - Because you could have broken other parts of your code at some point

Unit test

- Test class
  - Tests all methods of that class
    - All happy and all unhappy cases are tested in separate test methods

References

- https://marsner.com/blog/why-test-driven-development-tdd/