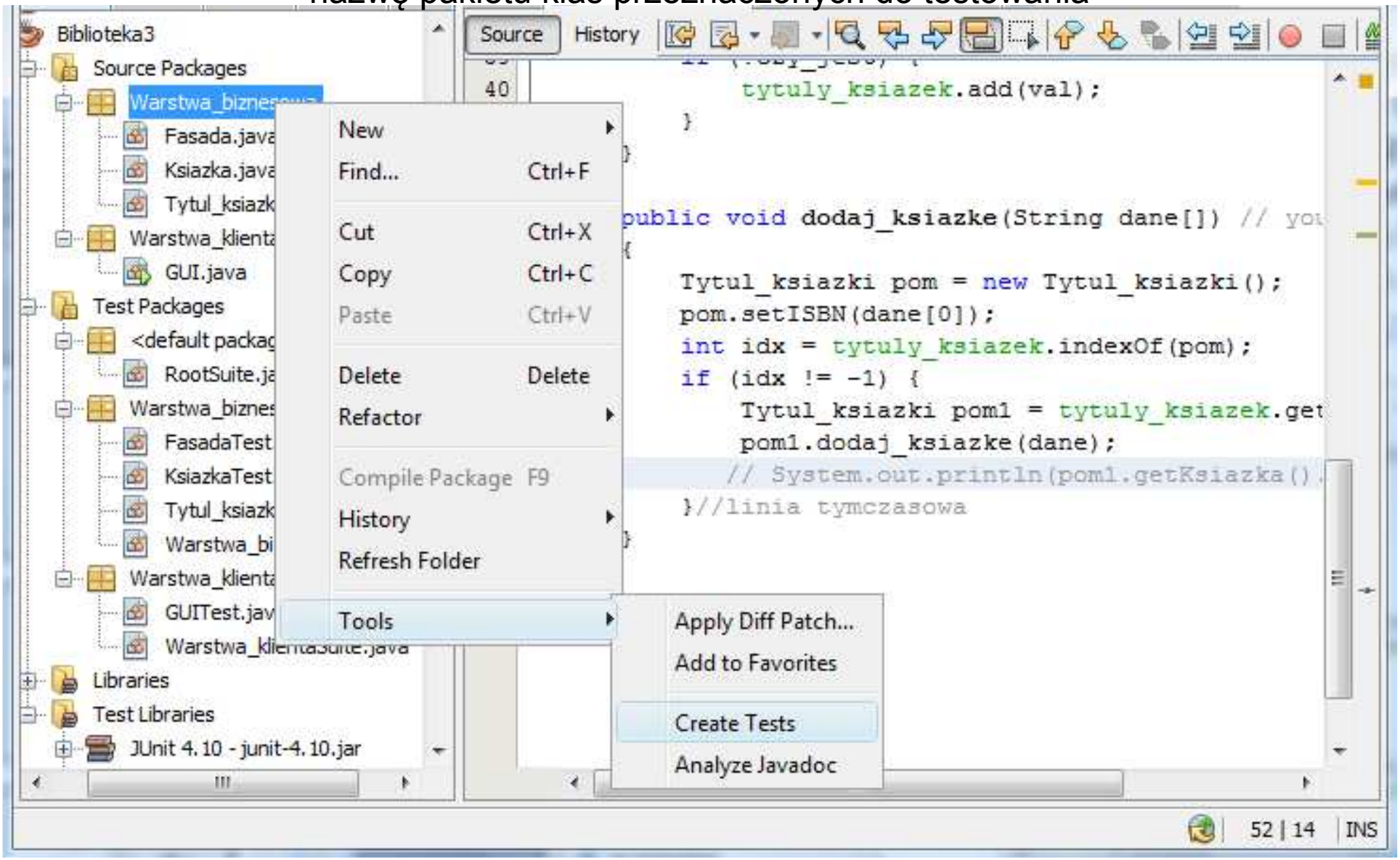
# Testy jednostkowe - zastosowanie oprogramowania JUNIT 4.0

http://www.junit.org/

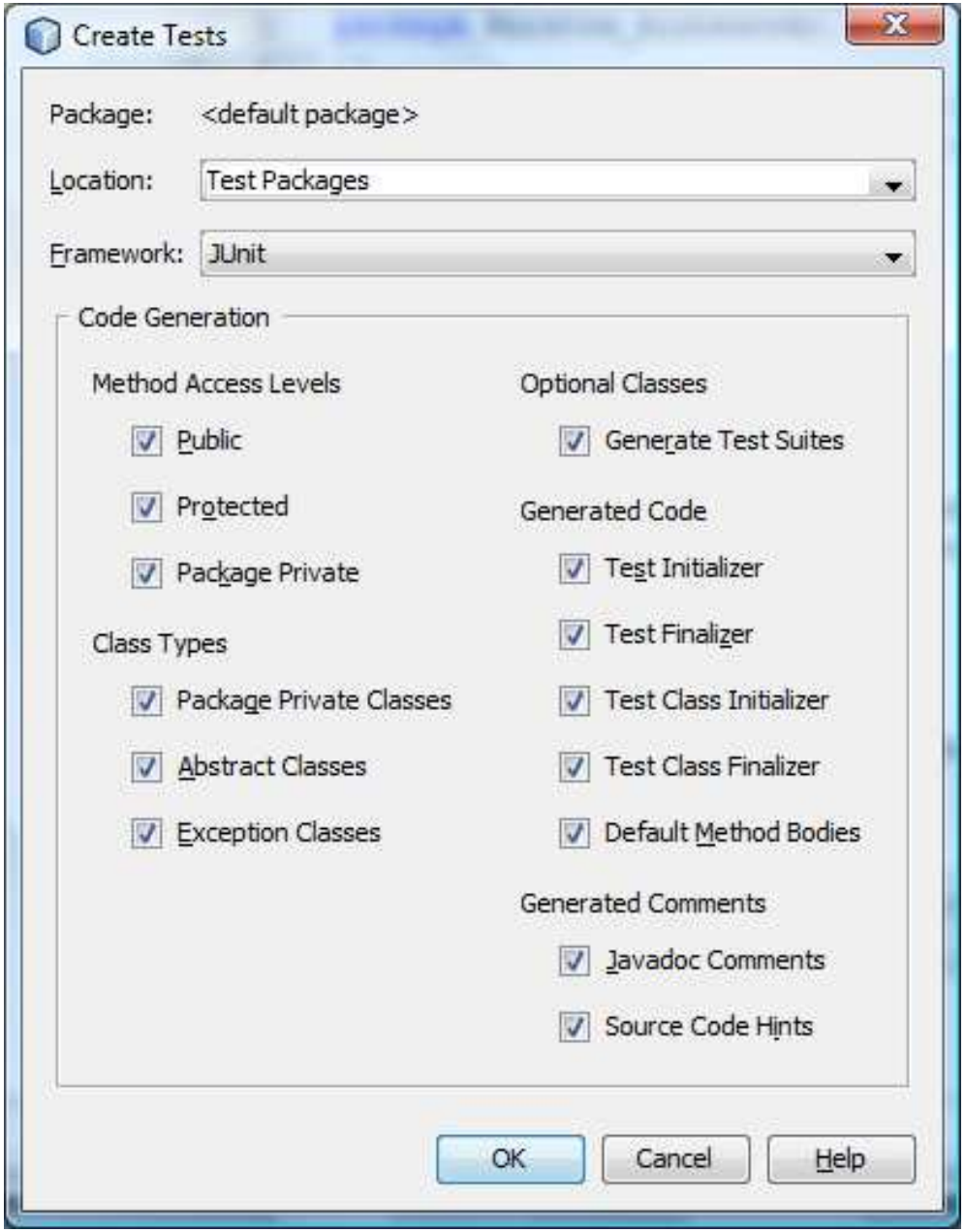
Zofia Kruczkiewicz

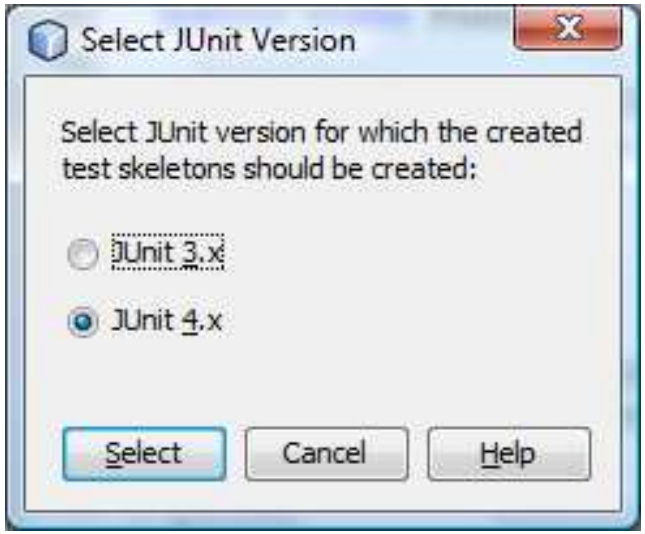
1. Aby utworzyć **test dla jednej klasy**, należy kliknąć prawym przyciskiem myszy w oknie *Projects* na wybraną klasę i wybrać *Tools> Create Tests (Ctrl-Shift-U),* aby otworzyć okno dialogowe Create Tests .

Utworzenie **zestawu testów** wykonuje się podobnie klikając prawym klawiszem na nazwę pakietu klas przeznaczonych do testowania

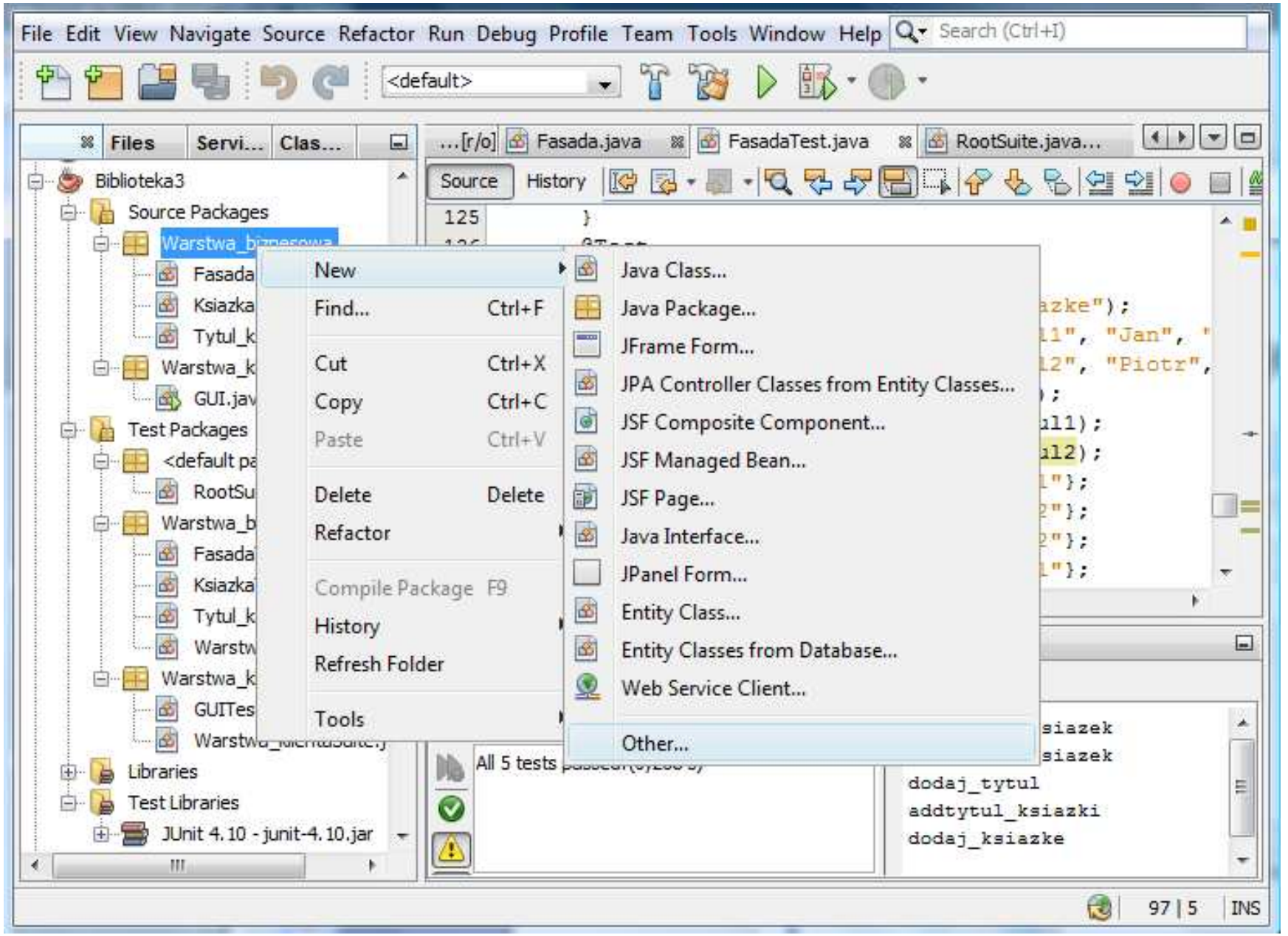


Należy wybrać framework typu **junit**. Na wybranym formularzu należy zaznaczyć właściwe opcje testowania.

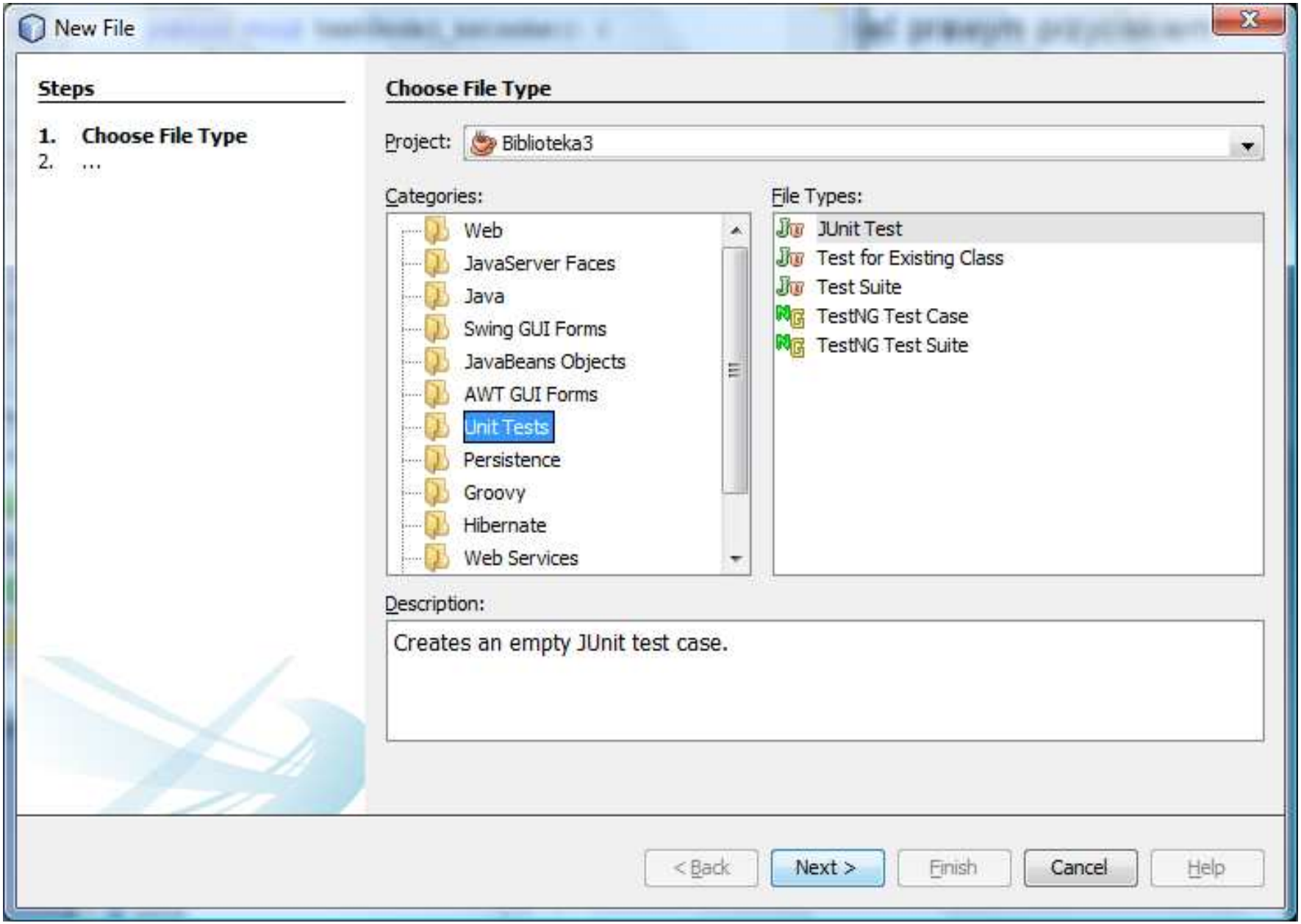


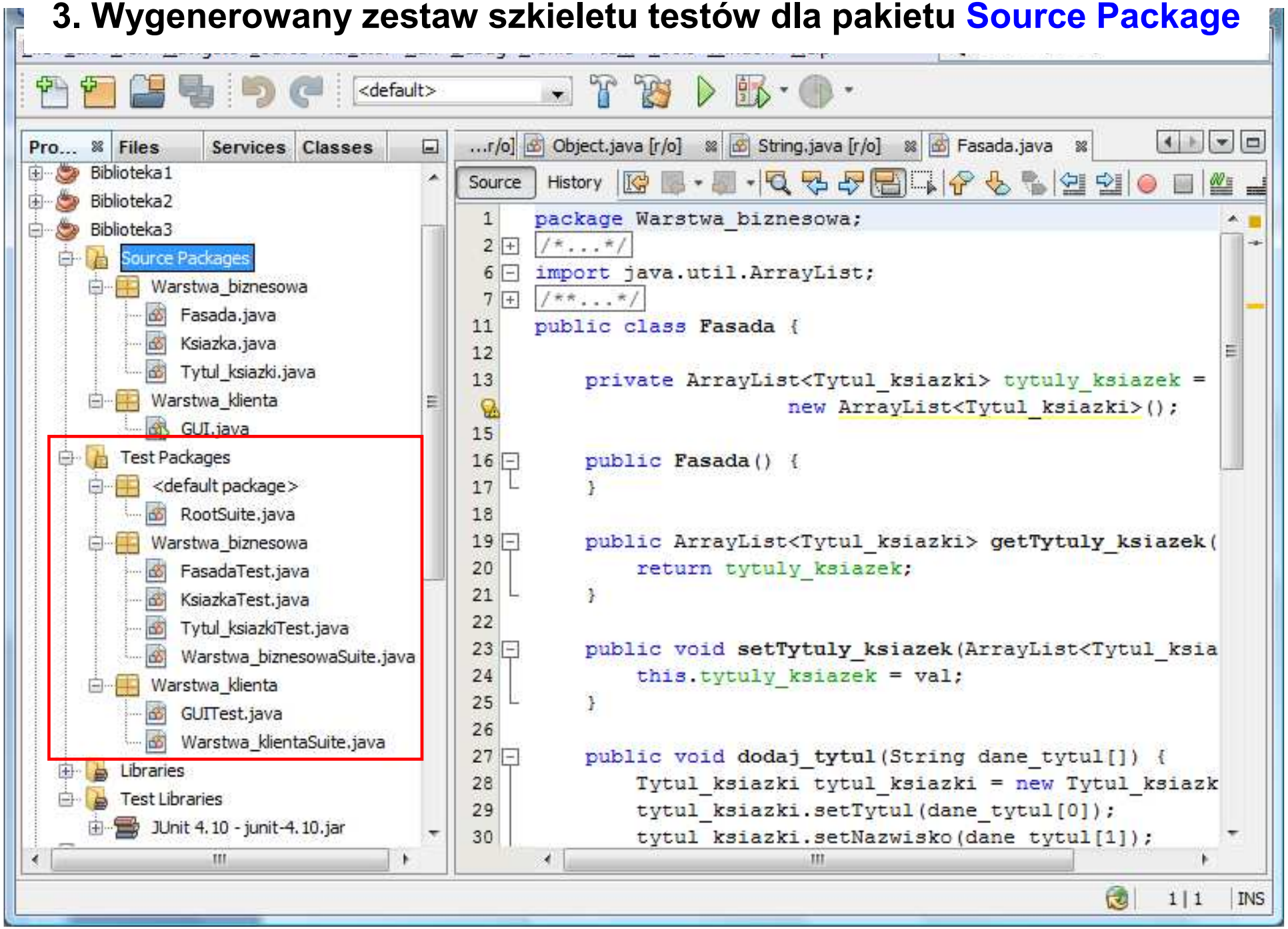
## 2. Klasy testowe można utworzyć drugim sposobem wykorzystując File Wizard – wybór pozycji Other…



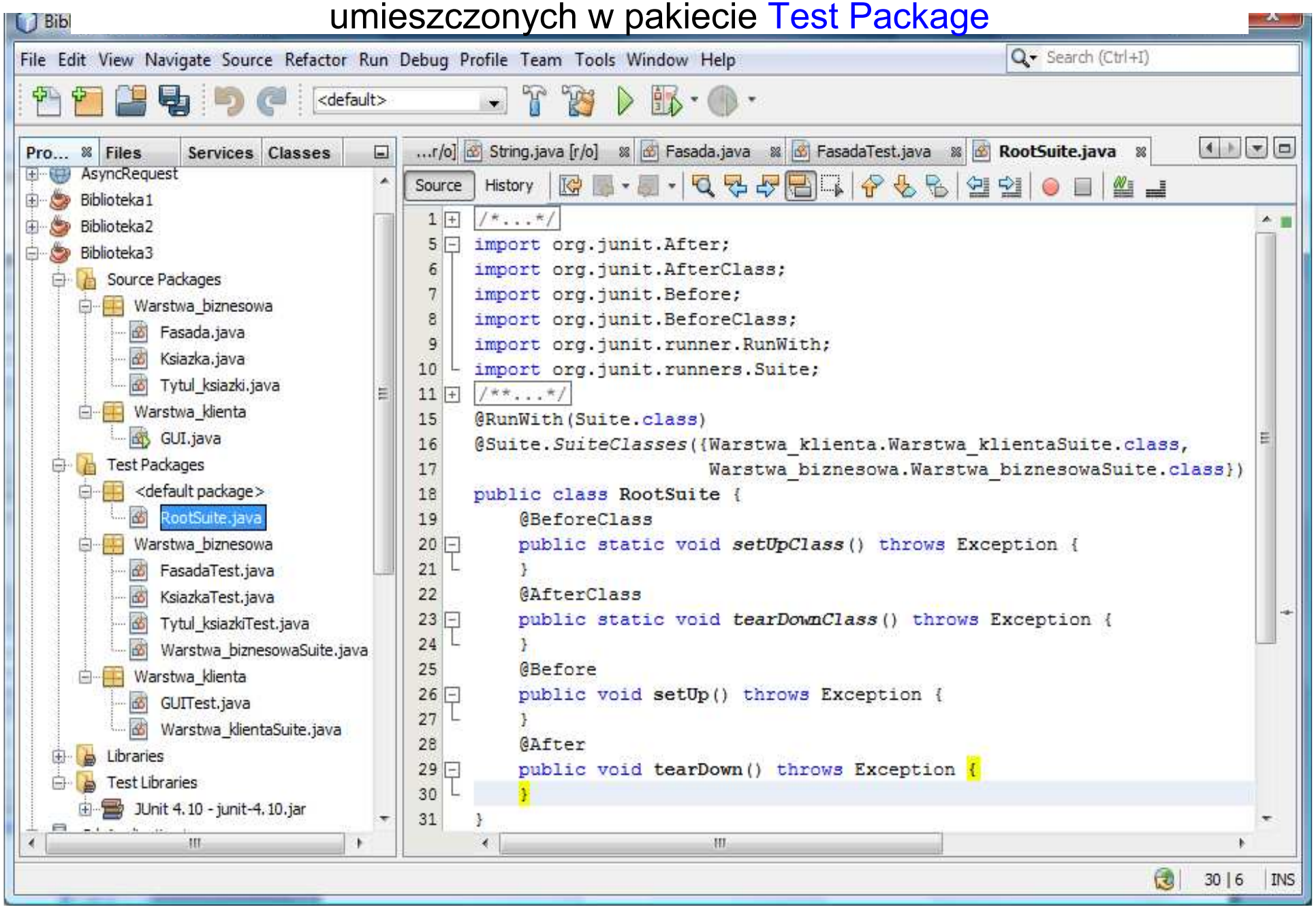
Wybór kategorii pliku:**Unit Test** oraz typ pliku testowego (prawa część formularza)



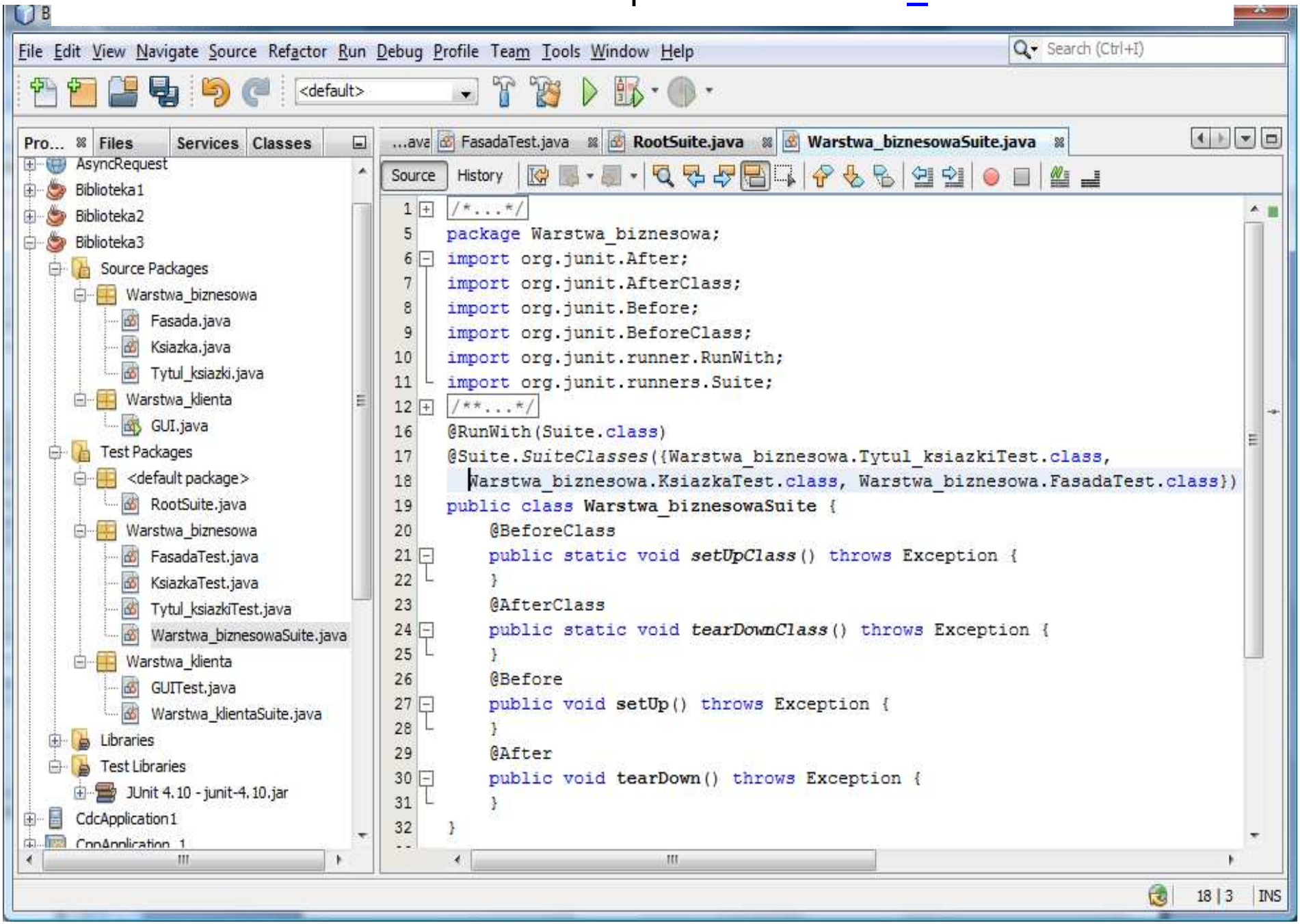
# 3. Wygenerowany zestaw szkieletu testów dla pakietu Source Package



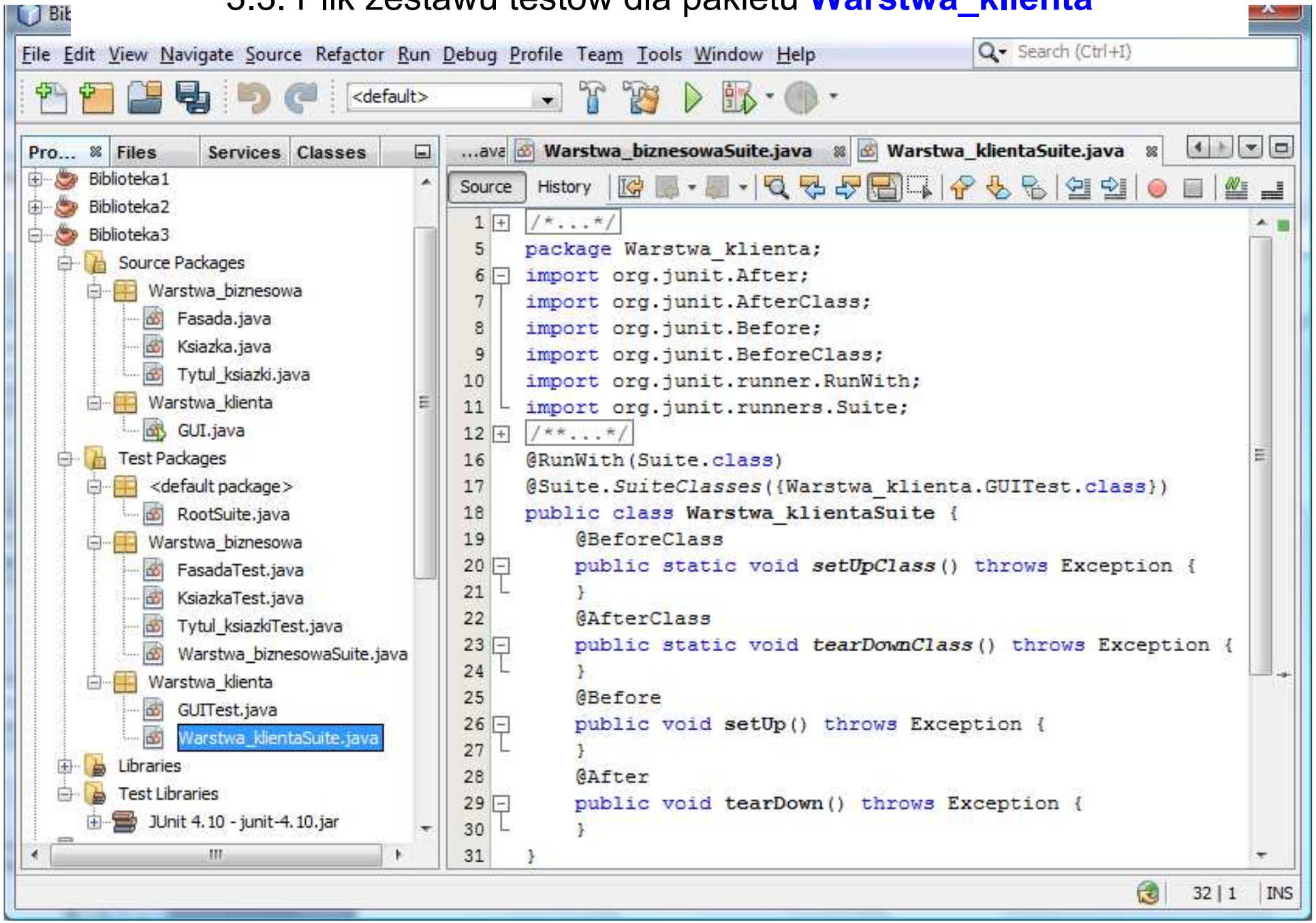
# 3.1. Plik zestawu wszystkich testów pakietu Source Package umieszczonych w pakiecie Test Package



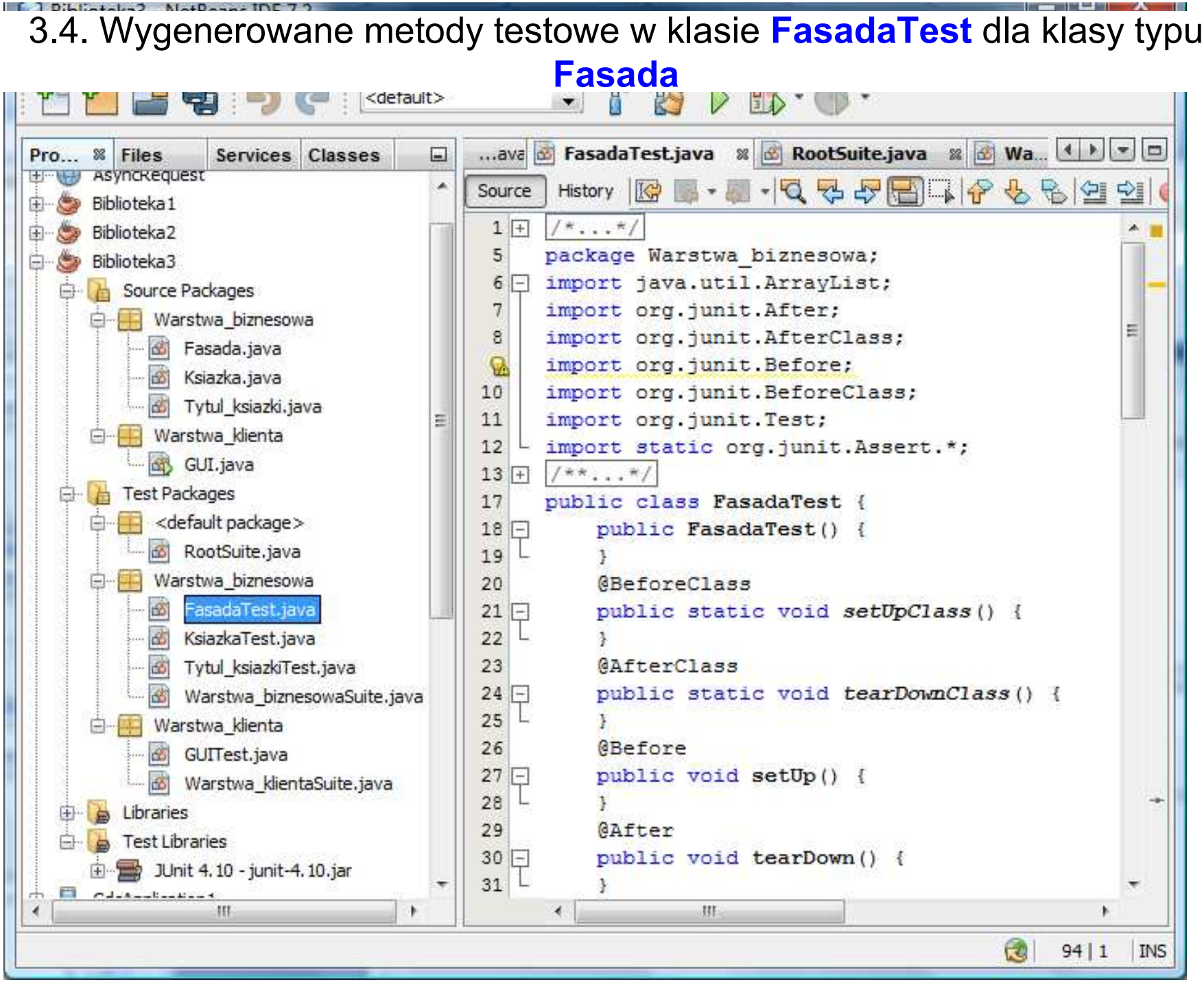
# 3.2. Plik zestawu testów dla pakietu **Warstwa_biznesowa**



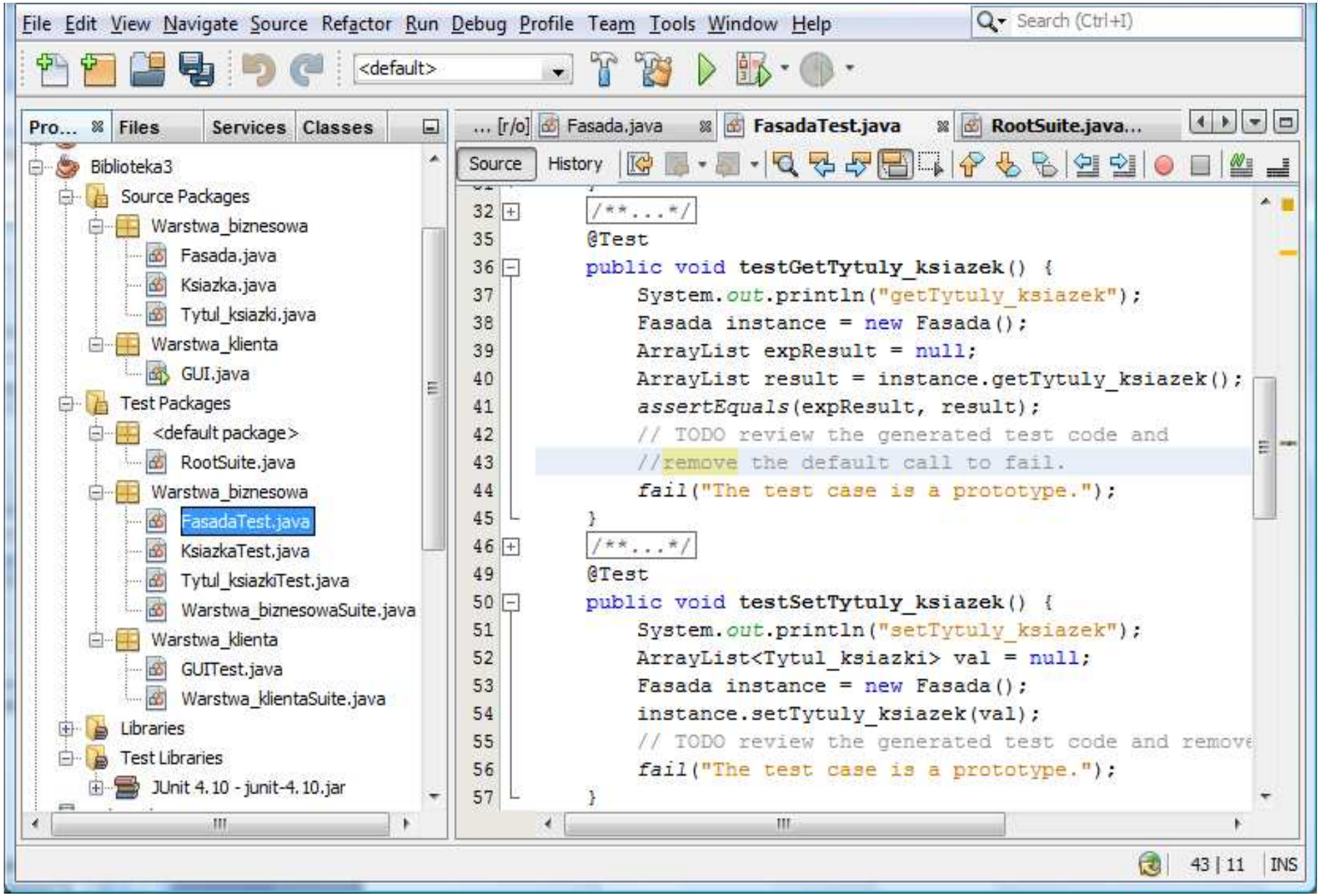
# 3.3. Plik zestawu testów dla pakietu **Warstwa_klienta**



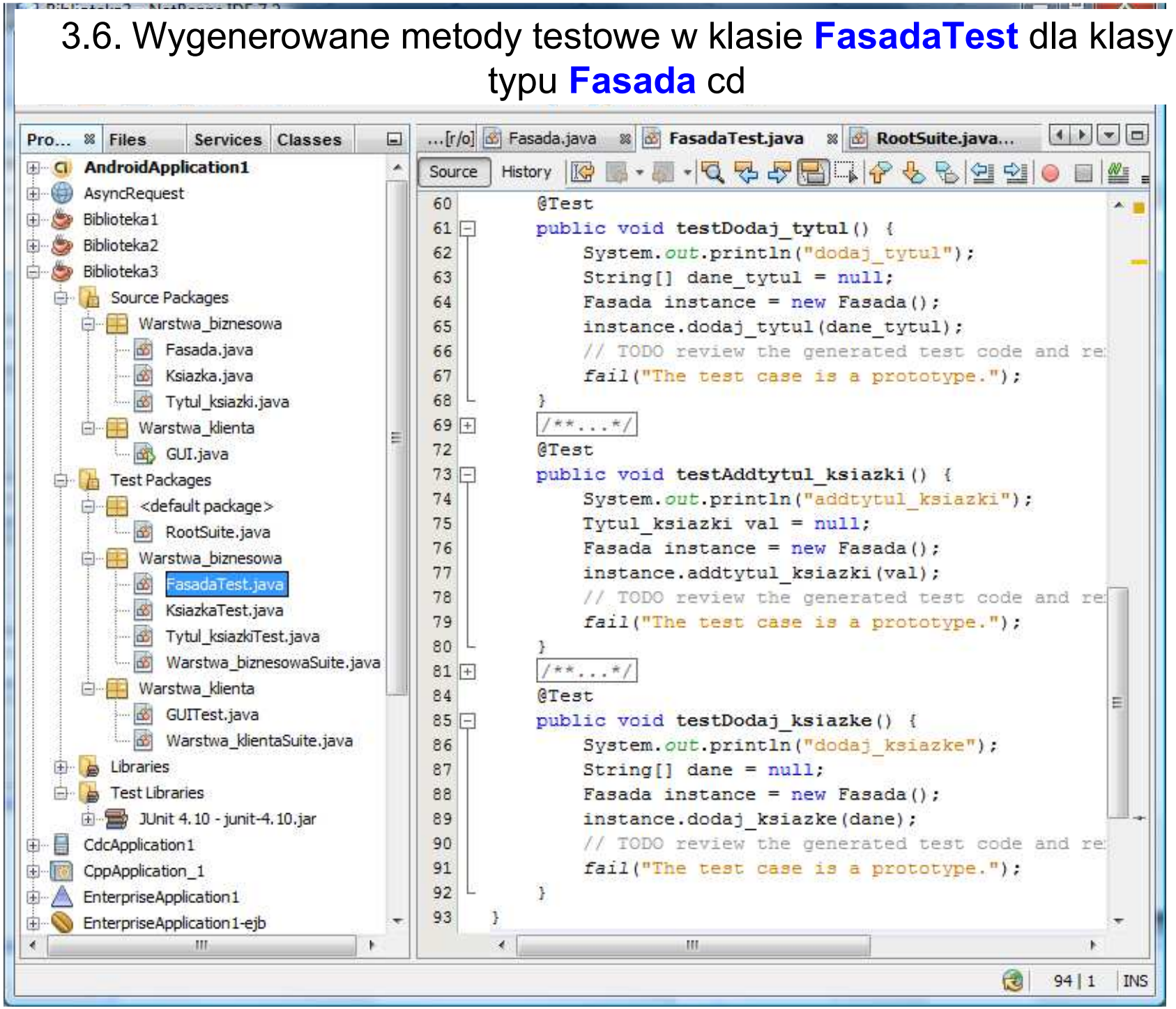
# 3.4. Wygenerowane metody testowe w klasie **FasadaTest** dla klasy typu **Fasada**



# 3.5. Wygenerowane metody testowe w klasie **FasadaTest** dla klasy typu **Fasada** cd



# 3.6. Wygenerowane metody testowe w klasie **FasadaTest** dla klasy typu **Fasada** cd



# 4. Adnotacje określające sposób i moment testowania
# http://www.vogella.com/articles/JUnit/article.html

| Annotation | Description |
|---|---|
| @Test<br>public void method(). | The annotation @Test identifies that a method is a test method. |
| @Before<br>public void method() | Will execute the method before each test. This method can prepare the test environment (e.g. read input data, initialize the class). |
| @After<br>public void method() | Will execute the method after each test. This method can cleanup the test environment (e.g. delete temporary data, restore defaults). |
| @BeforeClass<br>public void method() | Will execute the method once, before the start of all tests. This can be used to perform time intensive activities, for example to connect to a database. |
| @AfterClass<br>public void method() | Will execute the method once, after all tests have finished. This can be used to perform clean-up activities, for example to disconnect from a database. |
| @Ignore | Will ignore the test method. This is useful when the underlying code has been changed and the test case has not yet been adapted. Or if the execution time of this test is too long to be included. |
| @Test (expected = Exception.class) | Fails, if the method does not throw the named exception. |
| @Test(timeout=100) | Fails, if the method takes longer than 100 milliseconds. |

## 5. Metody wspomagające ocenę wyniku testu
## http://www.vogella.com/articles/JUnit/article.html

| Statement | Description |
|---|---|
| fail(String) | Let the method fail. Might be used to check that a certain part of the code is not reached. Or to have failing test before the test code is implemented. |
| assertTrue(true) / assertTrue(false) | Will always be true / false. Can be used to predefine a test result, if the test is not yet implemented. |
| assertTrue([message],boolean condition) | Checks that the boolean condition is true. |
| assertsEquals([String message], expected, actual) | Tests that two values are the same. Note: for arrays the reference is checked not the content of the arrays. |
| assertsEquals([String message], expected, actual, tolerance) | Test that float or double values match. The tolerance is the number of decimals which must be the same. |
| assertNull([message], object) | Checks that the object is null. |
| assertNotNull([message], object) | Checks that the object is not null. |
| assertSame([String], expected, actual) | Checks that both variables refer to the same object. |
| assertNotSame([String], expected, actual) | Checks that both variables refer to different objects. |

# 6. Pomocnicze klasy zdefiniowane przez użytkownika do przeprowadzenia testów jednostkowych metod dodaj_tytul i addtytul_ksiazki klasy Fasada

Source History

```
40      public Tytul_ksiazki Tytul(String dane_tytul[]) {
41          Tytul_ksiazki tytul_ksiazki = new Tytul_ksiazki();
42          tytul_ksiazki.setTytul(dane_tytul[0]);
43          tytul_ksiazki.setNazwisko(dane_tytul[1]);
44          tytul_ksiazki.setImie(dane_tytul[2]);
45          tytul_ksiazki.setISBN(dane_tytul[3]);
46          tytul_ksiazki.setWydawnictwo(dane_tytul[4]);
47          return tytul_ksiazki;
48      }
49      public ArrayList<Tytul_ksiazki> Tytuly() {
50          ArrayList<Tytul_ksiazki> tytuly = new ArrayList();
            String dane_tytulu1[] = {"tytul1", "Jan", "Kowalski", "12345", "W1"};
            String dane_tytulu2[] = {"tytul2", "Piotr", "Nowak", "67891", "W2"};
            tytuly.add(Tytul(dane_tytulu1));
            tytuly.add(Tytul(dane_tytulu2));
            return tytuly;
56      }
```

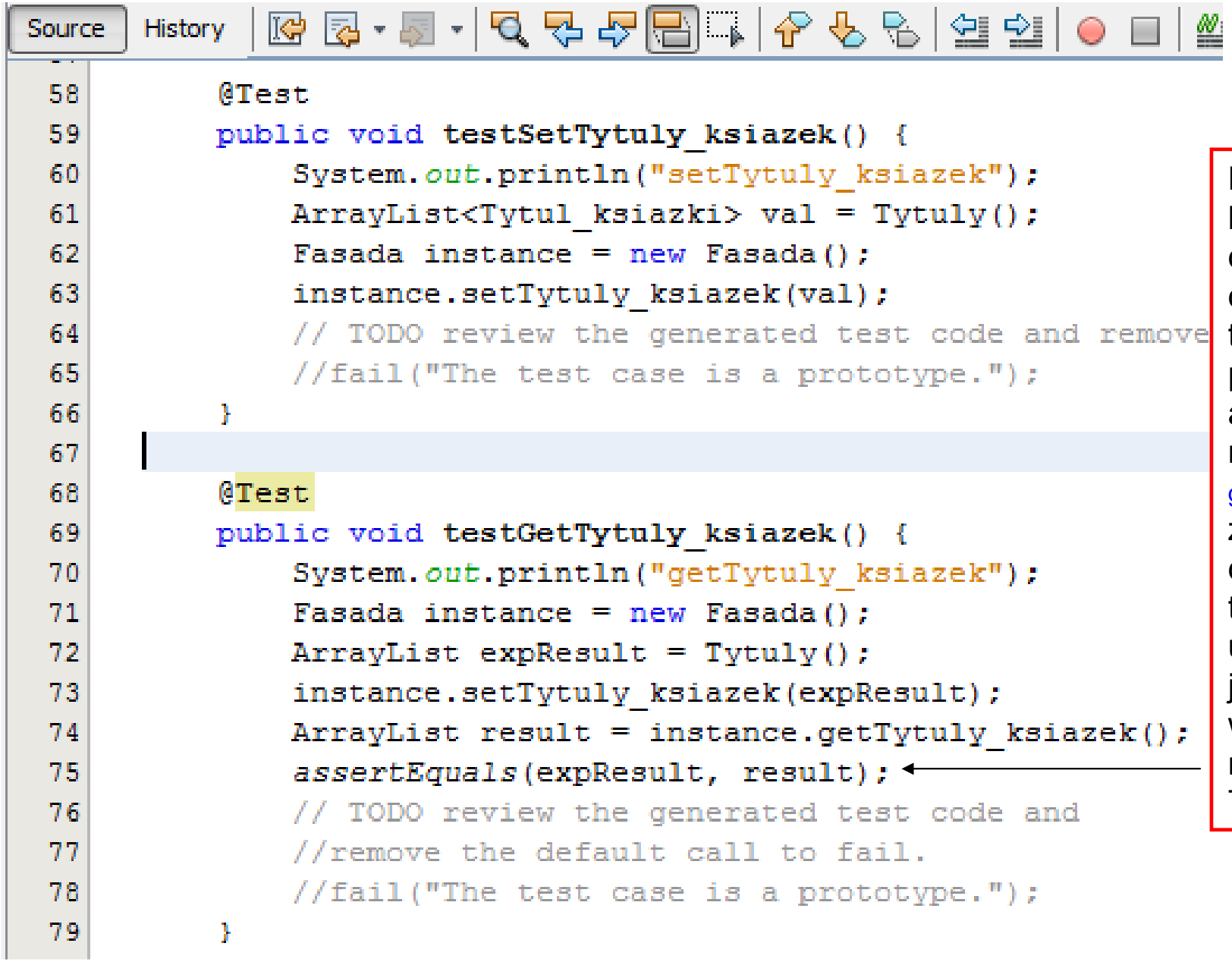
Definicja metody tworzącej wzorcowy obiekt typu Tytul_ksiazki

Definicja metody tworzącej wzorcową kolekcję typu ArrayList z elementami typu Tytul_ksiazki

Utworzenie wzorcowego obiektu typu Tytul_ksiazki

Utworzenie wzorcowej kolekcji typu ArrayList zawierającej elementy - obiekty typu Tytul_ksiazki

# 6.1. Uzupełnione metody do testowania metod **setTytuly_ksiazek** i **getTytuly_ksiazek** klasy **Fasada**



```
58      @Test
59      public void testSetTytuly_ksiazek() {
60          System.out.println("setTytuly_ksiazek");
61          ArrayList<Tytul_ksiazki> val = Tytuly();
62          Fasada instance = new Fasada();
63          instance.setTytuly_ksiazek(val);
64          // TODO review the generated test code and remove
65          //fail("The test case is a prototype.");
66      }
67
68      @Test
69      public void testGetTytuly_ksiazek() {
70          System.out.println("getTytuly_ksiazek");
71          Fasada instance = new Fasada();
72          ArrayList expResult = Tytuly();
73          instance.setTytuly_ksiazek(expResult);
74          ArrayList result = instance.getTytuly_ksiazek();
75          assertEquals(expResult, result);
76          // TODO review the generated test code and
77          //remove the default call to fail.
78          //fail("The test case is a prototype.");
79      }
```

Porównanie kolekcji z dodanymi dwoma tytułami pobranymi z aplikacji metodą getTytuly_ksiazek z kolekcją dwóch tytułów utworzonych jako kolekcja wzorcowa metodą Tytuly

# 6.2. Uzupełnione metody do testowania metod **dodaj_tytul** i **addtytul_ksiazki** klasy **Fasada**

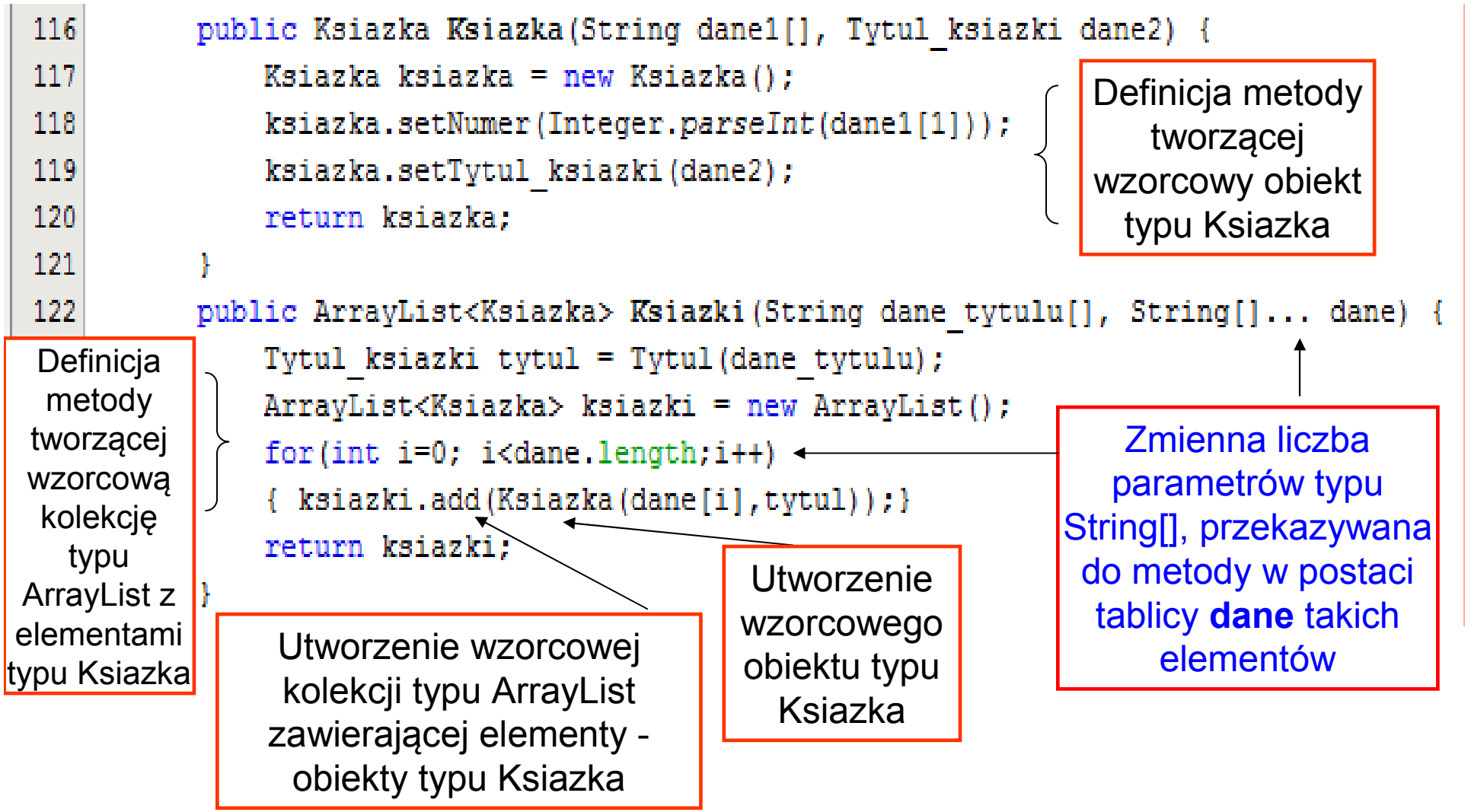
```
85      @Test
86      public void testDodaj_tytul() {
87          System.out.println("dodaj_tytul");
88          String[] dane_tytul = {"tytul1", "Jan", "Kowalski", "12345", "W1"};
89          Fasada instance = new Fasada();
90          instance.dodaj_tytul(dane_tytul);
91          Tytul_ksiazki expResult = Tytul(dane_tytul);
92          Tytul_ksiazki result = instance.getTytuly_ksiazek().get(0);
93          assertEquals(expResult, result);
94          // TODO review the generated test code and remove the defau
95          //fail("The test case is a prototype.");
96      }
97      @Test
98      public void testAddtytul_ksiazki() {
            System.out.println("addtytul_ksiazki");
            String[] dane_tytul = {"tytul1", "Jan", "Kowalski", "12345", "W1"};
            Tytul_ksiazki val = Tytul(dane_tytul);
            Fasada instance = new Fasada();
            instance.addtytul_ksiazki(val);
            Tytul_ksiazki result = instance.getTytuly_ksiazek().get(0);
            assertEquals(val, result);
            instance.addtytul_ksiazki(val);
            int result1 = instance.getTytuly_ksiazek().size();
            assertTrue("Blad", 1 == result1);
            // TODO review the generated test code and remove the default call t
            //fail("The test case is a prototype.");
        }
```

Porównanie dodanego tytułu pobranego z aplikacji z tytułem wzorcowym

Sprawdzenie ,czy po próbie dodania tego samego tytułu liczba tytułów nie zmieni się

Porównanie dodanego tytułu pobranego z aplikacji z tytułem wzorcowym

# 6.3. Pomocnicze klasy zdefiniowane przez użytkownika do przeprowadzenia testów jednostkowych metody **dodaj_ksiazke** klasy **Fasada**



# 6.4. Uzupełniona metoda do testowania metody **dodaj_ksiazke** klasy **Fasada**

```
@Test
public void testDodaj_ksiazke() {
    System.out.println("dodaj_ksiazke");
    String[] dane_tytul1 = {"tytul1", "Jan", "Kowalski", "12345", "W1"};
    String[] dane_tytul2 = {"tytul2", "Piotr", "Nowak", "67891", "W2"};
    Fasada instance = new Fasada();
    instance.dodaj_tytul(dane_tytul1);
    instance.dodaj_tytul(dane_tytul2);
    String dane_1[] = {"12345", "1"};
    String dane_2[] = {"67891", "2"};
    String dane_3[] = {"12345", "2"};
    String dane_4[] = {"54321", "1"};
    instance.dodaj_ksiazke(dane_1);
    instance.dodaj_ksiazke(dane_1);
    instance.dodaj_ksiazke(dane_2);
    instance.dodaj_ksiazke(dane_3);
    instance.dodaj_ksiazke(dane_4);
    assertEquals(Ksiazki(dane_tytul1,dane_1,dane_3),
                 instance.getTytuly_ksiazek().get(0).getKsiazka());
    assertEquals(Ksiazki(dane_tytul2,dane_2),
                 instance.getTytuly_ksiazek().get(1).getKsiazka());
    // TODO review the generated test code and remove the default call to
    //fail("The test case is a prototype.");
}
```

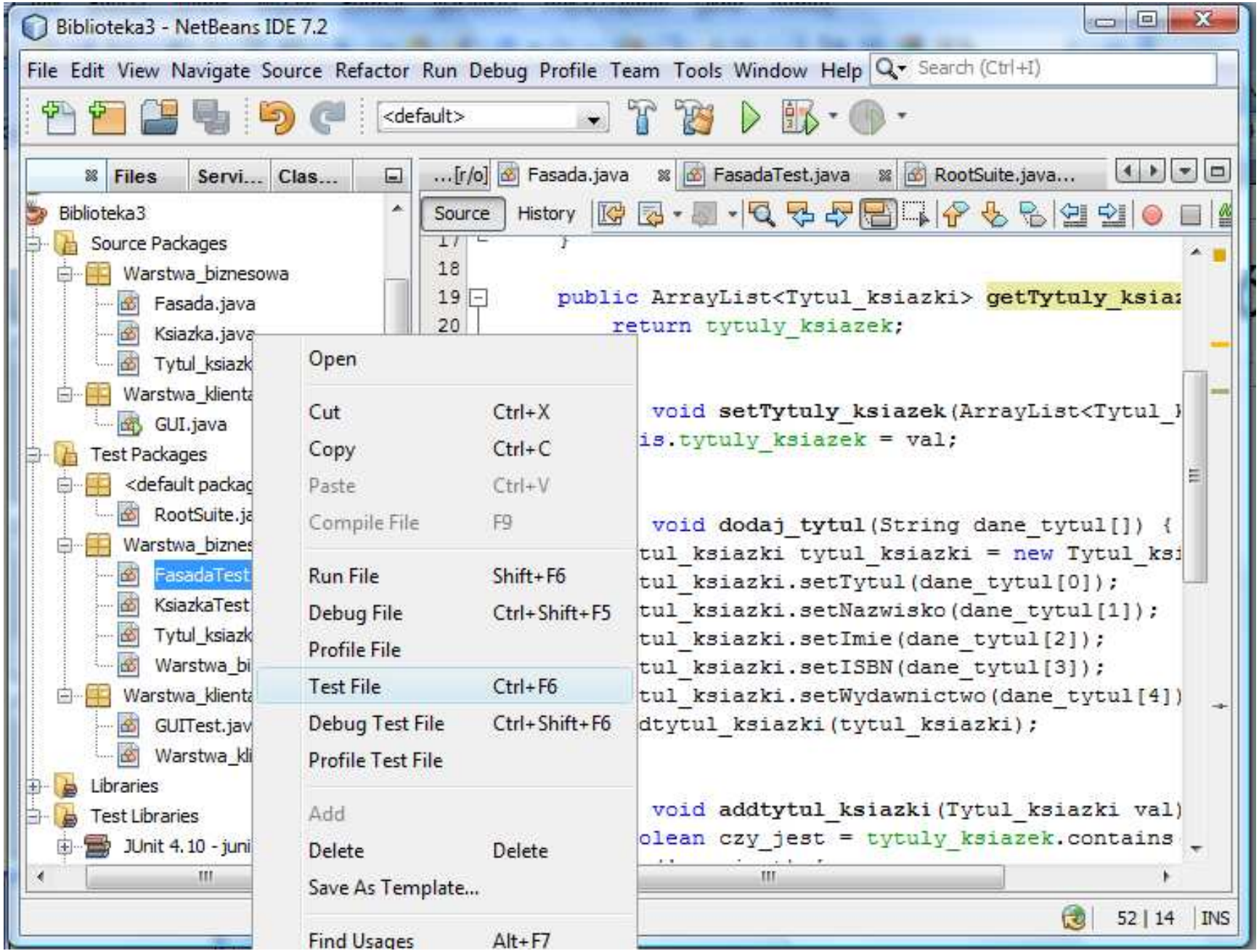
Danie 2 tytułów: z ISBN=12345 i ISBN=67891

Dodanie 3 książek: 2 do tytułu z ISBN=12345 i 1 do tytułu o ISBN=67891

Metoda Ksiazki zwraca wzorcowe kolekcje książek: dla tytułu z ISBN=12345 **(2 książki)** oraz ISBN=67891 **(1 książka)**

Metoda zwraca kolekcję książek pobranych z aplikacji dla tytułu z ISBN=12345 oraz ISBN=67891

# 7. Uruchomienie testów



# 8. Wynik testowania