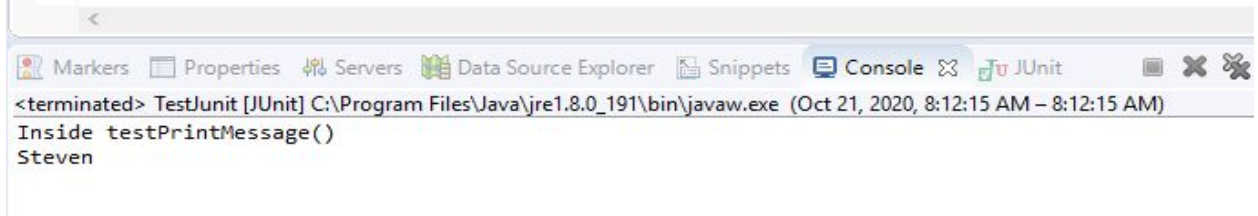


1. Хар хайрцагны шалгалт (Black box testing) болон Цагаан хайрцагны шалгалт (White box testing) хоорондын ялгааг тайлбарла.
  - Хар хайрцагны шалгалт - Функциуд тодорхойлолтын дагуу ажиллаж байгаа эсэхийг шалгахын тулд зөвхөн өгөгдөл болон үр дүнг голчлон авч үздэг. Энэхүү шалгалтанд оруулсан өгөгдөлд тохирсон үр дүн гарч байгаа эсэхийг батлах зорилгоор төрөл бүрийн өгөгдөл бэлтгэнэ. Тийм учраас энэ шалгалтыг голдуу тестерүүд гүйцэтгэдэг.
  - Цагаан хайрцагны шалгалт - Уг шалгалт нь програмын дотоод бүтцийг шалгахын зэрэгцээ өгөгдлийн урсгалыг голлон анхаарч програмын логик дарааллыг шалгах техник юм. Энэхүү шалгалтыг програмистууд өөрсдөө хийж шалгадаг.
2. Хар хайрцагны шалгалт (Black box testing) болон Цагаан хайрцагны шалгалт (White box testing) -ын "Алтан стандарт" (Gold standard) гэж юу вэ?
  - Хар хайрцагны шалгалт - Оролтын дэлгэрэнгүй тест.
  - Цагаан хайрцагны шалгалт - Явц дундын байдлыг тестлэх
- A. TestJUnit.java нэртэй шалгалтын класс үүсгэ. testPrintMessage() шалгалтын тохиолдолд алдааг хүлээн авах (Adding expected exception) боломжтой болго.

**@Test(expected = ArithmeticException.class)**

```
public void testPrintMessage() {  
    System.out.println("Inside testPrintMessage()");  
    messageUtil.printMessage();  
}
```

```
6 public class TestJUnit {  
7  
8     String message = "Steven";  
9     MessageUtil messageUtil = new MessageUtil(message);  
10  
11     //Complete the code in here  
12     @Test(expected = ArithmeticException.class)  
13     public void testPrintMessage() {  
14         System.out.println("Inside testPrintMessage()");  
15         messageUtil.printMessage();  
16     }  
17  
18     public void testSalutationMessage() {  
19         System.out.println("Inside testSalutationMessage()");  
20         message = "Hi!" + "Steven";  
21         assertEquals(message, messageUtil.salutationMessage());  
22     }  
23 }
```



B.

- a. ExpectedException нь шинэ дүрэм тодорхойлох боломж олгодог бөгөөд өөрийн шалгалтад тус алдаа үүсэх үед алдааны зурвас өгөх боломжтой байдаг. Хэрвээ new Person()-д сөрөг нас өгвөл алдаа гэдгийг мэдээлдэг ExpectedException rule бүхий шалгалтын тохиолдол үүсгэ.

**@Rule**

**public ExpectedException exception = ExpectedException.none();**

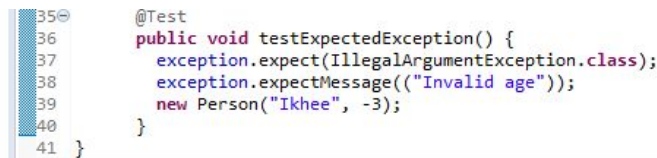
**@Test public void testExpectedException() {**

**exception.expect(IllegalArgumentException.class);**

**exception.expectMessage("Invalid age");**

**new Person("Ikhee", -4);**

**}**



```
35 @Test
36 public void testExpectedException() {
37     exception.expect(IllegalArgumentException.class);
38     exception.expectMessage("Invalid age");
39     new Person("Ikhee", -3);
40 }
41 }
```

JUnit runner output: Finished after 0.034 seconds. Runs: 1/1, Errors: 0, Failures: 0. Test: Q2.PersonTest [Runner: JUnit 4] (0.008 s).

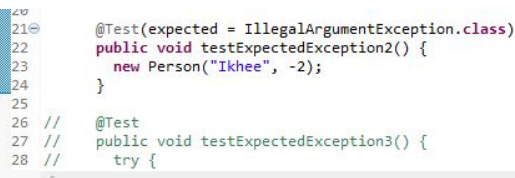
- b. Дээрх алдааг @Test тэмдэглэгээ ашиглан тодорхойл // сөрөг утга олгож шалгалт хийх

**@Test(expected = IllegalArgumentException.class)**

**public void testExpectedException2() {**

**new Person("Ikhee", -5);**

**}**



```
21 @Test(expected = IllegalArgumentException.class)
22 public void testExpectedException2() {
23     new Person("Ikhee", -2);
24 }
25
26 // @Test
27 // public void testExpectedException3() {
28 //     try {
```

JUnit runner output: Finished after 0.027 seconds. Runs: 1/1, Errors: 0, Failures: 0. Test: Q2.PersonTest [Runner: JUnit 4] (0.001 s).

- c. try-catch команд ашигла (дээр шалгахад гарсан алдааг барьж авч мэдээлэх)

**@Test**

**public void testExpectedException3() {**

**try {**

**new Person("Ikhee", -2);**

```

        fail("Should have thrown an IllegalArgumentException
        because age is invalid!");
    } catch (IllegalArgumentException e) {
        assertEquals(e.getMessage(), ("Nas sorog baij bolohgui"));
    }
}

```

```

25
26
27 @Test
28 public void testExpectedException3() {
29     try {
30         new Person("Ikhee", -4);
31         fail("Should have thrown an IllegalArgumentException because age is invalid!");
32     } catch (IllegalArgumentException e) {
33         assertEquals(e.getMessage(), ("Sorog too baij bolohgui"));
34     }
35 }

```

Finished after 0.046 seconds

Runs: 1/1   Errors: 0   Failures: 1

Q2.PersonTest [Runner: JUnit 4] (0.001 s)

testExpectedException3 (0.001 s)

Failure Trace

org.junit.ComparisonFailure: expected:<[Invalid age:-4]> but was:<[Sorog too baij bolohgui]>  
at Q2.PersonTest.testExpectedException3(PersonTest.java:32)

- C. JUnit-эд гүйцэтгэлийг шалгах нь хамгийн төвөгтэй байдаг. JUnit4 нь тус асуудлыг бүрэн шийдээгүй ч хэрэгцээтэй тусламжаар хангадаг: параметр нь хугацаанаас хэтэрвэл (timeout) мэдээлэх боломжтой. Хэрвээ шалгалт хийгдэж байхад хугацаа нь дуусвал гүйцэтгэл удаан байна гэж үзнэ. Performance.java –ийн run() функц-д timeout value =500 байхаар турш.

```

@Test(timeout=500)
public void run(){
    while(true){
    }
}

```

```

org.junit.runners.model.TestTimedOutException: test timed out after 500 milliseconds
at Q3.Performance.run(Performance.java:8)
at java.util.concurrent.FutureTask.run(Unknown Source)

```