

## CREATE RACE HORSE (Julen Larrañaga ~18ordu)

```
public RaceHorse createRaceHorse(double winGain, Race race, Horse horse)
    throws RaceHorseAlreadyExist, WrongParameterException, RaceFullException, RaceDoesntExist, HorseDoesntExist{

    if(race==null || horse==null || winGain<1) throw new WrongParameterException();

    Race newRace = db.find(race.getClass(), race.getKey());
    if(newRace==null) throw new RaceDoesntExist();

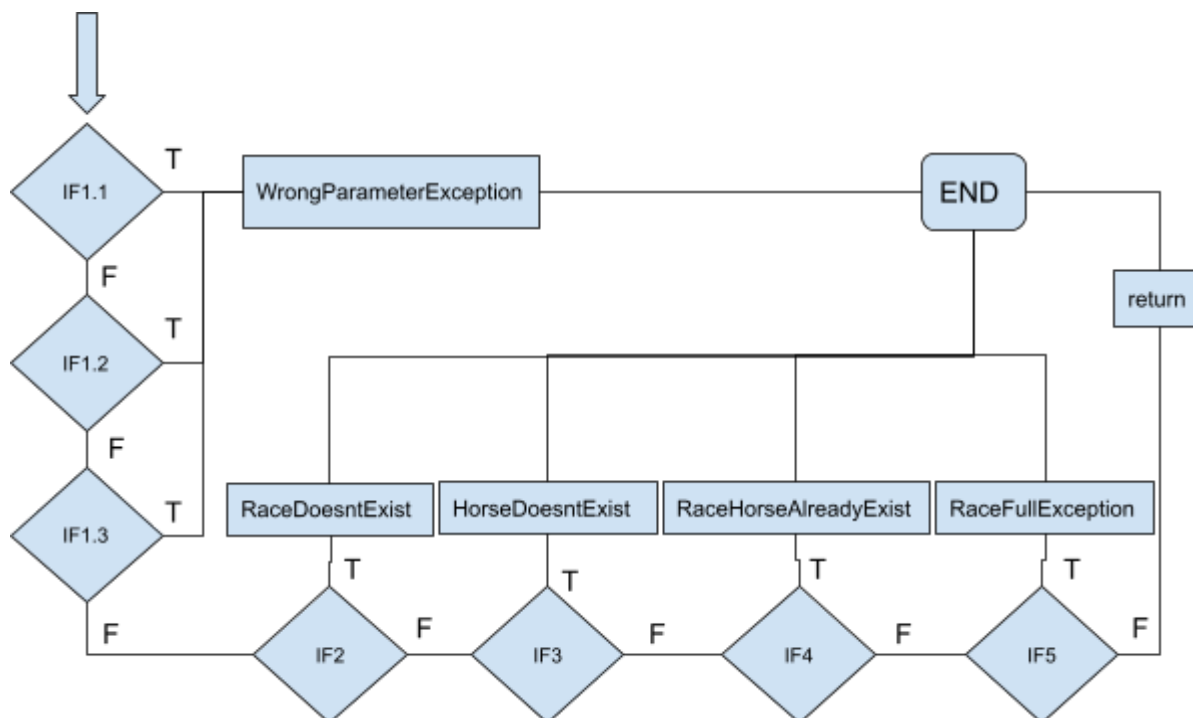
    Horse newHorse = db.find(horse.getClass(), horse.getKey());
    if(newHorse==null) throw new HorseDoesntExist();

    RaceHorse raceHorse = new RaceHorse(winGain, newRace, newHorse);
    if(newRace.getRaceHorses().contains(raceHorse)) throw new RaceHorseAlreadyExist();

    db.getTransaction().begin();
    if(!newRace.addRaceHorse(raceHorse)) throw new RaceFullException();
    db.getTransaction().commit();
    System.out.println("DB>>> New RaceHorse created and added to data base");
    return raceHorse;
}
```

## KUTXA TXURIKO PROBAK (DataAccess.java)

Lehenengo, fluxu kontrol grafoa irudikatu dugu, metodoaren konplexutasun ziklomatikoa kalkulatu ahal izateko:



Konplexutasuna kalkulatzeko, grafoko baldintza kopurua + 1 egingo dugu:

$$\mathbf{V(G)} = 7 + 1 = \mathbf{8}$$

Konplexutasuna 8 denez, programak segi ditzakeen 8 bide ezberdinak adieraziko ditugu:

- 1) 1.1 - WrongParameterException - END
- 2) 1.1 - 1.2 - WrongParameterException - END
- 3) 1.1 - 1.2 - 1.3 - WrongParameterException - END
- 4) 1.1 - 1.2 - 1.3 - 2 - RaceDoesntExist - END
- 5) 1.1 - 1.2 - 1.3 - 2 - 3 - HorseDoesntExist - END
- 6) 1.1 - 1.2 - 1.3 - 2 - 3 - 4 - RaceHorseAlreadyExist - END
- 7) 1.1 - 1.2 - 1.3 - 2 - 3 - 4 - 5 - RaceFullException - END
- 8) 1.1 - 1.2 - 1.3 - 2 - 3 - 4 - 5 - return - END

Ondoren, bide horietako bakoitzeko proba kasu bat adieraziko dugu:

#	DB egoera	Baldintza	Sarrera	Emaitza
1	-	race == null	1.5, null, h	WrongParameterException
2	-	horse==null	1.5, r, null	WrongParameterException
3	-	winGain<1	0.5, r, h	WrongParameterException
4	$r \notin \text{DB}$	newRace==null	1.5,r,h	RaceDoesntExist
5	$h \notin \text{DB}$ $r \in \text{DB}$	newHorse==null	1.5,r,h	HorseDoesntExist
6	$r, h \in \text{DB}$	newRace.getRaceHorses().contains(rh)	1.5,r,h	RaceHorseAlreadyExist
7	$r, h \in \text{DB}$	!newRace.addRaceHorse(rh);	1.5,r,h	RaceFullException
8	$r, h \in \text{DB}$	newRace.addRaceHorse(rh);	1.5,r,h	return

Proba kasuen inplementazioa: **createRaceHorseDAW.java**

Aurkitutako akatsak: **Esperotako emaitzak lortu dira proba kasu guztietan.**

## KUTXA BELTZEKO PROBAK (DataAccess.java)

```
public RaceHorse createRaceHorse(double winGain, Race race, Horse horse)
    throws RaceHorseAlreadyExist,WrongParameterException, RaceFullException, RaceDoesntExist
```

Lehenengo, baliokidetasun egoki eta ez egokiak definituko ditugu:

	Sarrera-baldintza	BK Egokia	BK Ez Egokia
Sarrera Parametroak	Race r	$r \neq \text{null}$ <b>(1)</b>	$r == \text{null}$ <b>(9)</b>
	Horse h	$h \neq \text{null}$ <b>(2)</b>	$h == \text{null}$ <b>(10)</b>
	double winGain	$\text{winGain} \geq 1$ <b>(3)</b>	$\text{winGain} < 1$ <b>(11)</b>
	Race r	$r \in \text{DB}$ <b>(4)</b>	$r \notin \text{DB}$ <b>(12)</b>
	Horse h	$h \in \text{DB}$ <b>(5)</b>	$h \notin \text{DB}$ <b>(13)</b>
Programaren Semantika	Race r	$r.\text{getFinished()} == \text{false}$ <b>(6)</b>	$r.\text{getFinished()} == \text{true}$ <b>(14)</b>
	RaceHorse rh	$rh \notin \text{DB}$ <b>(7)</b>	$rh \in \text{DB}$ <b>(15)</b>
	Race r	$r.\text{size()} < r.\text{numOfStreets}()$ <b>(8)</b>	$r.\text{size()} = r.\text{numOfStreets}()$ <b>(16)</b>

Eratortako proba kasuak:

Estalitako BK	DB Egoera	Sarrera	Emaitza
1,2,3,4,5,6,7,8	$r, h \in \text{DB}$ eta $rh \notin \text{DB}$	(1.5, r, h)	return RaceHorse
9	-	(1.5, null, h)	WrongParameterException
10	-	(1.5, r, null)	WrongParameterException
11	-	(0.5, r, h)	WrongParameterException
12	$r \notin \text{DB}$	(1.5, r, h)	RaceDoesntExist
13	$h \notin \text{DB}$	(1.5, r, h)	HorseDoesntExist
14	$r, h \in \text{DB}$	(1.5, r, h)	RaceFinished
15	$r, h, rh \in \text{DB}$	(1.5, r, h)	RaceHorseAlreadyExist
16	$r, h \in \text{DB}$ eta $rh \notin \text{DB}$	(1.5, r, h)	RaceFullException

Proba kasuen inplementazioa: **createRaceHorseDAB.java**

Aurkitutako akatsak: **Lasterketa amaitu den ez dut egiaztatzen datu baseko funtzioan, beraz erroreak suerta daitezke.**

### KUTXA BELTZEKO PROBAK (BLFacadeImplementation.java)

```
public RaceHorse createRaceHorse(double winGain, Race race, Horse horse)
    throws RaceHorseAlreadyExist,WrongParameterException, RaceFullException, RaceDoesntExist
```

Lehenengo, baliokidetasun egoki eta ez egokiak definituko ditugu:

	Sarrera-baldintza	BK Egokia	BK Ez Egokia
Sarrera Parametroak	Race r	$r \neq \text{null}$ <b>(1)</b>	$r == \text{null}$ <b>(9)</b>
	Horse h	$h \neq \text{null}$ <b>(2)</b>	$h == \text{null}$ <b>(10)</b>
	double winGain	$\text{winGain} \geq 1$ <b>(3)</b>	$\text{winGain} < 1$ <b>(11)</b>
	Race r	$r \in \text{DB}$ <b>(4)</b>	$r \notin \text{DB}$ <b>(12)</b>
	Horse h	$h \in \text{DB}$ <b>(5)</b>	$h \notin \text{DB}$ <b>(13)</b>
Programaren Semantika	Race r	$r.\text{getFinished()} == \text{false}$ <b>(6)</b>	$r.\text{getFinished()} == \text{true}$ <b>(14)</b>
	RaceHorse rh	$rh \notin \text{DB}$ <b>(7)</b>	$rh \in \text{DB}$ <b>(15)</b>
	Race r	$r.\text{size()} < r.\text{numOfStreets()}$ <b>(8)</b>	$r.\text{size()} = r.\text{numOfStreets()}$ <b>(16)</b>

Eratortako proba kasuak:

Estalitako BK	DB Egoera	Sarrera	Emaitza
1,2,3,4,5,6,7,8	$r, h \in \text{DB}$ eta $rh \notin \text{DB}$	(1.5, r, h)	return RaceHorse
9	-	(1.5, null, h)	WrongParameterException
10	-	(1.5, r, null)	WrongParameterException
11	-	(0.5, r, h)	WrongParameterException
12	$r \notin \text{DB}$	(1.5, r, h)	RaceDoesntExist
13	$h \notin \text{DB}$	(1.5, r, h)	HorseDoesntExist

14	$r, h \in DB$	$(1.5, r, h)$	RaceFinished
15	$r, h, rh \in DB$	$(1.5, r, h)$	RaceHorseAlreadyExist
16	$r, h \in DB$ eta $rh \notin DB$	$(1.5, r, h)$	RaceFullException

Proba kasuen implementazioa: **createRaceHorseMockInt.java**  
Aurkitutako akatsak:

**Ezin izan ditut proba kasuak implementatu, Mockito-ko @Mocks eta @InjectMocks etiketek honako errorea ematen zidatelako:**

The screenshot shows the Eclipse IDE with the following components:

- Package Explorer:** Shows the project structure with 'CreateRaceHorseMockInt' selected.
- JUnit Runner:** Shows the test results for 'test1' (0.164 s).
- Source Editor:** Contains the following Java code:
 

```

19 @RunWith(MockitoJUnitRunner.class)
20 public class CreateRaceHorseMockInt {
21
22     //Race mockedRace=Mockito.mock(Race.class);
23
24     @Mock
25     DataAccess horseRacesDAO;
26
27     @InjectMocks
28     BLFacadeImplementation sut;
29
30     @Test
31     public void test1(){
32         try {
33             Race race = new Race(new Date(), 4, new StartTime("10:30"));
34             Horse horse = new Horse("Julen", "Belauntza", 20, "male", 99);
35             double winGain = 1.5;
36
37             Mockito.doReturn(new RaceHorse(winGain, race, horse)).when(horseRacesDAO).createRaceHorse(winGain, race, horse);
38
39             RaceHorse rh = sut.createRaceHorse(winGain, race, horse);
40
41             assertEquals(rh.getWinGain(), winGain);
42             assertEquals(rh.getRace(), race);
43             assertEquals(rh.getHorse(), horse);
44
45         } catch (Exception e) {
46             e.printStackTrace();
47             fail();
48         } finally {
49             sut.close();
50         }
51     }
52 }

```
- Failure Trace:** Shows the error: 'java.lang.ExceptionInInitializerError' at 'org.mockito.cglib.core.KeyFactory\$Generator.generateClass(KeyFactory.java:167)'. The stack trace includes:
 

```

at org.mockito.cglib.core.KeyFactory$Generator.generateClass(KeyFactory.java:167)
at org.mockito.cglib.core.DefaultGeneratorStrategy.generate(DefaultGeneratorStrategy.java:25)
at org.mockito.cglib.core.AbstractClassGenerator.create(AbstractClassGenerator.java:217)
at org.mockito.cglib.core.KeyFactory$Generator.create(KeyFactory.java:145)
at org.mockito.cglib.core.KeyFactory.create(KeyFactory.java:117)
at org.mockito.cglib.core.KeyFactory.create(KeyFactory.java:109)
at org.mockito.cglib.core.KeyFactory.create(KeyFactory.java:105)

```

[GITHUB](#)

[SONARCLOUD](#)