# TEMA DE LABORATOR - JUNIT

# Se considera o aplicatie cu 2 sau mai multe clase. (Exemplu: Student-Grupa, Animal-Zoo de mai jos)

# Sa se testeze folosind JUnit (4 sau 5)

# Sa se utilizeze cat mai multe metode de tipul assert

# Sa se foloseasca suite de teste, teste parametrizate etc

# Suplimentar: se pot testa clase si in alte limbaje de programare folosindu-se un framework pentru scrierea de teste unitare.

# Exemplu de clase

Student.java

public class Student {

private String name;

private String age;

public Student(String name, String age) {

super();

this.name = name;

this.age = age;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getAge() {

return age;

}

public void setAge(String age) {

this.age = age;

}

}

Grupa.java

import java.util.ArrayList;

import java.util.List;

public class Grupa {

List<Student> students;

Grupa () {

students = new ArrayList<Student>();

}

public List<Student> getStudents() {

return students;

}

public void setStudents(List<Student> students) {

this.students = students;

}

public void addStudent(Student student) {

students.add(student);

}

public Student getStudent(String name) {

for (Student st : students) {

if (null != st.getName() && st.getName().equals(name)) {

return st;

}

}

return null;

}

public boolean areStudentsInGroup() {

if (students.size() == 0) {

return false;

}

return true;

}

}

Test.java

import org.junit.Assert.\*;

public class Test {

private Grupa grupa;

public Test() {

}

@BeforeClass

public static void setUpClass() {

}

@AfterClass

public static void tearDownClass() {

}

@Before

public void setUp() {

grupa = new Grupa();

}

@After

public void tearDown() {

grupa = null;

}

@Test

public void testNoStudentInGroup() {

Assert.assertEquals(false, grupa.areStudentsInGroup());

}

@Test

public void testAddStudent() {

Student st = new Student("Elena", "11");

grupa.addStudent(st);

Assert.assertTrue(grupa.getStudent("Elena").equals(st));

}

}

## Exerciţiu

1. Creaţi clasa Animal şi clasa Zoo. Clasa Zoo conţine un vector de animale. Implementaţi metodele: addAnimal(Animal a), removeAnimal(Animal a), boolean areAnimals(), getAnimals(), size(). Creaţi o clasa Test unde veţi verifica diverse scenarii:
   * (1p) La rularea fiecǎrei metode veti instanţia clasa Zoo.
   * (1p) Metoda testAddAnimal - adaugă un obiect Animal şi verificǎ daca adǎugarea a avut loc cu succes. Folosiţi: **assertEquals**
   * (1p) Metoda testRemoveAnimal - folosiţi **assertTrue**
   * (1p) Metoda testAreAnimalsInZoo - testul picǎ dacǎ metoda returneazǎ false. Hint: **Assert.fail()**
   * (1p) Metoda testGetAnimals - adǎugaţi douǎ obiecte Animal. Verificaţi ca adǎugarea a avut loc cu succes. Folosiţi **assertFalse**.