## Laboratório 15 - Herança e Polimorfismo III (22/10/2014)

1 / 2.

```
package lab15;

public interface HealthProfessional {
       public String care();
}
```

```
package lab15;

public interface Person {

       public String getName();
}
```

```
package lab15;

public class Hospital implements HealthProfessional{
       private HealthProfessional[] professionalsList;   // cria um array do tipo
HealthProfessional com tantos elementos quantos os indicados
       private int auxNum = 0;

       /**
        * Constructor
        * @param i
        */
       public Hospital(int i){
              professionalsList = new HealthProfessional[i];
       }

       /**
        * Adds a professional to one of the elements of professionalsList until it's full
        * @param professional
        * @return true  if the professional is in professionalsList, false if the
professionalsList array is full
        */
       public boolean addProfessional(HealthProfessional professional) {
              if (auxNum < professionalsList.length)
              {
                     professionalsList[auxNum] = professional;
                     auxNum++;
                     return true;
              }
              else
                     return false;
       }

/**
 * @return a string with the name of the professional and what  he/she is doing.
 */
       @Override
       public String care()
       {
              return null;
       }
}
```

```
package lab15;

public class Doctor implements HealthProfessional{
        public String name;

        public Doctor(String name) {
                this.name = name;
        }

        @Override
        public String care() {
                return "Dr. "+ name + " is treating the patient.";
        }

        public String getName() {
                return name;
        }
}
```

```
package lab15;

public class Nurse implements HealthProfessional {
        public String name;

        public Nurse(String name) {
                this.name = name;
        }

        @Override
        public String care() {
                // TODO Auto-generated method stub
                return "Nurse " + name + " is caring for a patient.";
        }

        public String getName() {
                return name;
        }
}
```

```
package lab15;

public class Helper implements HealthProfessional {
        public String name;

        public Helper(String name) {
                this.name = name;
        }

        @Override
        public String care() {

                return "Helper "+ name +" is preparing the patient's lunch.";
        }

        public String getName() {
                return name;
        }
}
```

```
package lab15;

import static org.junit.Assert.*;

import org.junit.Assert;
import org.junit.Before;
import org.junit.Test;

public class HospitalTests {

	private Hospital _hospital;

	@Before
	public void init()
	{
	// Arrange
	_hospital = new Hospital(3);
	}

	@Test
	public void testAddProfessionals()
	{
	// Act & Assert
	Assert.assertTrue(_hospital.addProfessional(new Doctor("Marco")));
	Assert.assertTrue(_hospital.addProfessional(new Nurse("Miguel")));
	Assert.assertTrue(_hospital.addProfessional(new Helper("Mateus")));
	}

	@Test
	public void testExceededNumberOfProfessionals()
	{
	// Act & Assert
	Assert.assertTrue(_hospital.addProfessional(new Doctor("Jorge")));
	Assert.assertTrue(_hospital.addProfessional(new Nurse("Joel")));
	Assert.assertTrue(_hospital.addProfessional(new Helper("Jeremias")));
	Assert.assertFalse(_hospital.addProfessional(new Helper("Jaime")));   // como só
criámos um array com 3 elementos, ele não permite colocar mais -> dá falso
	Assert.assertFalse(_hospital.addProfessional(new Doctor("Joaquim"))); // como só
criámos um array com 3 elementos, ele não permite colocar mais -> dá falso
	}

	@Test
	public void testHealthProfessionals()
	{
	// Arrange
	HealthProfessional p1 = new Doctor("Fernando");
	HealthProfessional p2 = new Nurse("Fabio");
	HealthProfessional p3 = new Helper("Fonseca");
	// Act & Assert
	Assert.assertEquals("Dr. Fernando is treating the patient.", p1.care());
	Assert.assertEquals("Nurse Fabio is caring for a patient.", p2.care());
	Assert.assertEquals("Helper Fonseca is preparing the patient's lunch.",
p3.care());
	}

@Test
	public void testPersons()
```

```
        {
                // Arrange
                Person p1 = new Doctor("Rui");
                Person p2 = new Nurse("Ricardo");
                Person p3 = new Helper("Renato");
                // Act & Assert
                Assert.assertSame("Rui", p1.getName());
                Assert.assertSame("Ricardo", p2.getName());
                Assert.assertSame("Renato", p3.getName());
        }
}
```