# Programming Advanced for QA – Regular Exam

# 13 April 2024

Submit your zip file here: <https://judge.softuni.org/Contests/Compete/Index/4553#2>

# 3. Unit Test: Employee Management System

You are given a class EmployeeManagementSystem holding a int **property** EmployeeCount and **three** **methods** AddEmployee(), RemoveEmployee() and GetAllEmployees():

Картина, която съдържа текст, Шрифт, линия, екранна снимка

Описанието е генерирано автоматично

Картина, която съдържа текст, Шрифт, линия, екранна снимка

Описанието е генерирано автоматично

Картина, която съдържа текст, Шрифт, екранна снимка, линия

Описанието е генерирано автоматично

You will need to **test** **the** **three** **methods** using the test file EmployeeManagementSystemTests.cs, inside they are **6 empty tests**:

Картина, която съдържа текст, екранна снимка, Шрифт

Описанието е генерирано автоматично

**Note! You may need to use EmployeeCount property and CollectionAssert to verify expected and actual result.**

When you are ready make sure your **tests run:**

**Картина, която съдържа текст, екранна снимка, Шрифт

Описанието е генерирано автоматично**

**IMPORTANT:** **DO NOT REMOVE OR CHANGE ANY NAMESPACES AND USINGS.**

# using NUnit.Framework;

# using System;

# using System.Collections.Generic;

# namespace TestApp.Tests

# {

# [TestFixture]

# public class EmployeeManagementSystemTests

# {

# [Test]

# public void Test\_Constructor\_CheckInitialEmptyEmployeeCollectionAndCount()

# {

# // Arrange

# EmployeeManagementSystem empSystem = new EmployeeManagementSystem();

# // Assert

# Assert.AreEqual(0, empSystem.EmployeeCount);

# CollectionAssert.IsEmpty(empSystem.GetAllEmployees());

# }

# [Test]

# public void Test\_AddEmployee\_ValidEmployeeName\_AddNewEmployee()

# {

# // Arrange

# EmployeeManagementSystem empSystem = new EmployeeManagementSystem();

# // Act

# empSystem.AddEmployee("John Doe");

# // Assert

# Assert.AreEqual(1, empSystem.EmployeeCount);

# CollectionAssert.Contains(empSystem.GetAllEmployees(), "John Doe");

# }

# [Test]

# public void Test\_AddEmployee\_NullOrEmptyEmployeeName\_ThrowsArgumentException()

# {

# // Arrange

# EmployeeManagementSystem empSystem = new EmployeeManagementSystem();

# // Assert

# Assert.Throws<ArgumentException>(() => empSystem.AddEmployee(null));

# Assert.Throws<ArgumentException>(() => empSystem.AddEmployee(""));

# Assert.Throws<ArgumentException>(() => empSystem.AddEmployee(" "));

# }

# [Test]

# public void Test\_RemoveEmployee\_ValidEmployeeName\_RemoveFirstEmployeeName()

# {

# // Arrange

# EmployeeManagementSystem empSystem = new EmployeeManagementSystem();

# empSystem.AddEmployee("John Doe");

# empSystem.AddEmployee("Jane Smith");

# // Act

# empSystem.RemoveEmployee("John Doe");

# // Assert

# Assert.AreEqual(1, empSystem.EmployeeCount);

# CollectionAssert.DoesNotContain(empSystem.GetAllEmployees(), "John Doe");

# }

# [Test]

# public void Test\_RemoveEmployee\_NullOrEmptyEmployeeName\_ThrowsArgumentException()

# {

# // Arrange

# EmployeeManagementSystem empSystem = new EmployeeManagementSystem();

# // Assert

# Assert.Throws<ArgumentException>(() => empSystem.RemoveEmployee(null));

# Assert.Throws<ArgumentException>(() => empSystem.RemoveEmployee(""));

# Assert.Throws<ArgumentException>(() => empSystem.RemoveEmployee(" "));

# }

# [Test]

# public void Test\_GetAllEmployees\_AddedAndRemovedEmployees\_ReturnsExpectedEmployeeCollection()

# {

# // Arrange

# EmployeeManagementSystem empSystem = new EmployeeManagementSystem();

# empSystem.AddEmployee("John Doe");

# empSystem.AddEmployee("Jane Smith");

# empSystem.RemoveEmployee("John Doe");

# // Act

# List<string> allEmployees = empSystem.GetAllEmployees();

# // Assert

# Assert.AreEqual(1, allEmployees.Count);

# CollectionAssert.DoesNotContain(allEmployees, "John Doe");

# CollectionAssert.Contains(allEmployees, "Jane Smith");

# }

# }

# }