# Lab: Unit Testing

Problems for in-class lab for the [Python OOP Course @SoftUni](https://softuni.bg/courses/python-oop). Submit your solutions in the SoftUni judge system at <https://judge.softuni.bg/Contests/1948>

## Test Worker

Load provided skeleton in the IDE you use. Add new project **Tests**

**class** Worker:  
  
 **def** \_\_init\_\_(self, name, salary, energy):  
 self.name = name  
 self.salary = salary  
 self.energy = energy  
 self.money = 0  
  
 **def** work(self):  
 **if** self.energy <= 0:  
 **raise** Exception(**'Not enough energy.'**)  
  
 self.money += self.salary  
 self.energy -= 1  
  
 **def** rest(self):  
 self.energy += 1  
  
 **def** get\_info(self):  
 **return** (**f'{**self.name**} has saved {**self.money**} money.'**)

Create a class WorkerTests

In judge you need to submit just the WokerTests class, with the unitttest module imported.

Create the following tests:

* Test if the worker is initialized with correct name, salary and energy
* Test if the worker's energy is incremented after the rest method is called
* Test if an error is raised if the worker tries to work with negative energy or equal to 0
* Test if the worker's money is increased by his salary correctly after the work method is called
* Test if the worker's energy is decreased after the work method is called
* Test if the get\_info method returns the proper string with correct values

## Test Cat

**class** Cat:  
  
 **def** \_\_init\_\_(self, name):  
 self.name = name  
 self.fed = **False** self.sleepy = **False** self.size = 0  
  
 **def** eat(self):  
 **if** self.fed:  
 **raise** Exception(**'Already fed.'**)  
  
 self.fed = **True** self.sleepy = **True** self.size += 1  
  
 **def** sleep(self):  
 **if not** self.fed:  
 **raise** Exception(**'Cannot sleep while hungry'**)  
  
 self.sleepy = **False**

Create a class CatTests

In judge you need to submit just the CatTests class, with the unitttest module imported.

Create the following tests:

* Cat's size is increased after eating
* Cat is fed after eating
* Cat cannot eat if already fed, raises an error
* Cat cannot fall asleep if not fed, raises an error
* Cat is not sleepy after sleeping

### Hints

Follow the logic of the previous problem

## Refactor Tests

Refactor the tests for Worker and Cat classes

Make sure that:

* **Names** of the tests are **descriptive**
* You use **appropriate** **assertions** (assert equals vs assert true/false)
* You use **assertion** **messages**
* There are **no magic values**
* There is **no code duplication** (Don't Repeat Yourself)

### Hints

Extract constants

Create a method that executes **before each test**

Add proper assertion messages