University Echahid Hamma Lakhdar El Oued
Faculty of Exact Sciences - Department of Computer Science
Level: Master 1
Module: Developing maintainable software
**Typical correction**

**Answer the questions: 6pts**

1- No, it is not. The cost of making legacy software maintainable is probably really high. Instead, we can work on gradually satisfying the requirements and making it maintainable.

2- Yes (no is also an acceptable answer if you motivate it correctly; you need to justify that we need to add only failing tests)

If you add a test then you find that the requirement is satisfied, you can go for the next step.

3-

1) Cohesion:
   1- The Reuse/Release Equivalence Principle (REP)
   2- The Common Reuse Principle (CRP)
   3- The Common Closure Principle (CCP)
2) Coupling
   1- The Acyclic Dependencies Principle (ADP)
   2- The Stable-Dependency Principle (SDP)
   3- The Stable-Abstractions Principle (SAP)

**Exercise 1: 7pts**

```
def test_exception_on_none_input():
    with pytest.raises(ValueError) as exception:
        average(None)

    assert str(exception.value) == "None inputs are not accepted!"
```

```
def average(numbers: list[int]):
    raise ValueError("None inputs are not accepted!")
```

```
def test_average_list_with_zero_as_an_element_return_zero():
    assert average([0]) == 0
```

```
def average(numbers: list[int]):
    if numbers is None:
        raise ValueError("None inputs are not accepted!")
    return 0
```

**Exercise 3: 7pts**

SRP: Change could be on dispensed goods and on payment methods.
OCP: We do not change the goods, instead we extend it with new types of goods. Same for paments
LSP: Coffee and tea could replace Good without any problem.