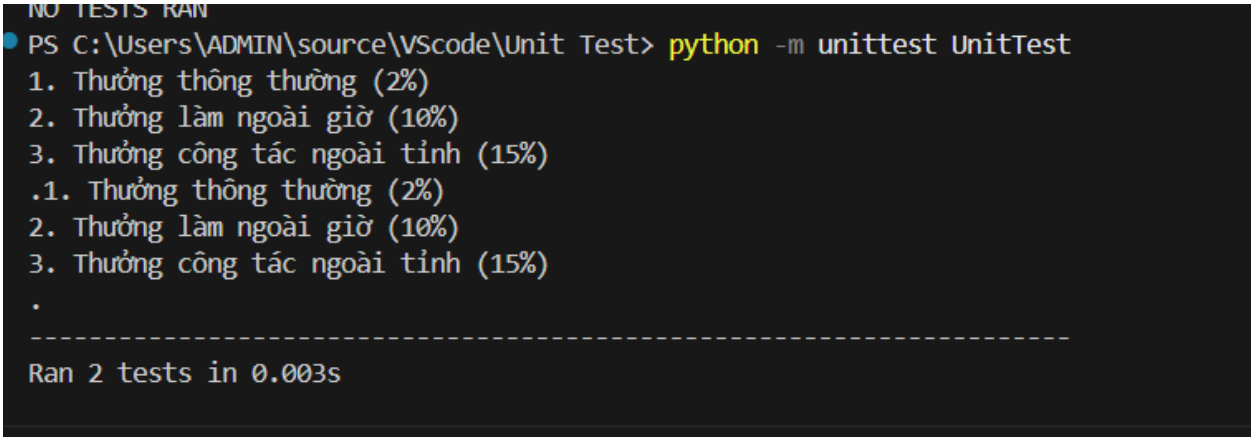
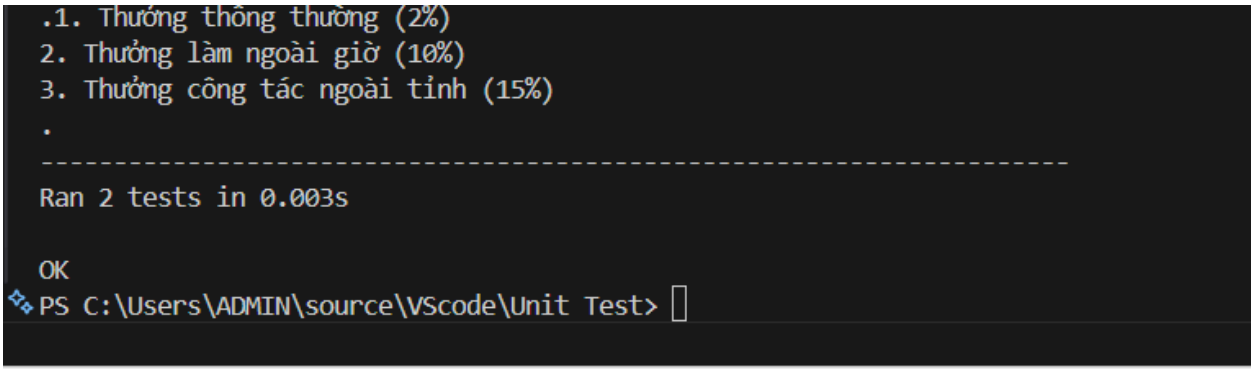
Ảnh mính chứng:





Code:

Bonus.py:

```python
from abc import ABC, abstractmethod

class BonusStrategy(ABC):
    @abstractmethod
    def calculate(self, base_salary):
        pass
class NormalBonus(BonusStrategy):
    def calculate(self, salary):
        return salary * 0.02
class OvertimeBonus(BonusStrategy):
    def calculate(self, salary):
        return salary * 0.10
class BusinessTripBonus(BonusStrategy):
    def calculate(self, salary):
        return salary * 0.15
```

employee.py:

```python
class Employee:

    def __init__(self, emp_id, name, base_salary, bonus_strategy):
```

```
        self.emp_id = emp_id
        self.name = name
        self.base_salary = base_salary
        self.bonus_strategy = bonus_strategy
    def calculate_bonus(self):
        return self.bonus_strategy.calculate(self.base_salary)
employee_factory.py:
from employee import Employee
from bonus import NormalBonus, OvertimeBonus, BusinessTripBonus

def create_employee_from_input():
    emp_id = input("Nhập mã nhân viên: ")
    name = input("Nhập tên nhân viên: ")
    salary = float(input("Nhập lương cơ bản: "))

    print("1. Thưởng thông thường (2%)")
    print("2. Thưởng làm ngoài giờ (10%)")
    print("3. Thưởng công tác ngoài tỉnh (15%)")

    choice = input("Chọn loại thưởng: ")

    if choice == "1":
        bonus = NormalBonus()
    elif choice == "2":
        bonus = OvertimeBonus()
    elif choice == "3":
        bonus = BusinessTripBonus()
    else:
        raise ValueError("Loại thưởng không hợp lệ")

    return Employee(emp_id, name, salary, bonus)

UnitTest.py:
import unittest
from unittest.mock import patch
from employee_factory import create_employee_from_input
from bonus import NormalBonus, OvertimeBonus


class TestEmployeeInput(unittest.TestCase):

    @patch("builtins.input", side_effect=[
        "E01",                # mã NV
        "Nguyen Van A",       # tên NV
        "10000000",           # lương
```

```
        "1"                    # thưởng thường
    ])
    def test_create_employee_normal_bonus(self, mock_input):
        employee = create_employee_from_input()

        self.assertEqual(employee.emp_id, "E01")
        self.assertEqual(employee.name, "Nguyen Van A")
        self.assertEqual(employee.base_salary, 10_000_000)
        self.assertIsInstance(employee.bonus_strategy, NormalBonus)
        self.assertEqual(employee.calculate_bonus(), 200_000)

    @patch("builtins.input", side_effect=[
        "E02",
        "Tran Van B",
        "8000000",
        "2"
    ])
    def test_create_employee_overtime_bonus(self, mock_input):
        employee = create_employee_from_input()

        self.assertEqual(employee.name, "Tran Van B")
        self.assertIsInstance(employee.bonus_strategy, OvertimeBonus)
        self.assertEqual(employee.calculate_bonus(), 800_000)


if __name__ == "__main__":
    unittest.main()
```