Tests Driven Development

```
from dataclasses import dataclass, field
from enum import Enum



class OrderStatus(Enum):
    OPEN = "open"
    PAID = "paid"



@dataclass
class LineItem:
    name: str
    price: int
    quantity: int = 1

    @property
    def total(self) -> int:
        return self.price * self.quantity



@dataclass
class Order:
    line_items: list[LineItem] = field(default_factory=list)
    status: OrderStatus = OrderStatus.OPEN

    @property
    def total(self) -> int:
        return sum(item.total for item in self.line_items)

    def pay(self) -> None:
        self.status = OrderStatus.PAID
```

```
from pay.order import LineItem



def test_line_item_total() -> None:
    line_item = LineItem(name="Test", price=100)
    assert line_item.total == 100



def test_line_item_total_quantity() -> None:
    line_item = LineItem(name="Test", price=100, quantity=2)
    assert line_item.total == 200
```

Revision #2
Created 7 March 2024 04:42:44 by victor
Updated 7 March 2024 04:53:04 by victor