```python
class TestOrder(TestCase):
    def test_total(self):
        order = Order('Item', 1, 2)
        self.assertEqual(order.total, 2)

    def test_init(self):
        with self.assertRaises(ValueError) as cm:
            order = Order('Item', -1, 2)

    def test_price_non_negative(self):
        with self.assertRaises(ValueError) as cm:
            order = Order('Item', 1, 1)
            order.price = -1

    def test_quantity_non_negative(self):
        with self.assertRaises(ValueError) as cm:
            order = Order('Item', 1, 1)
            o2.quantity = -1
```

```python
class Order:
    def __init__(self, name, price, quantity):
        self.name = name
        if price < 0 or quantity < 0:
            raise ValueError('Quantity and price must be positive')
        self.price = price
        self.quantity = quantity

    @property
    def total(self):
        return self.price * self.quantity
```

```python
class TestOrder(TestCase):
    def test_total(self):
        order = Order('Item', 1, 2)
        self.assertEqual(order.total, 2)

    def test_init(self):
        with self.assertRaises(ValueError) as cm:
            order = Order('Item', -1, 2)

    def test_price_non_negative(self):
        with self.assertRaises(ValueError) as cm:
            order = Order('Item', 1, 1)
            order.price = -1

    def test_quantity_non_negative(self):
        with self.assertRaises(ValueError) as cm:
            order = Order('Item', 1, 1)
            o2.quantity = -1
```