AI VIET NAM – COURSE 2024

Object Oriented Programming - Day 13

Ngày 23 tháng 6 năm 2024

Ngày thực hiện:	23/06/2024
Người thực hiện:	Đinh Thị Tâm
Nguồn:	AIO2024 - Day 13
Nguồn dữ liệu (nếu có):	Link Sources code
Từ khóa:	Object Oriented Programming
Người tóm tắt:	Đinh Thị Tâm

1. Mô tả
   Lập trình hướng đối tượng (OOP) là một phương pháp lập trình tập trung vào việc tạo ra các "đối tượng", là những đơn vị cơ bản đại diện cho các thực thể trong thế giới thực. Mỗi đối tượng có các thuộc tính (data) và phương thức (hành vi) riêng. OOP mang lại nhiều lợi ích cho việc lập trình, bao gồm:

   - Tái sử dụng mã: OOP cho phép bạn viết mã có thể được sử dụng lại nhiều lần cho các mục đích khác nhau. Điều này giúp tiết kiệm thời gian và công sức, đồng thời giúp mã dễ dàng bảo trì hơn.
   - Tính linh hoạt: OOP giúp bạn dễ dàng thay đổi và mở rộng mã của mình. Khi bạn cần thêm tính năng mới, bạn chỉ cần tạo ra các đối tượng mới hoặc sửa đổi các đối tượng hiện có.
   - Tính bảo trì: OOP giúp mã của bạn dễ đọc và dễ hiểu hơn. Điều này giúp bạn dễ dàng tìm và sửa lỗi, đồng thời giúp những người khác dễ dàng hiểu mã của bạn.

2. **Bài tập:** Giả sử bạn đang quản lý một cửa hàng bán đồ điện tử. Cửa hàng bán nhiều loại sản phẩm khác nhau, bao gồm điện thoại, máy tính xách tay, tivi, v.v. Mỗi sản phẩm có các thuộc tính chung như tên, giá, nhà sản xuất và số lượng hàng tồn kho.Cửa hàng cũng cần theo dõi các đơn hàng của khách hàng, bao gồm thông tin khách hàng, sản phẩm đã mua và số lượng. Yêu cầu:

   - Tạo một lớp trừu tượng có tên Product để mô tả các thuộc tính chung của tất cả các sản phẩm. Lớp này nên có các phương thức để lấy và đặt tên, giá, nhà sản xuất và số lượng hàng tồn kho.
   - Tạo các lớp con kế thừa từ lớp Product để mô tả các loại sản phẩm cụ thể, chẳng hạn như Phone, Laptop, và TV. Mỗi lớp con nên có các thuộc tính và phương thức bổ sung riêng cho loại sản phẩm đó.
   - Tạo một lớp có tên Order để mô tả một đơn hàng của khách hàng. Lớp này nên có các thuộc tính để lưu trữ thông tin khách hàng, danh sách các sản phẩm đã mua và số lượng của mỗi sản phẩm.

- Thêm các phương thức cho lớp Order để thêm sản phẩm vào đơn hàng, tính toán tổng giá trị của đơn hàng và xuất hóa đơn.
- Viết mã để tạo một số sản phẩm khác nhau, thêm chúng vào đơn hàng và xuất hóa đơn cho đơn hàng đó

(a) Code

```
from abc import ABC, abstractmethod

class Product (ABC):
    def __init__(self, name, price, manufacture, inventory_quality):
        self._name = name
        self._price = price
        self._manufacture = manufacture
        self._inventory_quality = inventory_quality
    # hien thuc 2 phuong thuc __eq__ va __hash__ de kiem tra hai san pham ko
    duoc trung ten

    def __eq__(self, other):
        if isinstance(other, Product):
            return self._name == other._name
        return False

    def __hash__(self):
        return hash(self._name)

    def __str__(self):
        info = '| {} | {} | {} | {} |'.format(
            self._name, self._price, self._manufacture, self.
    _inventory_quality)
        return info

    @abstractmethod
    def describe(self):
        pass

    def get_name(self):
        return self._name

    def set_name(self, name):
        self._name = name

    def get_price(self):
        return self._price

    def set_price(self, price):
        if price < 0:
            print('Price must be granter than or equal zero')
        else:
            self._price = price

    def get_manufacture(self):
        return self._manufacture

    def set_manufacture(self, manufacture):
        if manufacture is None:
            print('Manufacture must have value')
        else:
            self._manufacture = manufacture

```

```
    def get_inventory_quality(self):
        return self._inventory_quality

    def set_inventory_quality(self, quality):
        if quality < 0:
            print('inventory_quality must be greater than or equal zero')
        else:
            self._inventory_quality = quality


class Phone(Product):
    def __init__(self, name, price, manufacture, inventory_quality,
    opr_system, camera, ram, color):
        super().__init__(name, price, manufacture, inventory_quality)
        self.__opr_system = opr_system
        self.__camera = camera
        self.__ram = ram
        self.__color = color

    def get_opr_system(self):
        return self._opr_system

    def set_opr_system(self, opr_system):
        self._opr_system = opr_system

    def get_camera(self):
        return self._camera

    def set_camera(self, camera):
        self._camera = camera

    def get_ram(self):
        return self._ram

    def set_ram(self, ram):
        self._ram = ram

    def get_color(self):
        return self._color

    def set_color(self, color):
        self._color = color

    def describe(self):
        info = super().__str__()
        info = info + ' {} | {} | {} | {} |'.format(
            self.__opr_system, self.__camera, self.__ram, self.__color)
        print(info, end=' ')

# class Laptop


class Laptop(Product):
    def __init__(self, name, price, manufacture, inventory_quality, display,
    opr_system, ram, internal_storage, processor):
        super().__init__(name, price, manufacture, inventory_quality)
        self.__opr_system = opr_system
        self.__ram = ram
        self.__internal_storage = internal_storage
        self.__processor = processor
```

```
        self.__display = display

    def get_display(self):
        return self.__display

    def set_display(self, display):
        self.__display = display

    def get_opr_system(self):
        return self.__opr_system

    def set_opr_system(self, opr_system):
        self.__opr_system = opr_system

    def get_ram(self):
        return self.__ram

    def set_ram(self, ram):
        self.__ram = ram

    def get_internal_storage(self):
        return self.__internal_storage

    def set_internal_storage(self, internal_storage):
        self.__internal_storage = internal_storage

    def get_processor(self):
        return self.__processor

    def set_processor(self, processor):
        self.__processor = processor
    # implement descibe

    def describe(self):
        info = super().__str__()
        info = info + ' {} | {} | {} | {} | {} |'.format(
            self.__display, self.__ram, self.__opr_system, self.__processor,
    self.__internal_storage)
        print(info, end=' ')

# class TV


class TV(Product):
    def __init__(self, name, price, manufacture, inventory_quality,
    screen_size, resolution, display_tech, opr_system):
        super().__init__(name, price, manufacture, inventory_quality)
        self.__screen_size = screen_size
        self.__resolution = resolution
        self.__display_tech = display_tech
        self.__opr_system = opr_system

    def get_screen_size(self):
        return self.__screen_size

    def set_screen_size(self, screen_size):
        self.__screen_size = screen_size

    def get_resolution(self):
        return self.__resolution
```

```
168
169     def set_resolution(self, resolution):
170         self.__resolution = resolution
171
172     def get_display_tech(self):
173         return self.__display_tech
174
175     def set_display_tech(self, display_tech):
176         self.__display_tech = display_tech
177
178     def get_opr_system(self):
179         return self.__opr_system
180
181     def set_opr_system(self, opr_system):
182         self.__opr_system = opr_system
183
184     def describe(self):
185         info = super().__str__()
186         info = info + ' {} | {} | {} | {} |'.format(
187             self.__screen_size, self.__resolution, self.__display_tech, self.
    __opr_system)
188         print(info, end=' ')
189
190 # class Custommer
191
192
193 class Custommer:
194     def __init__(self, id_cus, full_name, cus_phone):
195         self.__id_cus = id_cus
196         self.__full_name = full_name
197         self.__cus_phone = cus_phone
198
199     def get_id_cus(self):
200         return self.__id_cus
201
202     def set_id_cus(self, id_cus):
203         self.__id_cus = id_cus
204
205     def get_full_name(self):
206         return self.__full_name
207
208     def set_full_name(self, full_name):
209         self.__full_name = full_name
210
211     def get_cus_phone(self):
212         return self.__cus_phone
213
214     def set_cus_phone(self, cus_phone):
215         self.__cus_phone = cus_phone
216
217
218 class Order(Custommer):
219     def __init__(self, id_cus, full_name, cus_phone):
220         super().__init__(id_cus, full_name, cus_phone)
221         self.__products = {}
222
223     def add_product(self, product, quality):
224         # kiem tra xem san pham nay co trong hoa don hay chua, neu co thi
    cong don so luong
225         if product in self.__products:
```

```
            self.__products[product] += quality
        else:
            self.__products[product] = quality

    def get_products(self):
        return self.__products

    def caculate_total_price(self):
        total_value = 0
        for pro, key in self.__products.items():
            total_value += pro.get_price() * key
        return total_value

    def show_list_product(self):
        for pro, key in self.__products.items():
            pro.describe()
            print(f' | quality:  {key}|')


# test
p1 = Phone('IP14', 27500000, 'Apple', 10, 'IOS', 'Camera 15', 128, 'White')
laptop1 = Laptop('Dell', 17500000, 'Dell', 25, 15, 'Windows', 16, 512, 'Intel
    ')
tv1 = TV('Samsung 51AO1', 11500000, 'Sam Sung',
         5, 50, 'Full HD', 'LCD', 'Android 4.0')
order = Order('k001', 'an khang', '1234566787')
order.add_product(p1, 2)
order.add_product(laptop1, 1)
order.add_product(tv1, 1)
laptop2 = Laptop('Dell', 17500000, 'Dell', 25, 15, 'Windows', 16, 512, 'Intel
    ')
order.add_product(laptop2, 3)
print('*'*80)
print('Product list in order:')
order.show_list_product()
print('*'*80)
print('\tTotal of bill: {:,}'.format(order.caculate_total_price()))
print('*'*80)
```

(b) Kết quả thực thi chương trình

```
************************************************************
Product list in order:
| IP14 | 27500000 | Apple | 10 | IOS | Camera 15 | 128 | White |  | quality:  2|
| Dell | 17500000 | Dell | 25 | 15 | 16 | Windows | Intel | 512 |  | quality:  4|
| Samsung 51AO1 | 11500000 | Sam Sung | 5 | 50 | Full HD | LCD | Android 4.0 |
   | quality:  1|
************************************************************
        Total of bill: 136,500,000
************************************************************
```