**Java core Review Exercises Part II**

**(Assign date: May 25, 2022)**

**Problem 1:**

**Hàng hóa** trong kho của một siêu thị gồm có **hàng thực phẩm, hàng sành sứ** và **hàng điện máy**.

Mỗi loại hàng đều có: **mã hàng** *(không được sửa, không được để trống)*, **tên hàng** *(không được rỗng, mặc định là “xxx”)*, **đơn giá** *(>=0),* **số lượng tồn** *(>=0)*.

Hàng thực phẩm thì cần quan tâm đến thông tin: **nhà cung cấp**, **ngày sản xuất** (*phải* *trước ngày hiện tại, mặc định là ngày hiện tại*) và **ngày hết hạn** *(phải sau ngày sản xuất, mặc định là ngày sản xuất)*.

Hàng điện máy cần biết: **thời gian bảo hành** bao nhiêu tháng *(>=0)*, **công suất** bao nhiêu KW *(>=0)*.

Hàng sành sứ thì cần biết thông tin về: **nhà sản xuất** và **ngày nhập kho** (*phải* *trước ngày hiện tại, mặc định là ngày hiện tại*).

Ngoài ra, người quản lý cần quan tâm đến số lượng tồn kho và các yếu tố khác của từng loại hàng hóa để **đánh giá mức độ bán buôn**, tiền **VAT** từng loại hàng hóa. *Biết rằng VAT của hàng điện máy và sành sứ là 10%, VAT của hàng thực phẩm là 5%.*

a) Dựa vào các thông tin trên, hãy xác định: o Các lớp có thể có, lớp nào là lớp trừu tượng (abstract class), lớp nào là lớp cụ thể. o Các thuộc tính cho từng lớp. o Các phương thức cho từng lớp (phương thức nào là phương thức trừu tượng (abstract method), danh sách các tham số có thể có cho từng phương thức và kiểu trả về của phương thức).

o Thiết kế mô hình lớp (xây dựng cây thừa kế, các interface nếu có).

b) Thực hiện cài đặt cho mỗi loại hàng cụ thể trên. Trong đó, để **đánh giá mức độ bán buôn**

thì:

* Hàng thực phẩm, nếu vẫn còn tồn kho và bị hết hạn thì đánh giá là *khó bán*. o Hàng điện máy, nếu số lượng tồn kho <3 thì được đánh giá là *bán được*.
* Hàng sành sứ, nếu số lượng tồn kho >50 và thời gian lưu kho >10 ngày thì đánh giá là *bán chậm*. o Các trường hợp còn lại xem như *không đánh giá*.

c) Hãy viết lớp quản lý **danh sách hàng hóa**. Dùng mảng để lưu trữ danh sách hàng hóa.

* Tạo constructor **khởi tạo** danh sách n phần tử. o Viết phương thức **thêm** một hàng hóa vào danh sách *(thêm thành công nếu không bị trùng mã hàng).*
* Viết phương thức **in** **toàn bộ** danh sách các hàng hóa.
* Viết các phương thức **in** **từng loại** hàng hóa. o Viết phương thức **tìm kiếm** hàng hóa khi biết mã hàng (trả về hàng hóa tìm thấy). o Viết phương thức **sắp xếp** hàng hóa theo tên hàng tăng dần. o Viết phương thức **sắp xếp** hàng hóa theo số lượng tồn giảm dần.
* Viết phương thức **xuất** các hàng thực phẩm khó bán. o Viết các phương thức **xóa** hàng hóa khi biết mã hàng.
* Viết các phương thức **sửa** thông tin đơn giá của hàng hóa khi biết mã hàng.

d) Tạo lớp cho phần thử nghiệm, với menu lựa chọn để thực hiện các chức năng trong câu c.

**Problem 2:**

## Cài đặt lớp chứa danh sách

1. Cài đặt lớp **CongNhan**, biết:

* *maCN, mHo, mTen, mSoSP* lần lượt là các thuộc tính họ, tên và số sản phẩm (>0) của công nhân.
* Viết các phương thức getter/setter cho các thuộc tính của lớp.
* Viết các phương thức khởi tạo (mặc định và đầy đủ tham số).
* Viết phương thức *tinhLuong()* để tính lương cho công nhân, lương = số sản phẩm \* đơn giá, với đơn giá tính theo từng cấp như sau:

|  |  |
| --- | --- |
| Số sản phẩm | Đơn giá |
| Từ 1 – 199 sản phẩm | 0.5 |
| Từ sản phẩm thứ 200 – 399 | 0.55 |
| Từ sản phẩm thứ 400 – 599 | 0.6 |
| Từ sản phẩm thứ 600 trở lên | 0.65 |

* Viết phương thức toString trả về thông tin của một công nhân.

1. Cài đặt lớp **DanhSachCongNhan**, có:

* Phương thức khởi tạo n phần tử cho lớp **DanhSachsCongNhan**.
* Phương thức thêm một công nhân vào danh sách, thêm thành công nếu kích thước còn cho phép.
* Phương thức xuất toàn bộ thông tin công nhân viên.
* Phương thức tính số lượng công nhân viên có trong danh sách.
* Phương thức xuất thông tin các công nhân làm trên 200 sản phẩm.
* Phương thức sắp xếp công nhân theo số sản phẩm giảm dần.

Viết hàm main sử dụng lớp **DanhSachCongNhan** theo yêu cầu sau: Cho nhập thông tin một số công nhân vào chương trình, thực thi các yêu cầu trong câu b với các đối tượng này.

**Problem 3:**

Tại một trung tâm đào tạo, người quản lý muốn lưu trữ thông tin học viên của các khóa học, biết một khóa học có thể có tối đa 20 học viên. Thông tin học viên cần lưu trữ là: họ tên, địa chỉ, số điện thoại; thông tin khóa học là: tên khóa học, ngày mở khóa học, thời gian học, danh sách học viên. Ngoài ra người quản lý còn có nhu cầu muốn: tìm kiếm khóa học; biết một học viên đã từng học khóa học nào; các khóa học nào chưa kết thúc; các khóa học nào chưa bắt đầu…

Yêu cầu:

1. Vẽ mô hình lớp và cài đặt các lớp đó để quản lý thông tin học viên và khóa học như mô tả trên.
2. Viết hàm main để thực hiện theo nhu cầu của người quản lý.

**Problem 4:**

Viết chương trình quản lý danh sách các giao dịch. Hệ thống quản lý 2 loại giao dịch:

* Giao dịch vàng: Mã giao dịch, ngày giao dịch, đơn giá, số lượng, loại vàng. Thành tiền tính như sau:

Thành tiền = số lượng \* đơn giá.

* Giao dịch tiền tệ: Mã giao dịch, ngày giao dịch, đơn giá, số lượng, tỉ giá, loại tiền tệ có 3 loại: tiền Việt Nam, tiền USD, tiền Euro. Thành tiền tính như sau:

- Nếu là tiền USD hoặc Euro thì: thành tiền = số lượng \* đơn giá \* tỉ giá.

- Nếu là tiền VN thì: thành tiền = số lượng \* đơn giá.

Thực hiện các yêu cầu sau:

1. Vẽ mô hình và cài đặt các lớp với chức năng thừa kế.
2. Trong hàm main, tạo sẵn mỗi loại 3 giao dịch vào danh sách (không cần cho nhập từ bàn phím), sau đó:
   * Tính tổng số lượng cho từng loại.
   * Tính trung bình thành tiền của giao dịch tiền tệ.
   * Xuất ra các giao dịch có đơn giá > 1 tỷ.

**Problem 5:**

Một khách sạn X cần quản lý các hóa đơn của khách hàng thuê phòng. Hóa đơn có 2 loại: hóa đơn theo giờ, hóa đơn theo ngày. Thông tin chung của chi tiết hóa đơn là: *mã hóa đơn, ngày hóa đơn, tên khách hàng, mã phòng, đơn giá*. Thông tin riêng của từng loại hóa đơn gồm:

* Hóa đơn theo giờ còn có *số giờ thuê*. Thành tiền = số giờ thuê\*đơn giá. Nếu trường hợp số giờ > 24 tiếng và < 30 tiếng thì cũng chỉ tính 24 giờ. Nếu trường hợp số giờ > 30 tiếng thì không dùng loại hóa đơn theo giờ (phát sinh ngoại lệ).
* Hóa đơn theo ngày sẽ có *số ngày thuê*. Thành tiền = số ngày thuê\*đơn giá. Nếu số ngày > 7 thì giảm 20% đơn giá cho những ngày còn lại.

Thực hiện các yêu cầu sau:

1. Vẽ mô hình và cài đặt các lớp với chức năng thừa kế theo mô tả trên.
2. Cài đặt thêm lớp chứa danh sách các hóa đơn theo mô hình sau:

|  |
| --- |
| **DanhSachHoaDon** |
| - danhSach : HoaDon[]  Lưu số lượng hóa đơn hiện tại đang có trong danh sách.  - count : int |
| + DanhSachHoaDon(n: int)  Trả về *true* nếu thêm thành công (không trùng mã\*).  + them(hd: HoaDon) : boolean  + xuat() : void  Xuất toàn bộ các hóa đơn theo dạng bảng, mỗi hóa đơn trên một dòng.  + thongKeSoLuongHDTheoGio() : int  + thongKeSoLuongHDTheoNgay() : int  + tinhTongThanhTien(thang : int, nam : int) : double |

*(\*) Yêu cầu override phương thức equals của lớp Object để kiểm tra trùng mã.*

Tạo menu case trong hàm main cho phép thực hiện các chức năng trong câu b.

**Problem 6:**

Giả sử cần xây dựng chương trình quản lý dùng cho một học viện nghiên cứu giảng dạy và ứng dụng. Đối tượng quản lý bao gồm các sinh viên đang theo học, các nhân viên đang làm việc tại học viện, các khách hàng đến giao dịch mua bán sản phẩm ứng dụng. Dựa vào một số đặc tính của từng đối tượng, người quản lý cần đưa ra cách thức đánh giá khác nhau.

Hãy xây dựng các lớp sau:

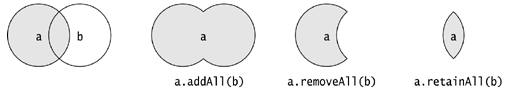
1. Lớp **Person**: bao gồm các thành phần *họ tên, địa chỉ,* phương thức *toString*.
2. Các lớp **Student, Employee, Customer** (theo mô tả bên dưới) thừa kế lớp **Person**.
   * Lớp ***Student:*** bao gồm các thuộc tính *điểm môn học 1, điểm môn học 2*; các phương thức: *tính điểm trung bình, đánh giá*, *toString* trả về bảng điểm sinh viên (gồm thông tin thuộc tính và điểm trung bình).
   * Lớp ***Employee:*** bao gồm thuộc tính *hệ số lương*; các phương thức: *tính lương, đánh giá*, *toString* trả về bảng lương cho nhân viên (gồm thông tin thuộc tính và tiền lương).
   * Lớp ***Customer:*** bao gồm thuộc tính *tên công ty, trị giá hóa đơn, đánh giá*; phương thức *toString* trả về thông tin hóa đơn cho khách hàng (gồm các thuộc tính của đối tượng).
3. Lớp **Management** lưu thông tin toàn bộ các sinh viên, nhân viên, khách hàng và tổng số người hiện tại có trong danh sách. Ngoài ra còn có các phương thức:
   * Constructor khởi tạo mảng với n phần tử.
   * Thêm một người vào danh sách.
   * Xóa một người khỏi danh sách (nhận thông số là họ tên của người cần xóa).
   * Sắp xếp danh sách theo thứ tự họ tên.
   * Xuất danh sách theo dạng bảng.

Viết lớp Test có hàm main cho phần kiểm nghiệm. Giao tiếp với người dùng bằng menu (thể hiện tính đa hình – polymorphism bằng cách cho phép lựa chọn nhập thông tin là sinh viên, nhân viên hay khách hàng).

**Problem 7:**

Viết chương trình tạo hai tập hợp số nguyên (Set). Tính giao, hội, hiệu hai tập trên, xuất kết quả tăng dần.

HD: Dùng TreeSet



* + - a.addAll(b) 🡪 tập a hội tập b
    - a.removeAll(b) 🡪 tập a trừ tập b
    - a.retainAll(b) 🡪 tập a giao tập b

**Problem 8:**

Phòng học được quản lý trong một trường đại học gồm: phòng học lý thuyết, phòng máy tính và phòng thí nghiệm. Mỗi phòng học đều có *mã phòng, dãy nhà, diện tích, số bóng đèn*. Ngoài ra còn:

* + Phòng học lý thuyết thì cần quan tâm xem *có máy chiếu không*.
  + Phòng máy tính thì cần biết là trang bị *bao nhiêu máy tính*.
  + Phòng thí nghiệm thì thêm thông tin *chuyên ngành, sức chứa, có bồn rửa không* (để rửa dụng cụ thí nghiệm/rửa tay).

Thêm nữa, người quản lý cần phải xem xét phòng học *có đạt chuẩn không*. Phòng học đạt chuẩn nếu: tất cả các phòng đều phải đủ ánh sáng *(trung bình 10m2 - 1 bóng đèn)*, và:

* + Phòng lý thuyết: phải có máy chiếu.
  + Phòng máy tính: trung bình 1.5m2 đặt một máy.
  + Phòng thí nghiệm: phải có bồn rửa đi kèm.

Yêu cầu:

1. Thiết kế và thực hiện cài đặt tường minh cho mỗi loại phòng được mô tả trên.
2. Viết lớp **quản lý danh sách phòng học**. Yêu cầu dùng một List (ArrayList, LinkedList, Vector) để lưu trữ danh sách phòng học. Trong đó:
   * Tạo constructor **khởi tạo** danh sách.
   * Viết phương thức **thêm** một phòng học vào danh sách *(thêm được nếu không trùng mã phòng)*.
   * Viết phương thức **tìm kiếm** một phòng học nào đó khi biết mã phòng.
   * Viết phương thức **lấy thông tin toàn bộ** danh sách các phòng học.
   * Viết các phương thức để **lấy thông tin các phòng học đạt chuẩn**.
   * Viết phương thức để **sắp xếp danh sách tăng dần theo dãy nhà**.
   * Viết phương thức để **sắp xếp danh sách giảm dần theo diện tích**.
   * Viết phương thức để **sắp xếp danh sách tăng dần theo số bóng đèn**.
   * Viết phương thức để **cập nhật** số máy tính cho một phòng máy tính nào đó khi biết mã phòng.
   * Viết phương thức để **xóa** một phòng học nào đó khi biết mã phòng. *Lưu ý khi test chương trình, khi xóa cần phải xác minh rằng có chắc chắn xóa không.*
   * Viết phương thức để **tính tổng số phòng học**.
   * Viết các phương thức để **lấy danh sách các phòng máy có 60 máy**.

Tạo lớp cho phần thử nghiệm, với menu lựa chọn để thực hiện các chức năng theo yêu cầu.

**Problem 9:**

Hiện thực theo mô hình lớp bên dưới bằng ngôn ngữ lập trình Java.



Đây là một phần ứng dụng quản lý nhân viên trong một công ty phần mềm *(SoftwareHouse)*. Công ty có nhiều nhân viên. Mỗi nhân viên *(Employee)* cần lưu trữ các thông tin: Mã số *(thePayrollNumber)*, tên nhân viên *(theName)*, lương cơ bản hằng tháng *(theBasicMonthlySalary)*. Mã số của mỗi nhân viên chỉ được tạo một lần duy nhất và không được phép sửa.

1. Nhân viên trong công ty gồm: Lập trình viên *(Programmer)*, Người quản lý *(Administrator)* và Người trưởng dự án *(ProjectLeader).*

* Tạo các phương thức setters/getters cho các thuộc tính của các lớp.
* Tiền lương hằng tháng *(getMonthlySalary)* là tiền lương cơ bản + phụ cấp. Phụ cấp được tính như sau:
  + Lập trình viên: Nếu ngôn ngữ lập trình là “Java” thì phụ cấp 20% của lương cơ bản.
  + Người quản lý: Phụ cấp 40% của lương cơ bản
  + Người trưởng dự án: Phụ cấp 20% lương cơ bản.

1. Công ty phần mềm *(SoftwareHouse)* có tên gọi *(theName)*, và danh sách các nhân viên. Tạo:
2. Constructor *SofwareHouse(aName : String)*, tạo một công ty có tên là *aName* và khởi tạo danh sách nhân viên (mỗi phần tử trong mảng là một nhân viên).
3. Phương thức *addEmployee(emp : Employee): boolean*, dùng để thêm một nhân viên *emp* vào công ty. Thêm thành công nếu không trùng mã số.
4. Phương thức *displayAllStaff(): void*, hiển thị toàn bộ nhân viên trong công ty lên màn hình theo dạng cột, định dạng đơn vị tiền tệ là $, phân cách hàng nghìn.
5. Phương thức *getMonthlySalaryBill(): double*, tính tổng tiền phải trả cho các nhân viên.
6. Phương thức *sortAllStaffByName(): void*, sắp xếp danh sách nhân viên theo tên.
7. Phương thức *displayListProgrammers(): void*, hiển thị danh sách các lập trình viên.
8. Phương thức *updateDepartmentForAdmin(aPayrollNo: int, deptNew: String): boolean*, cập nhật phòng ban là *deptNew* cho người quản lý có mã số là *aPayrollNo*, trả về *true* nếu cập nhật thành công.
9. Phương thức *deleteEmployee(id : int): boolean,* xóa nhân viên theo mã số, trả về *true* nếu xóa được.

Viết chương trình chính tạo menu case để thực hiện các chức năng trong câu b.

**Problem 10:**

Quản lý khách hàng xếp hàng mua vé tại nhà ga. Thông tin lưu trữ cho khách hàng gồm: *số CMND (String), tên khách hàng, ga đến, giá tiền (double)*.

Tạo hệ thống menu gồm các mục:

* Thêm một khách hàng mới vào hàng đợi mua vé.
* Bán một vé cho khách hàng. Chỉ bán cho người đăng ký trước.
* Hiển thị danh sách khách hàng.
* Hủy một khách hàng ra khỏi danh sách (khách hàng không mua vé nữa).
* Thống kê tình hình bán vé.
* Lưu danh sách vào file.
* Hiển thị danh sách các ga đang chờ mua vé.
* Hiển thị danh sách các ga đang chờ mua vé và số vé tương ứng cho ga.

Lưu ý:

* Số khách hàng trong danh sách hiện tại là số khách đang chờ, nhưng chưa có vé. Khi một khách hàng đã mua vé, thì loại khách hàng này ra khỏi danh sách chờ mua vé.
* Việc mua vé phải có thứ tự: ai vào trước thì mua vé trước (FIFO).
* Mỗi khi khách hàng mua được vé phải lưu lại khách hàng này để dùng cho việc thống kê.
* Mỗi khi thêm một khác hàng mới, nếu số CMND khách hàng đã có thì không tạo phần tử mới mà chỉ cập nhật lại ga và giá tiền đến cho khác hàng đó.
* Mục thống kê tình hình: cho biết còn bao nhiêu khách hàng chờ nhận vé, bao nhiêu khách hàng đã nhận vé, tổng số tiền đã thu về là bao nhiêu.
* Việc lưu danh sách: chỉ lưu các khách hàng chờ mua vé. Các khách hàng đã nhận vé xem như kết sổ trong ngày không cần lưu lại.
* Khi chương trình vừa được chạy, lập tức tự động nạp toàn bộ danh sách khách hàng từ file (cách khách hàng chưa có vé).

Khi hiển thị danh sách các ga đến đang chờ mua vé, chỉ hiển thị tên ga đó một lần. (Ví dụ: giả sử 10 khách hàng nhưng đăng ký đi đến 2 ga, thì chỉ hiển thị 2 hàng).