**Tổng hợp bài tập lập trình hướng đối tượng trong java**

Khi đã có kiến thức về [lập trình hướng đối tượng trong Java](https://shareprogramming.net/oop-java-step-by-step/) thì đây là chính là lúc chúng ta cần thực hành để vận dụng kiến thức để hiểu sâu hơn về hướng đối tượng cũng như lấy được kinh nghiệm thực tế từ các bài tập.

Dưới đây là loạt bài tổng hợp có lời giải hoặc hướng dẫn cụ thể.

**Bài 1**

Một đơn vị sản xuất gồm có các cán bộ là công nhân, kỹ sư, nhân viên. Mỗi cán bộ cần quản lý các dữ liệu: Họ tên, tuổi, giới tính(name, nữ, khác), địa chỉ.

* Cấp công nhân sẽ có thêm các thuộc tính riêng: Bậc (1 đến 10).
* Cấp kỹ sư có thuộc tính riêng: Nghành đào tạo.
* Các nhân viên có thuộc tính riêng: công việc.

Yêu cầu 1: Xây dựng các lớp *CongNhan*, *KySu*, *NhanVien* kế thừa từ lớp *CanBo*.

Yêu cầu 2: Xây dựng lớp *QLCB*(quản lý cán bộ) cài đặt các phương thức thực hiện các chức năng sau:

* Thêm mới cán bộ.
* Tìm kiếm theo họ tên.
* Hiện thị thông tin về danh sách các cán bộ.
* Thoát khỏi chương trình.

[Source code tham khảo](https://gitlab.com/excercise-oop/synthetic_exercise_oop_java/tree/master/src/b1)

**Bài 2**

Một thư viện cần quản lý các tài liệu bao gồm Sách, Tạp chí, Báo. Mỗi tài liệu gồm có các thuộc tính sau: Mã tài liệu(Mã tài liệu là duy nhất), Tên nhà xuất bản, số bản phát hành.

Các loại sách cần quản lý thêm các thuộc tính: tên tác giả, số trang.

Các tạp chí cần quản lý thêm: Số phát hành, tháng phát hành.

Các báo cần quản lý thêm: Ngày phát hành.

Yêu cầu 1: Xây dựng các lớp để quản lý tài liệu cho thư viện một cách hiệu quả.

Yêu cầu 2: Xây dựng lớp *QuanLySach* có các chức năng sau

* Thêm mới tài liêu: Sách, tạp chí, báo.
* Xoá tài liệu theo mã tài liệu.
* Hiện thị thông tin về tài liệu.
* Tìm kiếm tài liệu theo loại: Sách, tạp chí, báo.
* Thoát khỏi chương trình.

[Source code tham khảo](https://gitlab.com/excercise-oop/synthetic_exercise_oop_java/tree/master/src/b2)

**Bài 3**

Các thí sinh dự thi đại học bao gồm các thí sinh thi khối A, B, và khối C. Các thí sinh cần quản lý các thông tin sau: Số báo danh, họ tên, địa chỉ, mức ưu tiên.

Thí sinh thi khối A thi các môn: Toán, Lý, Hoá.

Thí sinh thi khối B thi các môn: Toán, Hoá, Sinh.

Thí sinh thi khối C thi các môn: Văn, Sử, Địa.

Yêu cầu 1: Xây dựng các lớp để quản lý các thi sinh dự thi đại học.

Yêu cầu 2: Xây dựng lớp *TuyenSinh* có các chức năng:

* Thêm mới thí sinh.
* Hiện thị thông tin của thí sinh và khối thi của thí sinh.
* Tìm kiếm theo số báo danh.
* Thoát khỏi chương trình.

[Source code tham khảo](https://gitlab.com/excercise-oop/synthetic_exercise_oop_java/tree/master/src/b3)

**Bài 4**

Để quản lý các hộ dân cư trong một khu phố, người ta cần các thông tin sau: Số thành viên trong gia đình, Số nhà, thông tin mỗi cá nhân trong gia đình. Với mỗi cá nhân, người ta quản lý các thông tin sau: Họ tên, Tuổi, Nghề nghiệp, số chứng minh nhân dân(duy nhất cho mỗi người).

Yêu cầu 1: Hãy xây dựng lớp *Nguoi* để quản lý thông tin của mỗi cá nhân.

Yêu cầu 2: Xây dựng lớp *HoGiaDinh*để quản lý thông tin của từng hộ gia đình.

Yêu cầu 2: Xây dựng lớp *KhuPho* để quản lý các thông tin của từng hộ gia đình.

Yêu cầu 3: Nhập n hộ dân. (n nhập từ bàn phím), hiển thị thông tin của các hộ trong khu phố.

[Link source tham khảo](https://gitlab.com/excercise-oop/synthetic_exercise_oop_java/tree/master/src/b4)

**Bài 5**

Để quản lý khách hàng đến thuê phòng của một khách sạn, người ta cần các thông tin sau: Số ngày thuê, loại phòng, thông tin cá nhân của những người thuê phòng.

Biết rằng phòng loại A có giá 500$, phòng loại B có giá 300$ và loại C có giá 100$.

Với mỗi cá nhân cần quản lý các thông tin sau: Họ tên, tuổi, số chứng minh nhân dân.

Yêu cầu 1: Hãy xây dựng lớp *Nguoi*để quản lý thông tin cá nhân của những người thuê phòng.

Yêu cầu 2: Xây dựng lớp *KhachSan*để quản lý các thông tin về các phòng trong khác sạn.

Yêu cầu 3: Xây dựng các phương thức thêm mới, xoá khách theo số chứng minh nhân dân. Tính tiền thuê phòng cho khách(xác định khách bằng số chứng minh nhân dân) dựa vào công thức: (số ngày thuê \* giá của từng loại phòng)

[Link source code tham khảo](https://gitlab.com/excercise-oop/synthetic_exercise_oop_java/tree/master/src/b5)

**Bài 6**

Để quản lý hồ sơ học sinh của trường THPT nhà trường cần các thông tin sau: Lớp,  và các thông tin về cá nhân của mỗi học sinh.

Mỗi học sinh có các thông tin sau: Họ tên, tuổi, quê quán.

Yêu cầu 1: Xây dựng *HocSinh* để quản lý thông tin của mỗi học sinh.

Yêu cầu 2: Xây dựng các phương thức thêm, hiển thị thông tin của mỗi học sinh.

Yêu cầu 3: Cài đặt chương trình có các chức năng sau:

* Thêm học sinh mới.
* Hiện thị các học sinh 20 tuổi.
* Cho biết số lượng các học sinh có tuổi là 23 và quê ở DN.

[Link source code tham khảo](https://gitlab.com/excercise-oop/synthetic_exercise_oop_java/tree/master/src/b6)

**Bài 7**

Khoa CNTT – DHKHTN cần quản lý việc thanh toán tiền lương cho các cán bộ giá viên trong khoa. Để quản lý được, khoa cần các thông tin sau:

Với mỗi cán bộ giáo viên có các thông tin sau: lương cứng, lương thưởng, tiền phạt, lương thực lĩnh, và các thông tin cá nhân: Họ tên, tuổi, quê quán, mã số giáo viên.

Yêu cầu 1: Xây dựng lớp *Nguoi*để quản lý các thông tin cá nhân của mỗi giáo viên.

Yêu cầu 2: Xây dựng lớp *CBGV*để quản lý các thông tin của các cán bộ giáo viên.

Yêu cầu 3: Xây dựng các phương thức thêm, xoá các cán bộ giáo viên theo mã số giáo viên.

Yêu cầu 4: Tính lương thực lĩnh cho giáo viên: Lương thực = Lương cứng + lương thưởng – lương phạt.

[Link source code tham khảo](https://gitlab.com/excercise-oop/synthetic_exercise_oop_java/tree/master/src/b7)

**Bài 8**

Thư viện trung tâm đại học quốc gia có nhu cầu quản lý việc mượn, trả sách. Sinh viên đăng ký tham gia mượn sách thông qua thẻ mà thư viện cung cấp.

Với mỗi thẻ sẽ lưu các thông tin sau: Mã phiếu mượn, ngày mượn, hạn trả, số hiệu sách, và các thông tin cá nhân của sinh viên mượn sách. Các thông tin của sinh viên mượn sách bao gồm: Họ tên, tuổi, lớp.

Để đơn giản cho ứng dụng console. Chúng ta mặc định ngày mượn, ngày trả là số nguyên dương.

Yêu cầu 1: Xây dựng lớp *SinhVien*để quản lý thông tin của mỗi sinh viên.

Yêu cầu 2: Xây dựng lớp TheMuon để quản lý việc mượn trả sách của các sinh viên.

Yêu cầu 3: Xây dựng các phương thức: Thêm, xoá theo mã phiếu mượn và hiển thị thông tin các thẻ mượn.

[Link source code tham khảo](https://gitlab.com/excercise-oop/synthetic_exercise_oop_java/tree/master/src/b8)

**Bài 9**

Bài tập tự luyện

Để quản lý biên lai thu tiền điện, người ta cần các thông tin sau:

* Với mỗi biên lai: Thông tin về hộ sử dụng điện, chỉ số điện cũ, chỉ số mới, số tiền phải trả.
* Các thông tin riêng của từng hộ gia đình sử dụng điện: Họ tên chủ hộ, số nhà, mã số công tơ điện.

Yêu cầu 1: Hãy xây dựng lớp *khachHang*để lưu trữu các thông tin riêng của mỗi hộ gia đình.

Yêu cầu 2: Xây dựng lớp *BienLai*để quản lý việc sử dụng và thanh toán tiền điện của các hộ dân.

Yêu cầu 3: Xây dựng các phương thức thêm, xoá sửa các thông tin riêng của mỗi hộ sử dụng điện.

Yêu cầu 4: Viết phương thức tính tiền điện cho mỗi hộ gia đình theo công thức: (số mới – số cũ ) \* 5.

**Bài 10**

Bài tập tự luyện

Để xử lý văn bản người ta xây dựng lớp *VanBan* có thuộc tính riêng là một xâu ký tự.

Yêu cầu 1: Xây dựng hàm khởi tạo VanBan(), VanBan(String st).

Yêu cầu 2: Xây dựng phương thức đếm số từ của văn bản.

Yêu cầu 3: Xây dựng phương thức đếm số lượng ký tự A( không phân biệt hoa thường) của văn bản.

Yêu cầu 4: Chuẩn hoá văn bản theo tiêu chuẩn sau: Ở đầu và cuối sâu không có ký tự trống, ở giữa sâu không có 2 hoặc nhiều hơn các ký tự khoảng trắng kiền kề nhau.

**Bài 11**

Bài tập tự luyện

Xây dựng lớp *SoPhuc* có các thuộc tính *PhanThuc*, *PhanAo* kiểu double.

Yêu cầu 1: Xây dựng các phương thức tạo lập

Yêu cầu 2: Xây dựng các phương thức:

* Nhập một số phức.
* Hiện thị số phức.
* Cộng 2 số phức.
* Nhân 2 số phức.

**Bài 12**

Bài tập tự luyện

Nghành công an cần quản lý các phương tiện giao thông gồm: ô tô, xe máy, xe tải. Mỗi loại gồm các thông tin: ID, Hãng sản xuất, năm sản xuất, giá bán và màu xe.

Các ô tô có các thuộc tính riêng: số chỗ ngồi, kiểu động cơ.

Các xe máy có các thuộc tính riêng: công xuất.

Xe tải cần quản lý thêm: Trọng tải.

Yêu cầu 1: Xây dựng các lớp để quản lý các phương tiện trên sao cho hiệu quả.

Yêu cầu 2: Xây dựng lớp QLPTGT có các chức năng:

* Thêm, xoá(theo ID) các phương tiện thuộc các loại trên.
* Tìm phương tiện theo hãng sản xuất, màu.
* Thoát chương trình.

**Bài 13**

Phần mềm quản lý nhân viên của công ty được mô tả nghiệp vụ như sau:

Mỗi nhân viên (**Employee**) được công ty chia thành 3 loại sau: Nhân viên có kinh nghiệm lâu năm (**Experience**), nhân viên mới ra trường (**Fresher**) , Nhân viên thực tập (**Intern**)

            Tất cả các **Employee**đều có các thuộc tính là: **ID**, **FullName**, **BirthDay**,**Phone**, **Email**,**Employee\_type**,**Employee\_count** và phương thức là **ShowInfo** để hiển thị thông tin của nhân viên đó (hiển thị thông tin nhân viên ra màn hình console).

**Trong đó :**

**Employee\_type**có giá trị tương ứng là 0: Experience, 1: Fresher , 2: Intern  (tùy vào người dùng nhập vào ứng viên loại nào)

**Employee\_count**dùng để người dùng đếm số lượng nhân viên trong một đợt người dùng nhập nhân viên mới  vào cơ sở dữ liệu. (**mỗi lần người dùng nhập thêm mới nhân viên thì thuộc tính Employee\_count của class Employee sẽ tăng lên 1)**

**Ngoài ra :**

            Đối với nhân viên**Experience** có thêm thuộc tính: Số năm kinh nghiệm (ExpInYear), Kỹ năng chuyên môn (ProSkill)

            Đối với nhân viênn**Fresher** có thêm thuộc tính: Thời gian tốt nghiệp(Graduation\_date), Xếp loại tốt nghiệp (Graduation\_rank) , Trường tốt nghiệp (Education)

            Đối với nhân viên**Intern**có thêm thuộc tính: Chuyên ngành đang học (Majors), Học kì đang học (Semester), Tên trường đang học (University\_name)

**Lưu ý:**Tùy mỗi loại nhân viên, phương thức **showMe** sẽ được bổ sung thêm các thuộc tính của riêng loại nhân viên đó.

\*\* Ngoài ra mỗi nhân viên khi vào làm cần phải nộp bằng cấp nghề nghiệp đi kèm, bộ phận tuyển dụng cần quản lý các bằng cấp này. **Một nhân viên có thể có nhiều bằng cấp (Certificate)**

Với mỗi bằng cấp có các thông tin bao gồm : **CertificatedID**, **CertificateName**, **CertificateRank**, **CertificatedDate**.

Yêu cầu

1. Hãy thiết kế và viết code của chương trình trên làm sao để tuân thủ theo đúng mô hình **OOP** đã học, áp dụng đầy đủ 4 tính chất : bao đóng (*encapsulation)*, kế thừa (*inheritance)*, đa hình (*polymorphism)*, trừu tượng (*abstraction).*
2. Xác định và viết code constructor cho tất cả các class.
3. Xác định và viết code cho các **abstract** **method**, **abstract** **class**, **override**/**overload** method, **static** field.
4. Xác định Is A, Has A relationship trong phần thiết kế code đã viết ở trên.
5. Sử dụng và giải thích được ý nghĩa của 2 keyword: **super**, **this** trong phần thiết kế code ở trên.
6. Hãy giải thích ý nghĩa của toán tử [instanceof](https://shareprogramming.net/so-sanh-kieu-voi-instanceof-trong-java/), hãy ứng dụng instanceof để [downcasting](https://shareprogramming.net/co-che-upcasting-va-downcasting-trong-java/) 1 object **Employee trở thành các object Experience, Fresher hoặc intern**trong các yêu cầu 2, yêu cầu 3, yêu cầu 4 bên dưới**.**
7. Viết chức năng cho phép thêm, sửa (Nhập ID để xác định một nhân viên, nếu tồn tại cho phép người dùng chỉnh sửa thông tin của nhân viên), xóa(xóa theo ID) các loại nhân viên trên.
8. Viết các hàm kiểm tra tính hợp lệ của ngày sinh, email, tên và số điện thoại của nhân viên. Áp dụng các hàm này vào chức năng số 7.
9. Viết chương trình tìm tất cả các nhân viên intern.
10. Viết chương trình tìm tất cả các nhân viên experience .
11. Viết chương trình tìm tất cả các nhân viên fresher.
12. Tạo ra các BirthDayException, PhoneException, EmailException, FullNameException để ném ra ngoại lệ trong trường hợp birthday, email, và phone người dùng nhập vào không hợp lệ.

**Bài 14**

Viết chương trình quản lý việc tuyển sinh viên sau khi tốt nghiệp đại học.

Có 2 loại sinh viên bao gồm **sinh viên tốt nghiệp loại khá giỏi (GoodStudent) và sinh viên tốt nghiệp loại trung bình (NormalStudent).**

Cả 2 loại sinh viên trên đều phải cung cấp các thông tin sau khi nộp hồ sơ xin việc: Họ tên **(fullName)**, Ngày tháng năm sinh **(doB)**, Giới tính **(sex)**, Số điện thoại **(phoneNumber)**, Tên trường đã học **(universityName)**, Xếp loại tốt nghiệp **(gradeLevel)**.

Riêng sinh viên loại khá giỏi phải có thêm thông tin: điểm trung bình học tập **(gpa)** theo thang điểm 10, tên của loại học bổng (hoặc giải thưởng) cao nhất đã từng giành được **(bestRewardName)**.

Riêng sinh viên loại trung bình thì phải có thêm các thông tin: điểm TOEIC **(englishScore)**, điểm thi đầu vào chuyên môn do công ty tổ chức thi **(entryTestScore)**theo thang điểm 10.

**Yêu cầu 1:**

Thí sinh hãy thiết kế và viết code thể hiện các class của các **sinh viên** và **lớp học** của chương trình làm sao để tuân thủ theo mô hình **OOP** đã học, áp dụng 4 tính chất : bao đóng (*encapsulation)*, kế thừa (*inheritance)*, đa hình (*polymorphism)*, trừu tượng (*abstraction).*

            Lưu ý: *Bất kì một sinh viên nào cũng cần có một phương thức có tên là****ShowMyInfor****để hiển thị thông tin của sinh viên đó ra màn hình cosole, yêu cầu này là bắt buộc với các thành viên xây dựng code cho chương trình này. Bạn hãy lưu ý khi thiết kế code để thỏa mãn yêu cầu này.*

**Yêu Cầu 2**: Kiểm tra ràng buộc dữ liệu cho chương trình. Dữ liệu của các file input phải tuân thủ theo ràng buộc sau:

– Họ tên sinh viên có chiều dài tối đa là 50 ký tự và tổi thiểu là 10 ký tự. *Học viên cần tạo Exception tương ứng có tên là InvalidFullNameException (1 điểm).*

*–*Chương trình phải bắt được lỗi sai định dạng ngày tháng năm đối với trường doB. doB phải theo định dạng dd/MM/YYYY*. Học viên cần tạo Exception tương ứng có tên là InvalidDOBException (1 điểm).*

– Số điện thoại: phải là chuỗi số có chiều dài 10 ký tự và phải**bắt đầu bằng một trong các chuỗi số: 090, 098, 091, 031, 035 hoặc 038.***Học viên cần tạo Exception tương ứng có tên là InvalidPhoneNumberException (1 điểm).*

– Ngoài ra nếu có bất cứ một exception nào khác trong quá trình thực thi chương trình, thí sinh hãy thông báo ra màn hình nội dung *“Input files have unknow errors !!!”* *(0.5 điểm).*

**Yêu cầu 3:** Chương trình cần có chức năng lựa chọn ứng viên trúng tuyển vào công ty theo nguyên tắc sau:

* Người dùng sẽ nhập vào số lượng sinh viên cần tuyển dụng (tối thiểu là 11, tối đa là 15).
* Chương trình sẽ tự động chọn ra ứng viên phù hợp cho công ty theo các bước như sau:
  + Nếu số lượng ứng viên là sinh viên khá giỏi có nhiều hơn số lượng cần tuyển thì xét ưu tiên theo điểm GPA. Nếu *xuất hiện ứng viên khá giỏi có cùng điểm GPA thì xét ưu tiên theo họ tên. Vd: nếu họ tên là Nguyễn Văn A và Nguyễn Văn B thì ứng viên Nguyễn Văn A được chọn.****(Giả sử không bao giờ có sinh viên có trùng họ tên nhau)****.*
  + Nếu số lượng ứng viên là sinh viên khá giỏi ít hơn hoặc bằng số lượng cần tuyển thì nhận hết sinh viên khá giỏi.
  + Sau khi tuyển hết ứng viên khá giỏi, nếu vẫn chưa đủ số lượng cần tuyển, chương trình sẽ lấy ứng viên trung bình. Các ứng viên trung bình được xét ưu tiên theo điểm thi đầu vào, nếu điểm thi đầu vào bằng nhau thì xét đến điểm TOEIC. *Nếu xuất hiện ứng viên trung bình có cùng điểm TOEIC thì xét ưu tiên theo họ tên*.

Yêu cầu 4: Chương trình có thể hiển thị được thông tin họ tên và số điện thoại của tất cả các sinh viên đã nhập vào hệ thống (yêu cầu dùng collection sort để sắp xếp giảm dần theo **fullName** và tăng dần theo **phoneNumber** đối với sinh viên trước khi hiển thị).

**Bài 15**

Xây dựng chương trình quản lý kết quả học tập của sinh viên tại một trường đại học. Có 2 loại sinh viên là sinh viên chính quy và sinh viên tại chức với các thông tin giống nhau: mã sinh viên, họ tên, ngày tháng năm sinh, năm vào học, điểm đầu vào và danh sách kết quả học tập. Sinh viên tại chức có thêm thông tin nơi liên kết đào tạo(Đồng Nai, Cà Mau, …). Khoa gồm có các thông tin: tên khoa và danh sách sinh viên đang theo học. Kết quả học tập gồm có tên học kỳ, điểm trung bình học kỳ đó.

**Hiện thực các yêu cầu sau:**  
Vẽ class diagram cho phát biểu bài toán trên OOP:

1. Hiện thực các lớp cần thiết cho bài toán trên
2. Phương thức khởi tạo (constructor) cho các lớp: constructor chuẩn, constructor  
   có tham số, constructor sao chép
3. Phương thức input và output để cho phép người dùng nhập thông tin cho các  
   loại sinh viên
4. Phương thức xác định sinh viên có phải là chính quy hay không?
5. Xử lý Exception khi người dùng nhập sai dữ liệu
6. Phương thức lấy điểm trung bình các môn học của sinh viên dựa vào học kỳ  
   cho trước
7. Phương thức xác định tổng số sinh viên chính quy của khoa?
8. Tìm ra sinh viên có điểm đầu vào cao nhất ở mỗi khoa
9. Ở mỗi khoa, lấy ra danh sách các sinh viên tại chức tại nơi liên kết đào tạo cho  
   trước
10. Ở mỗi khoa, lấy ra danh sách sinh viên có điểm trung bình ở học kỳ gần nhất  
    (là học kỳ cuối cùng trong danh sách kết quả học tập của sinh viên) từ 8.0 trở  
    lên
11. Ở mỗi khoa, tìm ra sinh viên có điểm trung bình học kỳ cao nhất (ở bất kỳ học  
    kỳ nào)
12. Ở mỗi khoa, sắp xếp danh sách sinh viên tăng dần theo loại và giảm dần theo  
    năm vào học (sử dụng interface Comparable hoặc Comparator)
13. Ở mỗi khoa, thống kê số lượng sinh viên theo năm vào học. Ví dụ 2020: 100,  
    2019: 90, 2018: 120.