**BÀI TẬP**

**HƯỚNG ĐỐI TƯỢNG TYPESCRIPT**

**(OJBECT ORIENTED PROGRAMMING EXERCISES)**

# CÁC KHÁI NIỆM CƠ BẢN LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

*Mục tiêu:*

* *Cài đặt lớp đối tượng.*
* *Tạo, sử dụng đối tượng.*
* *Định nghĩa getter/setter cho thuộc tính (kiểm tra ràng buộc trên thuộc tính nếu có).*
* *Tạo constructor có kiểm soát tham số truyền cho thuộc tính.*
* *Tạo cơ chế liên lạc giữa các lớp.*
* *Ghi chú cho lớp, phương thức (theo dạng document).*

*Yêu cầu:*

* *Tạo project tên ModulePlus.*
* *Mỗi bài làm trong một package có tên package là bai01, bai02,…*
* *Viết documentation comment cho class.*

## 

1. Cài đặt lớp hình tam giác, biết tam giác có 3 cạnh ma, mb, mc, viết các phương thức sau:

* Constructor mặc định (không tham số).
* Constructor đủ tham số: *nếu giá trị truyền có số âm hoặc nếu 3 giá trị không lập thành hình tam giác thì gán 3 thuộc tính bằng 0*.
* Các phương thức getter/setter: *nếu giá trị không hợp lệ thì không gán (giữ lại giá trị cũ)*.
* Các phương thức tính chu vi, phương thức tính diện tích.
* Phương thức trả về thông tin kiểu tam giác (thường, cân, đều, không phải tam giác).
* Phương thức *toString* để diễn tả đối tượng ở dạng chuỗi gồm: thông tin 3 cạnh, kiểu tam giác, chu vi, diện tích.

1. Viết hàm main tạo 5 hình tam giác: 2 hình vi phạm ràng buộc, 3 hình là tam giác thường, cân, đều. Xuất thông tin các hình này theo dạng bảng.

*HD:*

* *Ba giá trị lập thành một hình tam giác khi và chỉ khi tổng hai cạnh bất kỳ luôn lớn hơn cạnh còn lại.*
* *Công thức tính diện tích tam giác:  với .*

## 

Cho mô tả bài toán:

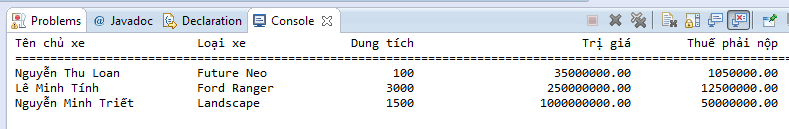
Sở giao thông cần theo dõi việc đăng ký xe (Vehicle) của người dân, biết mỗi xe cần lưu các thông tin là: *chủ xe, loại xe, trị giá xe* (>=0)*, dung tích xylanh* (>=0). Dựa vào thông tin *trị giá xe* và *dung tích xylanh*, sở giao thông cũng tính mức thuế phải đóng trước bạ khi mua xe như sau:

* Dưới 100cc, 1% trị giá xe.
* Từ 100 đến 200cc, 3% trị giá xe.
* Trên 200cc, 5% trị giá xe.

a. Hãy thiết kế và cài đặt lớp Vehicle với các attributes và methods phù hợp (có kiểm tra ràng buộc dữ liệu). Lớp phải có các constructor và phải bảo đảm tính encapsulation.

b. Xây dựng class chứa hàm main, có các công việc:

* Tạo 3 đối tượng Vehicle xe1, xe2, xe3. Dữ liệu được gán sẵn hoặc cho người dùng nhập.
* Xuất bảng kê khai tiền thuế trước bạ của các xe như mẫu:



Mẫu kết xuất của chương trình.

## 

Cài đặt cho lớp tài khoản (Account) được thiết kế như sau:

|  |
| --- |
| Account |
| - accountNumber : long  - name : String  - balance : double  - RATE = 0.035 : final double |
| <<constructors>>  + Account()  + Account(accNumber : long, name : String, balance : double)  + Account(accNumber : long, name : String)  <<property get>>  + getAccountNumber() : long  + getBalance() : double  <<other methods>>  + deposit(amount : double) : boolean  + withdraw(amount : double, fee : double) : boolean  + addInterest() : void  + transfer(acc2 : Account, amount : double): boolean  + toString() : String |

Mô tả lớp Account:

* Ba constructor: **constructor mặc định** (số tài khoản = 999999, tên tài khoản = “chưa xác định”, số dư = 50000); **constructor đầy đủ tham số**; **constructor có hai tham số**. *Trong trường hợp giá trị tham số không hợp lệ thì gán giá trị mặc định cho thuộc tính tương ứng (dữ liệu gán là hợp lệ khi số tài khoản >0, tên tài khoản khác rỗng, số dư >=50000).*
* **deposit**(*amount* : double): cho phép gởi thêm số tiền *amount* vào tài khoản, hàm trả về *true* nếu gởi thành công (*amount*>0).
* **withdraw**(*amount* : double, *fee* : double): cho phép rút số tiền *amount* từ tài khoản, hàm trả về *true* nếu rút tiền thành công (*amount*>0 và *amount+fee* <= *balance*).
* **addInterest**(): tính tiền lãi, *balance* = *balance* + *balance* \* *RATE*.
* **transfer**(*acc2* : Account, *amount* : double): chuyển một khoản tiền *amount* từ *account* này sang *account* kia, trả về true nếu chuyển thành công.
* **toString** trả về chuỗi chứa toàn bộ thông tin tài khoản, yêu cầu định dạng kiểu tiền tệ.

*HD: Định dạng tiền tệ:*

*1. Tạo đối tượng Locale: Xác định ngôn ngữ và quốc gia áp dụng: Locale local= new Locale(“mã NN”, “mã QG”). Ví dụ:*

**Locale** local = new **Locale**(“vi”, “vn”);

*2. Tạo đối tượng NumberFormater với tham số Locale ở trên: Ví dụ:*

**NumberFormat** formatter = **NumberFormat**.*getCurrencyInstance*(local);

*3. Dùng formater để định dạng: ví dụ: formater(456953.12);*

[formater sau khi định dạng sẽ trả về chuỗi số đã định dạng]

(c) . Test lớp Account như sau :

* Tạo 3 đối tượng acc1, acc2, acc3 với các giá trị name, accountNumber, balance lần lượt như sau: {"Ted Murphy", 72354, 102.56}; {"Jane Smith", 69713, 40.00}; {"Edward Demsey", 93757, 759.32}.
* Gởi thêm số tiền cho acc1 là 25.85, cho acc2 là 500.00.
* Rút khỏi acc2 số tiền là 430.75, mức phí 1.50.
* Tính tiền lãi cho acc3.
* Xuất thông tin của acc1, acc2, acc3 (kiểm tra lại kết quả).
* Chuyển tiền từ acc2 sang cho acc1 số tiền là 100.00.
* Xuất thông tin của acc1, acc2 (kiểm tra lại kết quả).

## Cài đặt các lớp có quan hệ Aggregation

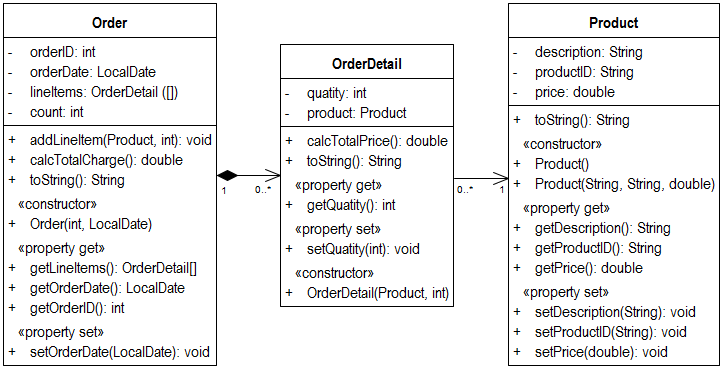
Cài đặt cho mô hình lớp sau:



Yêu cầu kết xuất: Giả sử nhập vào hình tròn có tâm O với 2 tọa độ x,y là 5,5, có bán kính là 10.5 thì xuất kết quả tương ứng là: “*Hình tròn có tâm* ***O(5,5)*** *với bán kính* ***10.5*** *có diện tích và chu vi lần lượt là* ***346.185*** *và* ***65.940****.*”

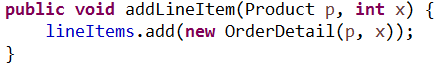
## Cài đặt các lớp có quan hệ Composition

Hiện thực mô hình lớp sau bằng ngôn ngữ lập trình Java.

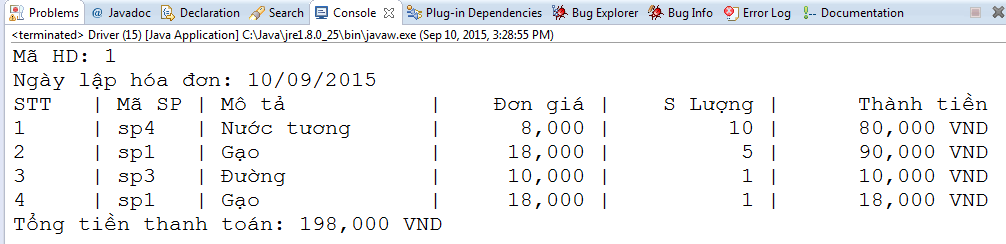


*Trong đó:*

* Tính thành tiền *(calcTotalPrice())* = quantity \* price.
* Tính tổng tiền hóa đơn *(calcTotalCharge())* = thành tiền.
* Phương thức *addLineItem(Product p, int q): void*, để thêm một sản phẩm p với số lượng q vào hóa đơn.



* Phần viết lớp cho phần kiểm nghiệm, in ra màn hình theo mẫu:



## 

Xây dựng lớp HocVien để quản lý họ tên, năm sinh, điểm 5 môn học của các học viên trong lớp học. Cho biết bao nhiêu người trong lớp được làm luận văn tốt nghiệp, bao nhiêu người thi tốt nghiệp, bao nhiêu người phải thi lại là tên các môn thi lại.

Tiêu chuẩn để xét như sau: *làm luận văn* phải có điểm trung bình lớn hơn 7 trong đó không có môn nào dưới 5; *thi tốt nghiệp* khi điểm trung bình không lớn hơn 7 và điểm các môn không dưới 5; *thi lại* khi có môn dưới 5 điểm.

## 

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.

# KẾ THỪA – ĐA HÌNH

*Mục tiêu:*

* *Hiểu và áp dụng được cách viết code kế thừa và đa hình trên Java.*
* *Hiểu và áp dụng được cách sử dụng mô hình lớp với mô tả kế thừa.*

*Yêu cầu:*

* *Tạo project tên Module03.*
* *Mỗi bài làm trong một package có tên package là bai01, bai02,…*
* *Viết documentation comment cho class.*

## 

Công ty du lịch X có quản lý thông tin các chuyến xe. Có 2 loại chuyến xe:

* Chuyến xe nội thành: Mã số chuyến, Họ tên tài xế, số xe, số tuyến, số km đi được, doanh thu.
* Chuyến xe ngoại thành: Mã số chuyến, Họ tên tài xế, số xe, nơi đến, số ngày đi được, doanh thu.

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 2 chuyến xe vào danh sách (không cần cho nhập từ bàn phím). Xuất tổng doanh thu cho tất cả các chuyến xe và tổng doanh thu của từng loại chuyến xe.

## 

Thư viện X quản lý danh sách các loại sách. Thông tin về các loại sách gồm:

* Sách giáo khoa: Mã sách, ngày nhập, đơn giá, số lượng, nhà xuất bản, tình trạng (“mới”, “cũ”).

Nếu tình trạng sách là “mới” thì: **thành tiền = số lượng \* đơn giá.**

Nếu tình trạng sách là “cũ” thì**: thành tiền = số lượng \* đơn giá \* 50%.**

* Sách tham khảo: Mã sách, ngày nhập, đơn giá, số lượng, nhà xuất bản, thuế. **Thành tiền = số lượng \* đơn giá + thuế.**

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 cuốn sách vào danh sách (không cần cho nhập từ bàn phím), sau đó:
   * Tính tổng thành tiền cho từng loại.
   * Tính trung bình cộng đơn giá của các sách tham khảo.
   * Xuất ra các sách giáo khoa của nhà xuất bản K (yêu cầu nhập K).

## 

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.

## 

Viết chương trình quản lý danh sách các giao dịch nhà đất. Thông tin bao gồm:

* Giao dịch đất: Mã giao dịch, ngày giao dịch, đơn giá, loại đất (“A”, “B”, “C”), diện tích.
* Nếu là loại B, C thì: thành tiền = diện tích \* đơn giá.
* Nếu là loại A thì: thành tiền = diện tích \* đơn giá \* 1.5.
* Giao dịch nhà: Mã giao dịch, ngày giao dịch, đơn giá, loại nhà (“cao cấp”, “thường”), địa chỉ, diện tích.
* Nếu là loại nhà cao cấp thì: thành tiền = diện tích \* đơn giá.
* Nếu là loại thường thì: thành tiền = diện tích \* đơn giá \* 90%.

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 đất.
   * Xuất ra các giao dịch của tháng 9 năm 2013.

## 

Viết chương trình quản lý danh sách hoá đơn tiền điện của khách hàng. Thông tin bao gồm các loại khách hàng:

* Khách hàng Việt Nam: mã khách hàng, họ tên, ngày lập hoá đơn, đối tượng khách hàng (“sinh hoạt”, “kinh doanh”, “sản xuất”), số KW tiêu thụ, đơn giá, định mức. Thành tiền được tính như sau:
* Nếu số KW tiêu thụ <= định mức thì thành tiền = số KW tiêu thụ\*đơn giá.
* Ngược lại, thành tiền = số KW tiêu thụ\*đơn giá\*định mức + số KW vượt định mức\*đơn giá\*2.5.
* Khách hàng nước ngoài: mã khách hàng, họ tên, ngày lập hoá đơn, quốc tịch, số KW tiêu thụ, đơn giá. Thành tiền được tính = số KW tiêu thụ\*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 khách hàng vào danh sách (không cần nhập từ bàn phím), sau đó:
   * Tính tổng số lượng cho từng loại khách hàng.
   * Tính trung bình thành tiền của khách hàng người nước ngoài.
   * Xuất ra các hoá đơn trong tháng 09 năm 2018 (cùa cả 2 loại khách hàng).

## 

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.
4. 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).

## 

Tạo lớp trừu tượng **Shape** với 3 phương thức trừu tượng **draw()**, **erase( )** và **move(int** *x***, int** *y***)**. Tạo các lớp con như liệt kê ở bảng dưới đây đồng thời override các phương thức trừu tượng (các phương thức này chỉ in câu thông báo tương ứng ra console).

|  |  |  |
| --- | --- | --- |
| **Class** | **Superclass** | **Subclass** |
| Shape | - | Circle, Quad, Triangle, Polygon |
| Circle | Shape | - |
| Quad | Shape | Rectangle |
| Rectangle | Quad | - |
| Triangle | Shape | - |
| Polygon | Shape | - |

Viết lớp **Drawing** có phương thức **drawShape**(Shape*theShape***)**, phương thức có tham số là đối tượng Shape. Trong phương thức gọi tới **draw()** của từng đối tượngShape. Thực thi phương thức này.

## 

Với một tập mini các loại xe trong thế giới thực cho bên dưới:

****

Yêu cầu quản lý:

* Thông tin từng loại xe.
* Tính tiền thuế cho từng loại xe dựa trên giá trị xe như sau:
  + **Xe đạp:** Không đóng thuế.
  + **Xe máy**: gồm VAT=10% và thuế trước bạ 5%.
  + **Xe ô tô khách**: gồm thuế tiêu thụ đặc biệt 30% (nếu số chỗ >=5), 50% (nếu số chỗ <5); VAT=10% và thuế trước bạ 20%.
  + **Xe ô tô tải:** gồm VAT=10% và thuế trước bạ 2%.

Yêu cầu sinh viên:

1. Thiết kế lược đồ lớp theo mô tả trên.
2. Cài đặt cho lược đồ đã thiết kế.

## 

Trường đại học X có nhiều cấp đào tạo, thông tin về sinh viên được tổ chức như sau:

* Sinh viên trung cấp (trung cấp 2 năm)
* Sinh viên cao đẳng
* Sinh viên đại học
* Sinh viên liên kết (Austraulia, US)

Yêu cầu

1. Xác định các thông tin, dữ liệu chung, riêng.
2. Định nghĩa các lớp (các thuộc tính, và phương thức) và mô hình phân cấp các lớp.
3. Viết lớp cho phép nhập và hiển thị thông tin về sinh viên.