I. Bài tập thực hành

Yêu cầu: Tất cả các bài phải được comment đầy đủ theo phong cách sau: https://www.howkteam.vn/course/lap-trinh-java-co-ban-java-core/cach-comment-trong-java-1217

Tuần 1. Cài đặt

1. Cài đặt

- Download JDK và cài đặt
- Cài đặt biến môi trường PATH trỏ đến thư mục bin của phiên bản jdk vừa tạo
- Mở màn hình dòng lệnh (command line)
- Gõ "java –version": Các thông tin gì được hiển thị trên màn hình?
- Gõ "javac –version": Các thông tin gì được hiển thị trên màn hình?

2. Biên dịch và chạy thử chương trình

- Tạo một thư mục, tạm lấy tên là "OOP"
- Trong thư mục OOP tạo file "HelloWorld.java" in ra dòng chữ "Hello World, Java!"
- Mở command line và chuyển đến thư mục trên
- Biên dịch với lệnh "javac HelloWorld.java"
- Sau khi biên dịch xong, kiểm tra lại thư mục "OOP", trong thư mục sẽ có thêm file "HelloWorld.class"
- Chạy chương trình với lệnh "java HelloWorld". Nếu chương trình được chạy đúng dòng "Hello World, Java!" sẽ được in ra màn hình.

Tuần 2. Setter/Getter, Constructor

1. Tạo lớp Student với các thuộc tính (String) gồm: name (tên sinh viên), id (mã số sinh viên), group (lớp học), emai (địa chỉ email)
2. Tạo thêm lớp StudentManagement với phương thức main(). Trong phương thức main(), tạo một số đối tượng thuộc lớp Student (dùng từ khóa "new")
3. Khai báo các thuộc tính trong Student là "private"
4. Thêm các phương thức get/set cho các thuộc tính (gọi là getter/setter). Ví dụ, với thuộc tính "name", hai phương thức cần thêm gồm "public String getName()" và "public void setName(String n)".
5. Thêm phương thức "String getInfo()" cho lớp Student. Phương thức này in ra màn hình tên, mã số SV, lớp, và email của sinh viên.
6. Thay đổi phương thức main() trong lớp StudentManagement để

    a. Tạo ra một sinh viên
    b. Thiết lập các thông tin về sinh viên (là thông tin của bạn)
    c. In ra tên của sinh viên
    d. In ra toàn bộ thông tin của sinh viên (nghĩa là gọi phương thức getInfo())
7. Thêm 3 phương thức khởi tạo cho lớp Student
    a. Phương thức khởi tạo không có tham số: Student(). Nếu khởi tạo bằng phương thức này, sinh viên được tạo ra sẽ có giá trị cho các thuộc tính như sau: name = "Student", id="000", group="K59CB", email="uet@vnu.edu.vn"
    b. Phương thức khởi tạo có tham số Student(String n, String sid, String em). Khởi tạo bằng phương thức này sẽ có sinh viên với các thuộc tính "name", "id", và "email" là các giá trị từ tham số, còn "group" có giá trị là "K59CB".
    c. Phương thức khởi tạo sao chép Student(Student s). Với phương thức này, đối tượng tạo ra sẽ có các thuộc tính với trị giống như của đối tượng s.
8. Thay đổi phương thức main() của StudentManager để kiểm tra 3 loại phương thức khởi tạo trên
9. Trong lớp StudentMangement, viết một phương thức "public boolean sameGroup(Student s1, Student s2)" để kiểm tra xem hai sinh viên s1 và s2 có cùng lớp hay không
10. Thay đổi phương thức main() để tạo ra 3 sinh viên với 2 sinh viên thuộc lớp "K59CLC", 1 sinh viên thuộc "K59CB" và kiểm tra phương thức sameGroup() ở trên
11. Sửa lại lớp StudentManagement để lớp này có một thuộc tính students là array chứa các đối tượng thuộc lớp Student (max. 100)
12. Viết phương thức "studentsByGroup()" cho lớp StudentManagement để in ra danh sách sinh viên theo từng lớp
13. Viết phương thức "removeStudent(String id)" cho lớp StudentManagement để xóa sinh viên với mã số là id ra khỏi danh sách

Tuần 3. Kiểu dữ liệu nguyên thủy, equals

Câu 1. Viết các hàm sau và kiểm tra kết quả trong phương thức main

a. Viết hàm tìm ước số chung lớn nhất của 2 số nguyên a và b. In kết quả ra màn hình.

b. Viết hàm tính Fibonaxi của một số nguyên n, công thức như sau:

$$F_n := F(n) := \begin{cases} 0, & \text{khi } n = 0; \\ 1, & \text{khi } n = 1; \\ F(n-1) + F(n-2) & \text{khi } n > 1. \end{cases}$$

In kết quả ra màn hình.

Câu 2. Tạo lớp phân số PS có hai thuộc tính là tử và mẫu

a. Viết hàm khởi tạo có tham số cho lớp PS có sử dụng từ khóa this

b. Xây dựng các phương thức cộng, trừ, nhân, chia phân số

c. Viết phương thức "public boolean equals(Object obj)" so sánh hai phân số

Câu 3. Hãy mô tả ít nhất mười đối tượng quanh cuộc sống của bạn (ví dụ: giáo viên, con mèo, tổng thống, lập trình viên, v.v.). Với mỗi đối tượng, bạn hãy mô tả trong một class tương ứng. Chú ý rằng các thuộc tính và phương thức phải thể hiện được đặc trưng của đối tượng đó.

Yêu cầu:

- Tất cả mọi chương trình phải có đủ comment cho từng class, từng hàm
- Các thuộc tính cần có đủ setter, getter tương ứng

Tuần 4. Static, mảng, JUnit

Viết các hàm static sau, sử dụng JUnit viết từng hàm ít nhất 5 bộ test để kiểm tra tính đúng đắn (sử dụng hàm assertEquals).

a. Tìm giá trị lớn nhất của hai số nguyên, giá trị trả về của hàm là số lớn nhất

b. Tìm giá trị nhỏ nhất của của một mảng số nguyên (kích thước mảng <= 100 phần tử)

d. Viết chương trình tính chỉ số BMI theo công thức sau:

BMI = Cân nặng (kg) / (Chiều cao(m) ^ 2)

In ra kết quả đánh giá chỉ số BMI dựa theo công thức trên:

- Nếu BMI dưới 18.5 thì hiển thị “Thiếu cân”

- Nếu BMI từ 18.5 đến 24.99 thì hiển thị “Bình thường”

- Nếu BMI từ 23 đến 24.99 thì hiển thị “Thừa cân“

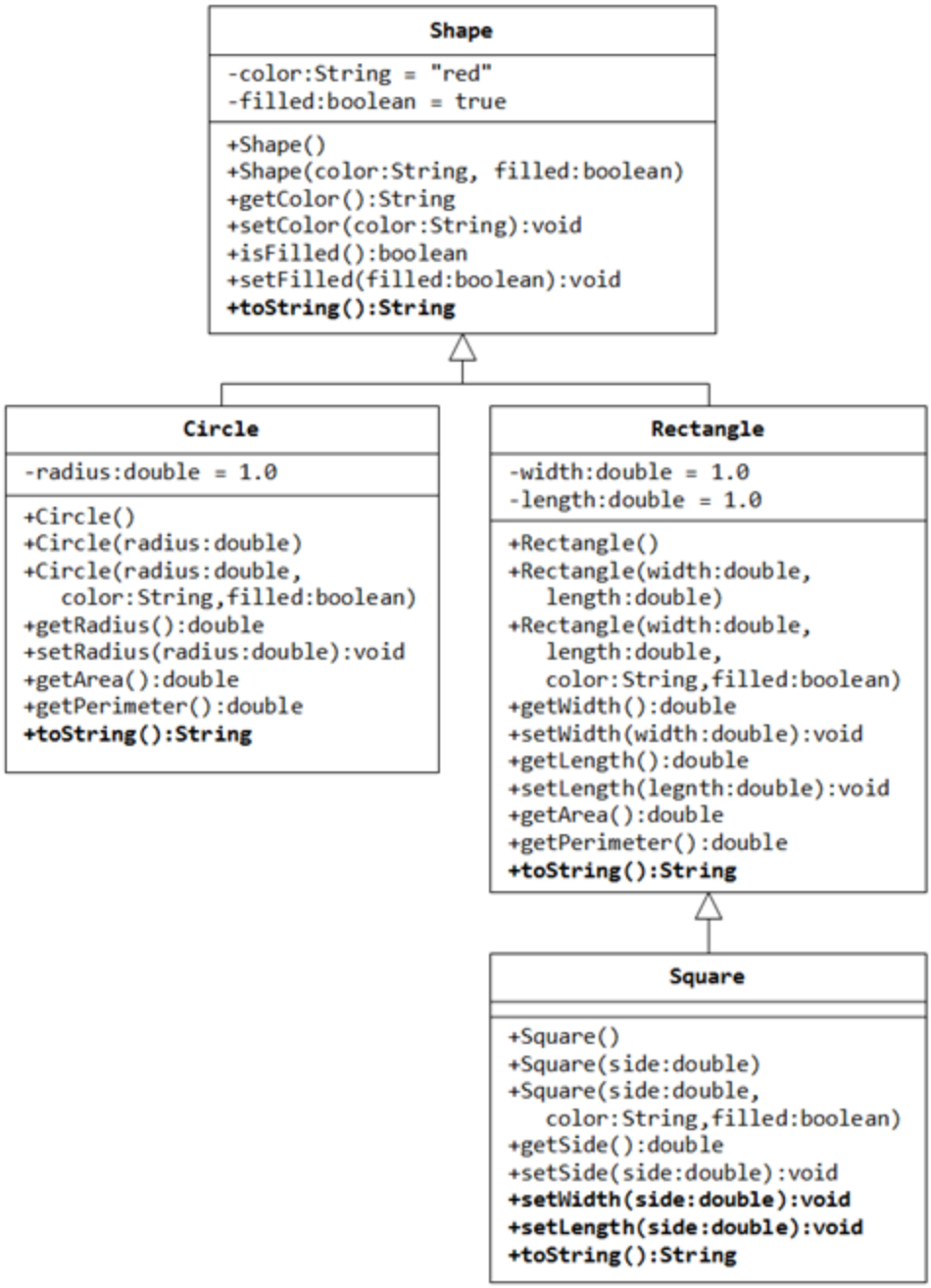
- Nếu BMI > 25 thì hiển thị “Béo phì”

Tuần 5. Kế thừa, final

Câu 1. Cho các đối tượng sau gồm Hoa Quả, Quả Cam, Quả Táo, Cam Cao Phong, Cam Sành.

- Dựa trên cách hiểu của mình về thừa kế, hãy viết chương trình mô tả quan hệ các đối tượng trên.
- Với từng đối tượng, hãy bổ sung các thuộc tính và phương thức bạn cho là cần thiết (VD: giá bán/cân, nguồn gốc xuất xứ, ngày nhập, số lượng, v.v.)
- Viết hàm main khởi tạo các đối tượng trên

Câu 2. Cho biểu đồ lớp như sau:



- Trong biểu đồ trên có tất cả bao nhiêu quan hệ thừa kế (is-a)? Tại sao lớp Circle có thể thừa kế lớp Shape mà không phải lớp Rectangle?
- Hiện thực các lớp trong sơ đồ trên. Viết hàm main để kiểm tra chương trình.
- Định nghĩa thêm số PI trong lớp Circle ở chương trình vừa tạo; sau đó sử dụng giá trị PI để tính chu vi và diện tích hình tròn. Biến PI này có nên để final không?

Yêu cầu:

- Tất cả mọi chương trình phải có đủ comment cho từng class, từng hàm
- Các thuộc tính cần có đủ setter, getter tương ứng

Tuần 6. (1) Đa hình, instanceof, abstract

Giả sử bạn cần viết một ứng dụng đồ họa với những thông tin thiết kế ban đầu như sau

- Diagram: là lớp đại diện cho sơ đồ đang được vẽ
- Layer: một đối tượng thuộc lớp Diagram có một hoặc nhiều đối tượng thuộc lớp Layer
- Shape: là lớp đại diện cho các hình vẽ khác nhau (Rectangle, Square, Triangle, Circle). Mỗi đối tượng Layer chứa nhiều đối tượng lớp Shape
- Các hình vẽ có thuộc tính để xác định vị trí và kích thước
- Các hình vẽ có thể được tô màu và có thể được di chuyển

Hãy:

- Định nghĩa các lớp trên (Diagram, Layer, Shape, Rectangle, Square, Triangle, Circle, và các lớp khác nếu cần thiết)
- Bổ sung phương thức cho lớp Layer để xóa tất cả các đối tượng thuộc lớp Triangle trong lớp
- Bổ sung phương thức cho lớp Diagram để xóa tất cả các đối tượng thuộc lớp Circle trong Diagram
- Viết phương thức main để kiểm thử các phương thức trên

Tuần 7. (2) Đa hình

Tiếp tục từ bài buổi trước, hãy:

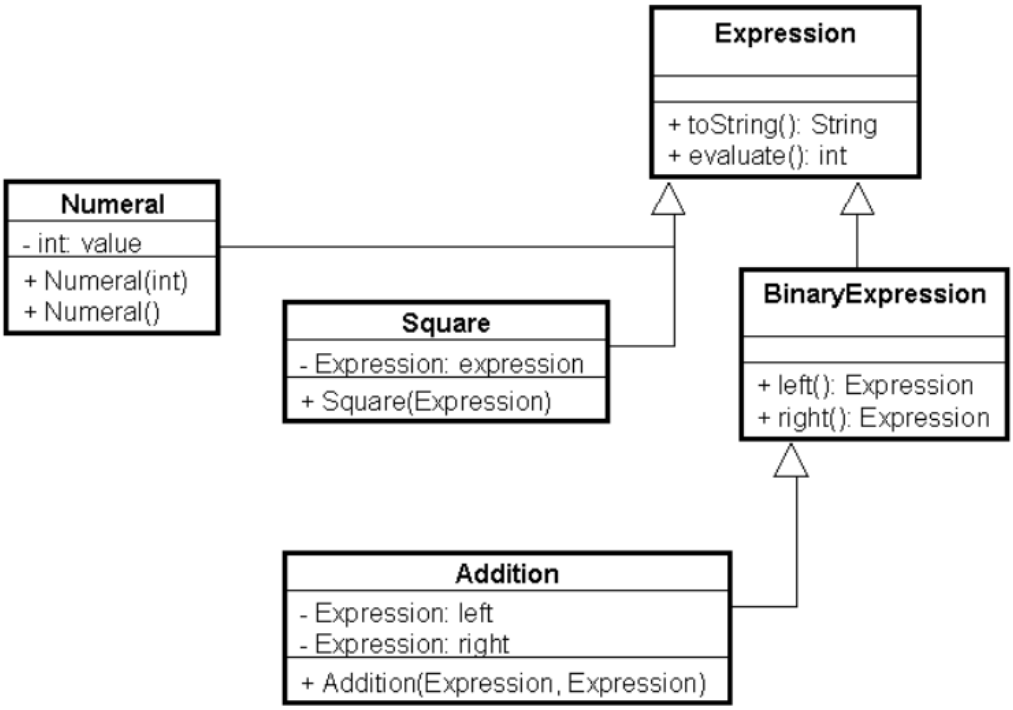
- Thêm thuộc tính visible cho Layer. Khi thuộc tính này là false, các hình thuộc đối tượng Layer đấy sẽ không được vẽ trên Diagram
- Viết phương thức xóa các hình trùng nhau trong một Layer (ví dụ: với Circle, 2 hình trùng nhau sẽ tọa độ tâm và độ lớn bán kính như nhau)
- Viết phương thức để chuyển từng loại hình vẽ vào từng đối tượng Layer. Sau khi chạy phương thức này, tất cả các Square của Diagram sẽ được chuyển vào 1 Layer, các Circle được chuyển vào 1 Layer khác,… Nếu số loại hình nhiều hơn số Layer hiện có thì tạo thêm.
- Ngoài những hình đã có (Rectangle, Square, Triangle, Circle), bổ sung thêm hình lục giác đều Hexagon

Tổng hợp lại tất cả các lab bài Diagram vào 1 project. Yêu cầu:

- Tất cả mọi chương trình phải có đủ comment cho từng class, từng hàm
- Các thuộc tính cần có đủ setter, getter tương ứng

Tuần 8. Ngoại lệ (try-catch, throw)

Câu 1. Cài các class như trong hình dưới đây.



- Đặt chế độ abstract cho các class và method thích hợp
- Viết class ExpressionTest để thử, trong đó tạo biểu thức (10^2 - 1 + 2*3)^2 và tính kết quả, in ra màn hình. Chú ý: không cần nhập dữ liệu từ tệp ngoài hoặc xử lý xâu kí tự chứa biểu thức.
- Bổ sung lớp Subtraction (trừ), Multiplication (nhân), và Division (chia). Sau đó, sửa lớp ExpressionTest để chạy thử.
- Trong hàm main lớp ExpressionTest, viết mã nguồn in thông báo lỗi "Lỗi chia cho 0" trên console khi phép chia cho 0 xảy ra (gợi ý: sử dụng ngoại lệ java.lang.ArithmeticException).

Câu 2. Sử dụng từ khóa throws để tạo các hàm gây ra các trường hợp ngoại lệ sau. Với từng kiểu ngoại lệ, hãy viết hàm main() tương ứng cài đặt trường hợp xảy ra ngoại lệ.

- java.lang.NullPointerException
- java.lang.ArrayIndexOfBoundException
- java.lang.ArithmeticException
- java.lang.ClassCastException
- java.io.IOException
- java.io.FileNotFoundException

Tuần 9. I/O

Xây dựng lớp tiện ích Utils

a. Viết hàm static để đọc một tệp .txt từ ổ cứng:

public static String readContentFromFile(String path),

trong đó biến path là đường dẫn đến tệp cần đọc.

b. Viết hàm static để xuất nội dung một xâu vào ổ cứng:

public static void writeContentToFile(String path),

trong đó biến path là đường dẫn đến tệp cần ghi nội dung. Nếu tệp đã có nội dung thì ta xóa nội dung đó trước khi ghi nội dung mới.

c. Tương tự ý câu b nhưng thay vì xóa nội dung cũ đi, ta bổ sung nội dung mới vào cuối tệp hiện tại

d. Viết hàm static để tìm kiếm một tệp trong một thư mục:

public static File findFileByName(String folderPath, String fileName),

trong đó folderPath là tên thư mục, fileName là tên tệp cần tìm kiếm.

Tuần 10. Data structures: List, Array, String

Câu 1: Dựa trên bài thực hành tuần trước, hãy phân tích tệp Utils.java để

- Áp dụng các phép biến đổi xâu, hãy viết hàm "List<String> getAllFunctions(File path)" để lấy toàn bộ hàm static trong tệp đó qua các phép xử lý xâu, trong đó path là đường dẫn đến tệp Utils.java. Mỗi một phần tử String trong đối tượng trả về lưu mã nguồn hàm static tìm được.
- Viết hàm "public String findFunctionByName(String name)" để tìm kiếm một hàm trong tệp. Trong đó, name là tên của hàm cùng với kiểu biến. Ví dụ, hàm "public static File findFileByName(String folderPath, String fileName)" có tên là "findFileByName(String,String)".
  Chú ý trường hợp hàm đặt trong comment.
- Viết hàm main để kiểm tra.

Câu 2: Sử dụng thuật toán sắp xếp nổi bọt, hãy viết chương trình sắp xếp tăng dần mảng 1000 phần tử số thực. Dữ liệu mảng cần sắp xếp được sinh ngẫu nhiên. Viết hàm main để kiểm tra.

Tuần 11. Template

Câu 1. Áp dụng lập trình tổng quát, hãy viết chương trình sắp xếp cho các kiểu dữ liệu nguyên thủy trong java. Viết hàm main để kiểm tra.

Câu 2. Áp dụng lập trình tổng quát, hãy viết chương trình tìm số lớn nhất của một mảng các kiểu dữ liệu nguyên thủy trong java (dùng ArrayList). Viết hàm main để kiểm tra.

Tuần 12. Design pattern: Composite, Strategy

Câu 1. Hãy giúp Bob tìm mẫu thiết kế thích hợp với cấu trúc dữ liệu cho bài toán phả hệ như sau:

- Từng cá nhân cần lưu các thông tin gồm: ngày tháng năm sinh, giới tính, v.v.
- Một cá nhân có thể có một nhiều con hoặc không có con

- Một cá nhân có thể kết hôn hoặc không

Ví dụ: James kết hôn với Hana sinh ra hai người con là Ryan và Kai. Ryan không lấy vợ. Kai lấy Jennifer sinh ra bốn người con gồm hai nam, hai nữ. Bốn người con này tiếp tục kết hôn và sinh con đẻ cái.

Dựa trên mẫu thiết kế vừa tạo, hãy:

- Tìm tất cả các cá nhân không kết hôn trong danh sách phả hệ
- Tìm tất cả các cặp vợ chồng có hai con trong danh sách phả hệ
- Tìm tất cả các thế hệ mới nhất trong danh sách phả hệ
- Viết hàm main kiểm tra

Gợi ý: Sử dụng mẫu thiết kế Composite vì mẫu này thích hợp để lưu danh sách dạng cây

Câu 2. Cho một mảng các số nguyên có thể được sắp xếp tăng dần, hoặc giảm dần sử dụng thuật toán sắp xếp nổi bọt, hoặc sắp xếp chọn. Sau này, James mong muốn áp dụng thêm các thuật toán sắp xếp khác (hiện tại James chưa muốn cài đặt). Hãy giúp James xây dựng mẫu thiết kế thích hợp nhất. Viết hàm main để chạy thử chương trình.

Gợi ý: Sử dụng mẫu thiết kế Strategy