Домашнее задание 2

Задача 1:

Реализовать свой класс Vector.
Создайте класс, описывающий вектор в трехмерном пространстве (координаты - три поля double).

У него должны быть:

- конструктор с параметрами в виде списка координат *x, y, z*
- метод, вычисляющий длину вектора. Корень можно посчитать с помощью Math.sqrt():

$$\sqrt{x^2 + y^2 + z^2}$$

- метод, вычисляющий скалярное произведение:

$$x_1 x_2 + y_1 y_2 + z_1 z_2$$

- метод, вычисляющий векторное произведение с другим вектором:

$$(y_1 z_2 - z_1 y_2, z_1 x_2 - x_1 z_2, x_1 y_2 - y_1 x_2)$$

- метод, вычисляющий угол между векторами (или косинус угла): косинус угла между векторами равен скалярному произведению векторов, деленному на произведение модулей (длин) векторов:

$$\frac{(a, b)}{|a| \cdot |b|}$$

- методы для суммы и разности:

$$(x_1 + x_2, y_1 + y_2, z_1 + z_2)$$

$$(x_1 - x_2, y_1 - y_2, z_1 - z_2)$$

В классе вектор нужно переопределить hashCode и equals (hashCode & equals, можно средствами IDEA, см. разбалловку).

Также должен быть отдельный класс с методом main, в котором будут инициализированы два вектора. Ввод координат осуществлять с клавиатуры (https://javarush.ru/groups/posts/klass-scanner). Затем нужно вызвать для пары векторов все перечисленные методы и вывести их значения на экран.

Всего 3 балла за задачу:

- реализован класс для вектора - 1,5 балла
- переопределены методы hashCode и equals - 0,2 балла, если средствами IDEA; 0,5 балла, если своими силами без использования класс-обертки Double (придется познакомиться с приведением типов до 3 занятия самостоятельно)
- реализован метод main для ввода данных, в котором инициализировано 2 объекта и вызваны все функции - 0,5 балла

Задача 2:

Реализовать свою иерархию классов для следующих видов компьютеров:

- персональный компьютер (ПК, он же desktop)
- ноутбук
- нетбук
- моноблок
- сервер
- неттоп

Для выполнения этой задачи вам понадобится композиция, о которой я вам забыл рассказать на занятии. Изучить ее самостоятельно будет несложно:

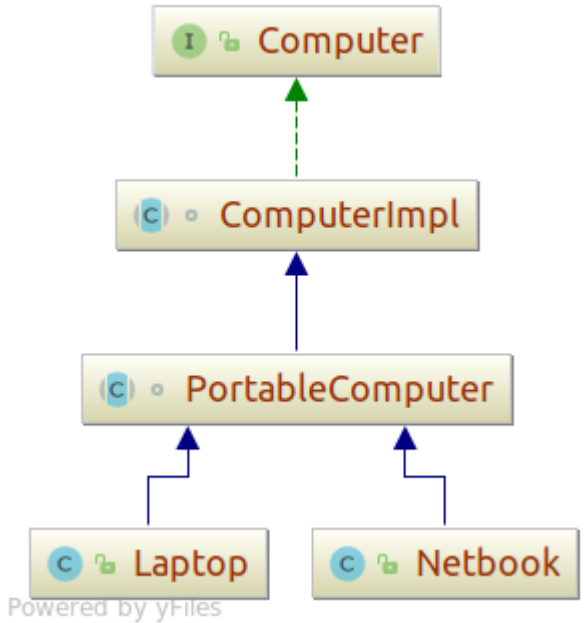
Композиция и агрегирование

Просто про композицию

Если после прочтения не понятно - ничего страшного, можно сделать и без нее.

Иерархия классов составляется на ваше усмотрение, вот несколько советов для выполнения задачи:

- все виды компьютеров должны реализовывать один интерфейс
- планшет, ноутбук, нетбук, являются портативными компьютерами
- ПК, неттоп и моноблок являются настольными компьютерами
- серверный компьютер не имеет ничего общего с остальными
- у моноблока, нетбука, ноутбука есть дисплей
- у ноутбука и нетбука есть клавиатура
- "являются" по сути равно "наследуют"
- "есть" по сути равно "композиция"
- Вот подсказка, как будет выглядеть диаграмма классов для ноутбука и нетбука:



То есть есть интерфейс Computer, в котором прописаны базовые методы. Далее его реализует абстрактный класс ComputerImpl, от которого мы наследуем абстрактный класс PortableComputer, от которого уже наследуем обычные классы ноутбука и нетбука. Ваша задача - аналогичным образом реализовать иерархию для других классов.

Общий интерфейс для всех компьютеров должен описывать следующие методы:

- включить
- выключить
- подключиться к интернету

При включении/выключении должна выводится следующая информация: "Компьютер такой-то, с такими характеристиками, включение/выключение" При подключении к интернету: "Компьютер такой-то, с такими характеристиками, подключился к интернету"

Подводя итог, у вас будут следующие классы:

- интерфейс "компьютер"
- клавиатура
- дисплей
- персональный компьютер (ПК, он же desktop)
- ноутбук
- нетбук
- моноблок
- сервер
- неттоп

Какие поля будут у каждого класса:

- ПК: процессор, ОЗУ, жесткий диск, видеокарта
- Ноутбук: процессор, ОЗУ, жесткий диск, монитор, клавиатура
- Нетбук: процессор, ОЗУ, жесткий диск, монитор, клавиатура

- Моноблок: процессор, ОЗУ, жесткий диск, монитор
- Неттоп: процессор, ОЗУ, жесткий диск
- Сервер: процессор, ОЗУ, жесткий диск

Для простоты процессор, ОЗУ, жесткий диск, видеокарта, крепление - строка (например, процессор "intel i5", жесткий диск "ssd 250gb" и т.п.). Монитор и клавиатура - отдельные классы, "вставляем" их с помощью композиции.

Также нужно реализовать main в отдельном классе, где создать по одному объекту каждого класса, добавить их в массив Computer[ ], пройтись циклом и вызвать для каждого поочередно: включить, подключиться к интернету, выключить.

Оценка:

- описаны все классы - 2 балла
- нет ошибок в иерархии - 0,5 балла
- есть композиция - 1 балл
- есть main с инициализацией минимум одного объекта каждого класса компьютера - 1 балл

Также предусмотрены дополнительные баллы:

- инициализация из файла (т.е. считывание из файла информации о всех компьютерах, вызов конструктора, тут вам поможет String.split()) - 0,5 балла
- реализовать классы (у всех есть поле производитель, поэтому можно создать абстрактный класс Component и наследовать от него): процессор (количество ядер, частота), ОЗУ (память, частота), жесткий диск (тип, объем), видеокарта (память), положить их в отдельный пакет и использовать их с помощью композиции в коде - 0,5 балла