

External java training

FIRST task Chapter A

Kick start

[www.training.by](http://www.training.by)

First Task KickStart Chapter A

Необходимо написать приложение и тесты к нему согласно требованиям, приведенным ниже. В приложении должна быть реализована функциональность, определенная индивидуальным заданием.

Требования

|  |
| --- |
| Пусть задание сформулировано следующим образом: «**Круг.** *Разработать классы Точка и Круг.* Создать методы и тесты: вычисления площади и периметра круга; является ли фигура кругом(радиус не может быть <=0); пересекает ли фигура только одну из осей координат на заданное расстояние.» |
|  |
| * Разработать entity-классы, например: «*Разработать классы Точка и Круг»* * **Entity-классы не следует наполнять** **методами**, выполняющими функциональные действия (методами бизнес-логики, такими как вычисление, поиск и т.д.). * Разработать классы логики реализующие заданные функциональности, например: «*Реализовать методы вычисления площади и периметра круга*» * **Параметры**, необходимые для создания объектов, организовать как чтение информации из файла (.txt). ***Часть данных должны некорректной.*** Если встретилась некорректная строка, приложение должно переходить к следующей строке. Все файлы должны находиться в отдельном каталоге. * Разработать классы для валидации исходных данных для создания объектов entity-классов.   Например: *корректная* строка в файле для создания объекта **Круг**: *«1.0 2.0 3.0»,* где первое число задает радиус круга, второе и третье – координаты центра;  *некорректная* строка в файле для создания объекта **Круг**: «*1.z0 2.0 3.0»* - недопустимый символ «z», всю строку следует считать некорректной здесь и в случаях приведенных ниже;  *некорректная* строка в файле для создания объекта **Круг**: *«1.0 2.0»* - недостаточно информации для создания объекта;  *некорректная* строка в файле для создания объекта **Круг**: *«-1.0 2.0 3.0»* - невозможно создать Круг с отрицательным радиусом.   * Для классов-сущностей следует **переопределять методы класса Object**: toString(), equals(), hashCode(). Методы класса Objects не использовать. |
|  |
| * Все **классы** приложения должны быть структурированы **по пакетам**. |
| * **Использовать собственные классы исключительных ситуаций**. |
| * **Оформление кода** должно соответствовать **Java Code Convention**. |
| * Для записи логов использовать Log4J2. * Код должен быть покрыт Unit-тестами. При написании тестов **запрещено**: писать логи и использовать операторы ветвления: if, for, while, do\while, switch; |
| * **Класс с методом main в задании должен отсутствовать**. Запуск только тестами. * ***Обратить внимание на примечания 1 и 2.*** |

Индивидуальные задания

|  |  |
| --- | --- |
| **1** | **Треугольник.** Разработать классы Точка и Треугольник. Создать методы и тесты: вычисления площади и периметра треугольника; составляют ли точки треугольник (не лежат ли точки на одной прямой); является ли треугольник прямоугольным, равнобедренным, равносторонним, остро/тупоугольным. |
| **2** | **Четырехугольник.** Разработать классы Точка и Четырехугольник. Создать методы и тесты: вычисления площади и периметра фигуры; составляют ли точки четырехугольник(не лежат ли три точки на одной прямой); является ли четырехугольник выпуклым; является ли четырехугольник квадратом, ромбом, трапецией. |
| **3** | **Овал.** Разработать классы Точка и Овал (задан двумя точками описанного прямоугольника). Создать методы и тесты: вычисления площади и периметра фигуры; составляют ли точки овал(не лежат ли две точки на одной прямой, параллельной осям координат); пересекает ли фигура только одну из осей координат на заданное расстояние; является ли фигура овалом, кругом. |
| **4** | **Шар.** Разработать классы Точка и Шар. Создать методы и тесты: вычисления площади поверхности шара, объема шара, соотношения объемов получаемых в результате рассечения шара координатной плоскостью; является ли объект шаром; касается ли шар любой из координатных плоскостей. |
| **5** | **Куб.** Разработать классы Точка и Куб. Создать методы и тесты: вычисления площади поверхности куба, объема куба; соотношения объемов получаемых в результате рассечения куба координатной плоскостью является ли объект кубом; находится ли основание куба на одной из координатных плоскостей. |
| **6** | **Тетраэдр.** Разработать классы Точка и Тетраэдр. Создать методы и тесты: вычисления площади поверхности фигуры и ее объема, а также соотношения объемов получаемых в результате рассечения фигуры координатной плоскостью; является ли объект заданной фигурой; находится ли основание фигуры на одной из координатных плоскостей. |
| **7** | **Пирамида.** Разработать классы Точка и Пирамида. Создать методы и тесты: вычисления площади поверхности фигуры и ее объема, а также соотношения объемов получаемых в результате рассечения фигуры координатной плоскостью; является ли объект заданной фигурой; находится ли основание фигуры на одной из координатных плоскостей. |
| **8** | **Конус.** Разработать классы Точка и Конус. Создать методы и тесты: вычисления площади поверхности фигуры и ее объема, а также соотношения объемов получаемых в результате рассечения фигуры координатной плоскостью; является ли объект заданной фигурой; находится ли основание фигуры на одной из координатных плоскостей. |

**Примечание 1:**

***Для задач 4-8. Cекущие плоскости, фигуры и основания следует ориентировать в пространстве параллельно осям и плоскостям координат, чтобы формулы вычисления сечений и параметров фигур не были слишком сложными.***

**Примечание 2:**

1. После if всегда следует положительный сценарий, после else - отрицательный
2. Если только if, то возможен и отрицательный сценарий
3. В if, for, while обязательно использовать { }
4. Если генерируется exception, не ловить его сразу же
5. В finally не генерировать исключения и не использовать return
6. Не генерировавать стандартные исключения. Разрешено только в методах private
7. fileWriter.close(); - в блок finally
8. Регулярные выражения в константы
9. В именах пакетов не использовать большие буквы
10. Файлы для тестов класть в test/resources
11. Использовать для файлов только относительные пути.
12. В классах логики использовать путь к файлу, а не getResource
13. Константы final statiс, имя должно быть в верхнем регистре слова отделять \_
14. Элементы перечисления именуются как константы
15. Не увлекаться статическими методами
16. Не объявлять get-теры и set-теры абстрактными, а также в интерфейсах
17. Не давать классам имена, совпадающие с именами стандартных классов и интерфейсов Java
18. Не разделять объявление переременной и присвоение ей значения в методах, то есть не писать:  
    Integer count;   
    count = 0;  
    а надо Integer count = 0;
19. Расстояние (в строчках кода) между использование переменной и ее объявление должно быть минимально!
20. В одной строчке – одна точка, то есть надо использовать локальные переменные, не надо:  
    sample.getSomething().getData()  
    надо:  
    Something something = sample.getSomething();  
    Data data = something.geData();
21. Не писать if (isValid == true), а писать if (isValid)
22. Не писать:  
     if (someValue == EXPECTED\_VALUE) {  
     return true;  
     } else {  
     return false;  
     }  
    писать:  
     return someValue == EXPECTED\_VALUE;
23. Использваоть assertEquals вместо assertTrue(some == other)
24. Использваоть assertTrue(isValid) вместо assertEquals(true, isValid)
25. Лучше тестовые объекты размещать в тестах в виде констант, а не создавать их в самом методе
26. Не использовать существующий FactoryMethod в тестах для создания объектов, объекты в тестах делвать через new
27. Тест должен иметь структуру: given, when, then, где given – преконддишены (инициализация данных), when – вызов тестируемого метода (всегда одна строчка), then – посткондишены (assert)
28. Перед сдачей проекта проверить соблюдение пунктов 1-27, иначе задание будут отклонено

| Ver. | Description of Change | Author | Date | Approved | |
| --- | --- | --- | --- | --- | --- |
| Name | Effective Date |
| <1.0> | Первая версия | Игорь Блинов | 15-09-2016 |  |  |
| <1.1> | update | Игорь Блинов | 14-02-2017 |  |  |
| <1.2> | update | Игорь Блинов | 09-06-2017 |  |  |
| <2.0> | Вторая версия | Игорь Блинов | 02-11-2017 |  |  |
| <2.1> | update | Игорь Блинов | 17-01-2018 |  |  |