**Задание на 1-ю лабораторную " Объектно-ориентированный анализ и моделирование, разработка статических моделей "** (я думаю под 4 часа),

состоящую из 2 задач (схем) "ООП, Проектирование и реализация иерархии сущностей на основе UML и Java" и

4-х этапов (а,б,в,г) для каждая из задач:

**а). Проектирование моделей в стандарте UML.**

**б). Программная реализация на JAVA в отдельном java-пакете.**

**в). Использование и полученного пакета для разработки программной системы.**

**г). Придумать для данных схем (моделей) описание (легенду) задачи реального мира. (это в конце, для обеих моделей).**

***Примечание*. Видим, сначала (на первых 2 часа лаб выполнить проектирование, а затем (на следующих 2 часах, когда я им прочитаю лекции) программирование. Ну, а если сделают за 2 часа все, то честь им и хвала! :)**

Задача 1:

**Этап а:**

а) Для схемы 1:

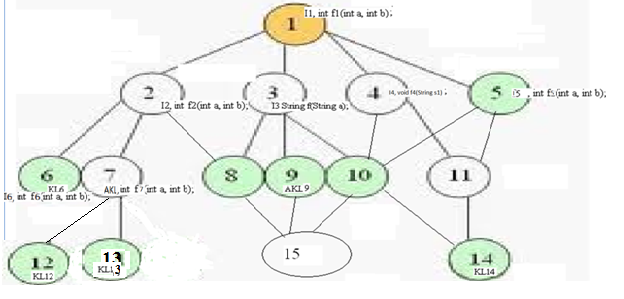
1.1. **Доопределить** схему (идентифицировать тип сущности при этом, предложить наиболее обоснованное возможное решение, по минимизации уровня абстракции каждого не определенного узла, сохраняя уже имеющиеся определенные ) и дописать до, как минимум, по одной собственной функции для каждой сущности.

**1.2. Для схемы Разработать Модель** классов в стандарте UML (Enterprise Architect), используя соглашения по именам Java[[1]](#endnote-2), соответствующую полученной схеме, которую Вы доопределили и дописать, как минимум по одной функции для каждой сущности.

**Этап б.**

**Реализовать** **модель** в программном коде JAVA):

СХЕМА 1, подробное описание:



, где I – это интерфейс; f() – это функция принадлежащая данной абстракции. Причем в каждой из абстракций (I,AKL,KL) своя дополнительная функция (f5, f6, f7, f8,…);

AKL - абстрактный класс (создать его как таковой, часть функций в нем должна быть реализована)? ;

KL настоящий (реальный) класс.

**2.1. Доопределить** листья дерева как таковые (их тип и предполагаемую функцию Вы должны сами, так, чтобы это было правильно с точки зрения формальной логики построения иерархии сущностей и фундаментальных свойств ООП (инкапсуляция, наследование и полиморфизм), и с максимальным уровнем реализации (минимизировать абстракции, где это можно в данной схеме, не меняя типа указанных уже абстракций.

Необходимо:

2.2.1. Разработать правильный проект этой структуры, проставить тип сущностей (интерфейс, абстрактный класс, класс), объявить или реализовать где это необходимо! простенькие методы, типа: сложение двух чисел, вывод чего-то и т.п.

Для этого:

* Проанализировать модель и расписать иерархическую структуру программных абстракций (расписать иерархию абстракций с методами (функциями), где какие можно поставить программные сущности (KL, AKL,I), исходя из логики схемы и правил определения производных сущностей (нижние элементы – все должны быть реальными классами, чтобы их можно было использовать для создания объектов), т.е. определить (снизу вверх) сначала KL, везде, где это допустимо, затем AKL везде, где это допустимо, затем I (интерфейсы), где не может быть чего-то менее абстрактного.
* Ввести методы, как минимум по одному в интерфейсах и других программных сущностях. Реализовать эти методы, в местах, где это возможно, но так, чтобы абстрактные классы в действительности соответствовали своему типу.

**2.2 Выполнить** программную реализацию (это второй этап по каждой задаче) в java-коде этой структуры. И далее считать, что Вы получаете такую иерархию «интерфейсов-абстрактных классов-классов» в качестве отдельного пакета, от субподрядчика (Аутсорсинговой компании) который можно использовать в решениях своих профессиональных задач.

**ЭТАП в.**

ИСПОЛЬЗОВАТЬ пакет в разработке системы.

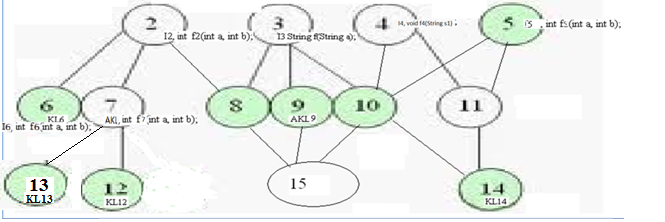
**3.1. В отдельном пакете** вне этой иерархии объявите класс My\_class, в нем реализуйте метод main() и необходимые объекты, чтобы продемонстрировать вызовы, любого из методов данной иерархии через имя объекта класса (правильно расставляйте ограничения доступа, учитывайте , что вызов осуществляется из другого пакета.

**3.2. Показать, также (написать в коде java),** как и какие методы можно вызвать через любую из интерфейсных ссылок (например I1 i2; I3 k2; …,и т.д.) и через любую из ссылок на абстрактный класс (например, AKL2 f2; …, и т.д.) (…это - приведение объекта к типу интерфейса).

**Задача 2.**

**То же, тоько структура, примерно такая:**

**СХЕМА 2**

****

**Здесь также необходимо доопределить иерархию (расставить тип и идентификаторы сущностей, определить методы и далее согласно формулировкам задания , приведенного выше описания выполнить пункты этапов 1,2,3.**

**Этап г.**

**Придумать для данных схем (моделей) описание (легенду) задачи реального мира, для которой бы подошло описание подобными структурами**

1. Согласно принятым в сообществе Java соглашениям, есть ряд правил которых желательно придерживаться при написании программы, если вы нарушите эти правила, компилятор не отметит имя как ошибочное и программа будет работать, но вашим коллегам будет трудно читать ваш код, да и вам самим возможно в будущем будет сложно понимать свой код который вы раньше написали, ну и вам соответственно будет сложно читать чужой код незная правил.  
   **Соглашения об именовании переменных, классов, методов, интерфейсов, пакетов, констант в Java.**

   |  |  |  |
   | --- | --- | --- |
   | **Тип** | **Правила именования** | **Примеры** |
   | Классы | **Имя класса начинается с большой буквы**, если в имени несколько слов, каждое слово пишется с заглавной буквы слитно. Имена классов должны быть существительными. Старайтесь, чтобы ваши имена классов выглядели просто и наглядно. Используйте целые слова, избегайте сокращений и аббревиатур. | class Raster; class ImageSprite; |
   | Интерфейсы | Интерфейсы именуются точно так же как и классы. | interface RasterDelegate;  interface Storing; |
   | Переменные | **Переменные начинаются со строчной первой буквы**, если в имени несколько слов, каждое следующее слово пишется с заглавной буквы слитно. Имена переменных не должны начинаться с подчеркивания "\_" или знака доллара "$". Имена переменных должны быть короткими, но со смыслом. Переменных состоящих из одного символа следует избегать, за исключением временных(одноразовых) переменных. Общие имена для временных переменных:  i, j, k, m Общие имена для числовых переменных:  n Общие имена для символьных переменных:  c, d, e | int i; char c; float myWidth; |
   | Методы | **Методы начинаются со строчной первой буквы**, если в имени несколько слов, каждое следующее слово пишется с заглавной буквы слитно. Методы должны быть глаголами. | run(); runFast(); getBackground(); |
   | Константы | **Константы должны состоять из заглавных символов**, если в имени несколько слов, каждое следующее слово отделяется от предыдущего символом подчеркивания "\_". | static final int MIN\_WIDTH = 4; static final int MAX\_WIDTH = 999; static final int GET\_THE\_CPU = 1; |
   | Пакеты | **Имя пакета пишется только строчными буквами**. Префикс уникального имени пакета должен быть одним из имен верхнего доменного уровня: ru, su, com, org, net, edu, gov и т.п. как указано в стандарте ISO 3166, 1981. Последующие компоненты имени пакета варьируются в зависимости от собственной внутренней организации домена. | com.sun.eng com.apple.quicktime.v2 edu.cmu.cs.bovik.cheese |

   Источник: [http://www.oracle.com/technetwork/java/javase/documentation](http://www.oracle.com/technetwork/java/javase/documentation/codeconventions-135099.html) [↑](#endnote-ref-2)