1. Что такое UML?

**UML** – унифицированный язык моделирования (Unified Modeling Language) – это система обозначений, которую можно применять для объектно-ориентированного анализа и проектирования. Его можно использовать для визуализации, спецификации, конструирования и документирования программных систем.

Проще говоря, если посмотреть картинки в поисковых системах, то станет понятно, что UML – это что-то про схемы, стрелочки и квадратики.  
  
Важно, что UML переводится как Unified Modeling Language. Главное здесь слово Unified. То есть наши картинки поймём не только мы, но и остальные, знающие UML. Получается, это такой международный язык рисования схем.

Плюсы и минусы UML проектирования

**Минусы:**

* трата времени;
* необходимость знания различных диаграмм и их нотаций.

**Плюсы:**

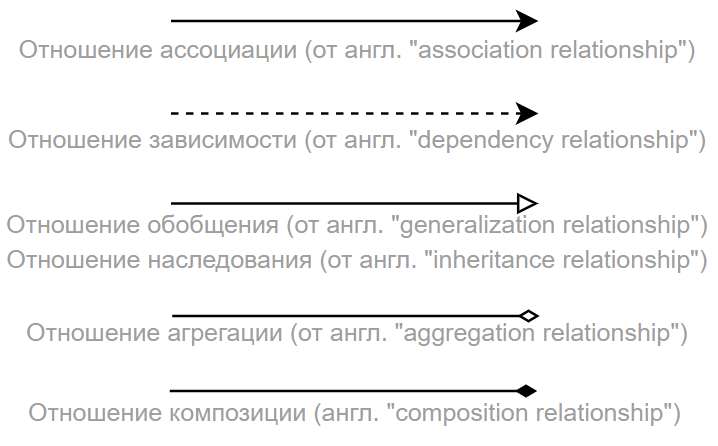
* возможность посмотреть на задачу с разных точек зрения;
* другим программистам легче понять суть задачи и способ ее реализации;
* диаграммы сравнительно просты для чтения после достаточно быстрого ознакомления с их синтаксисом.

2. Перечислите типы диаграмм UML.

Диаграмма последовательностей, диаграмма состояний, диаграмма классов, диаграмма деятельности, диаграмма компонентов, диаграмма вариантов использования…

3. Какие отношения между классами могут быть на диаграмме классов?

* Отношение ассоциации
* Отношение зависимости
* Отношение обобщения, также известное как отношение наследования.
* Отношение агрегации
* Отношение композиции



Отношение ассоциации используют, чтобы показать, что между классами (например, между двумя классами) существует некоторая связь. Обычно с помощью него на диаграмме классов показывают, что один класс пользуется функционалом другого класса.

Отношение зависимости используют, чтобы показать, что изменение одного класса требует изменение другого класса.

Итак, отношение наследования используется, чтобы показать, что один класс является родителем (базовым классом или суперклассом) для другого класса (потомка, производного класса).

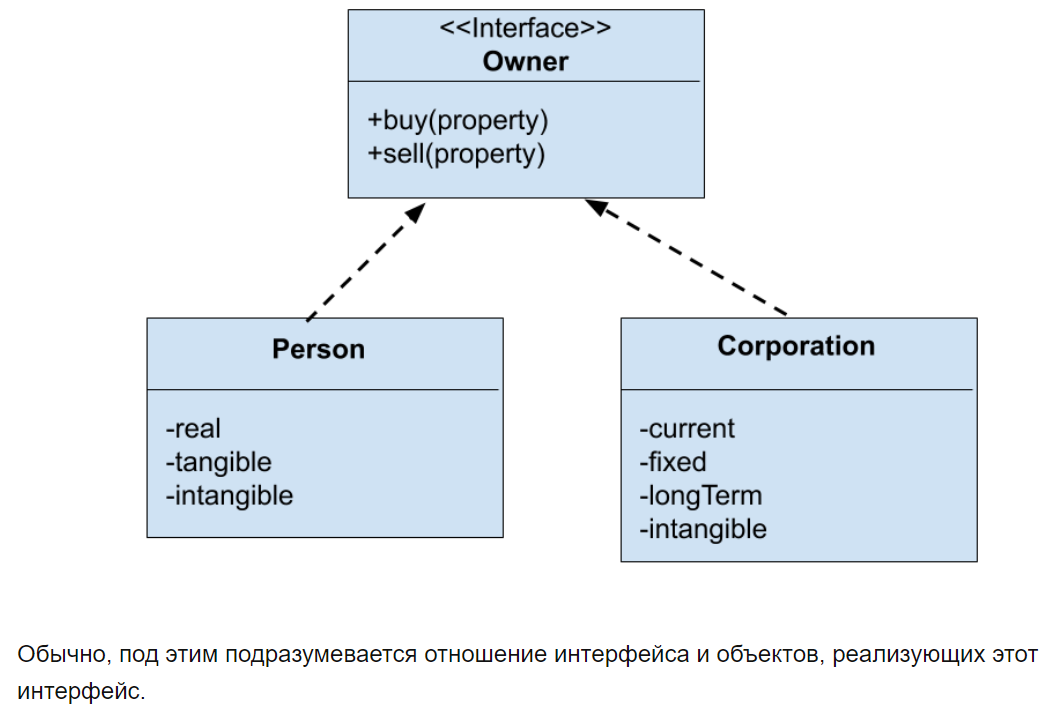
Отношение агрегации между двумя классами показывает, что один из них включает в себя другой класс в качестве составной части. При этом класс-часть **может и существовать обособленно** от класса-целого

Отношение композиции является частным случаем отношения агрегации. Однако у него есть одно отличие – классы-части, которые он соединяет с классом-целым, **не могут существовать обособленно.**

4. Как обозначаются абстрактные классы на диаграмме классов?

Абстрактный класс на диаграмме изображается так же, как и обычной класс, однако имя такого класса должно быть записано курсивом.

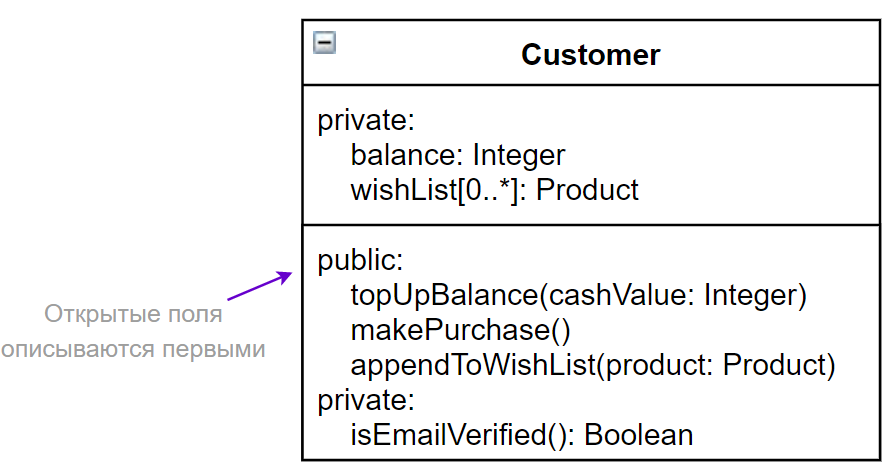
5. Как обозначаются интерфейсы на диаграмме классов?



6. Как отображается доступность членов класса на диаграмме классов?

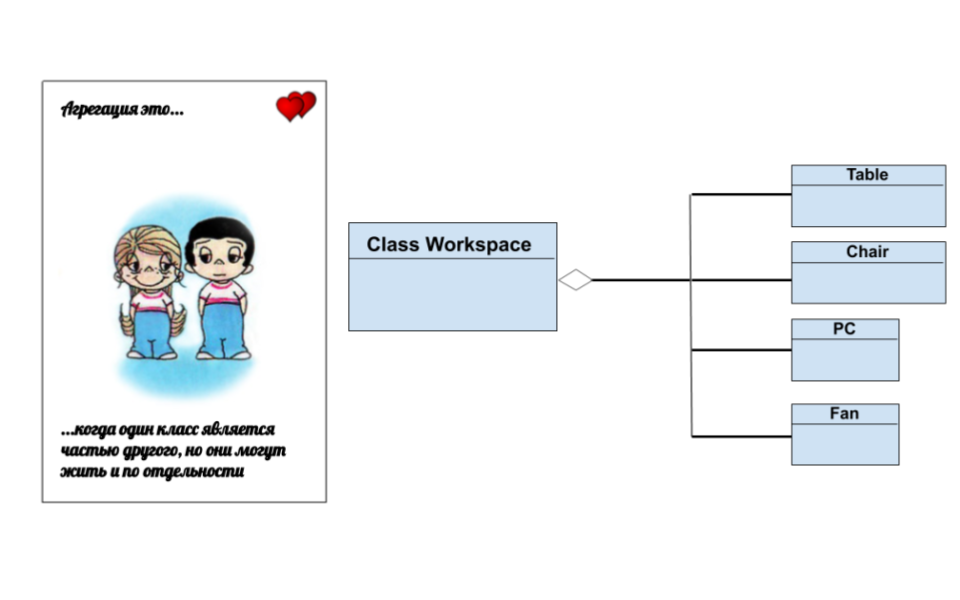
* "+" - открытое поле. Аналог public в языках программирования. Означает, что к полю можно обратиться из любой части программы.
* "-" - закрытое поле. Аналог private в языках программирования. Означает, что получить доступ к полю можно только внутри класса.
* "#" - защищённое поле. Аналог protected в языках программирования. Означает, что получить доступ к полю можно внутри класса и внутри производных классов.

Но можно и:



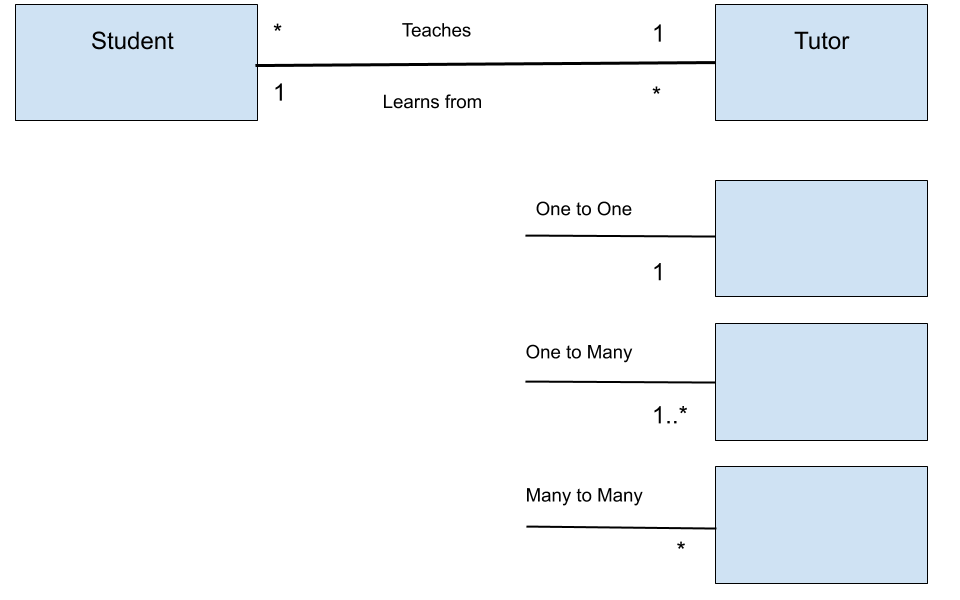
7. Что такое агрегация? Как обозначается?

Особый тип отношений между классами, когда один класс является частью другого.  
  
Например, рабочее место программиста состоит из стула, стола, компьютера и вентилятора, но при удалении класса «рабочее место», у нас просто останутся все эти классы, только по отдельности.  
  
Агрегация показана в виде непрерывной линии с полым ромбом направленным от классов, являющимися частью какого-либо класса к классу-агрегатору.



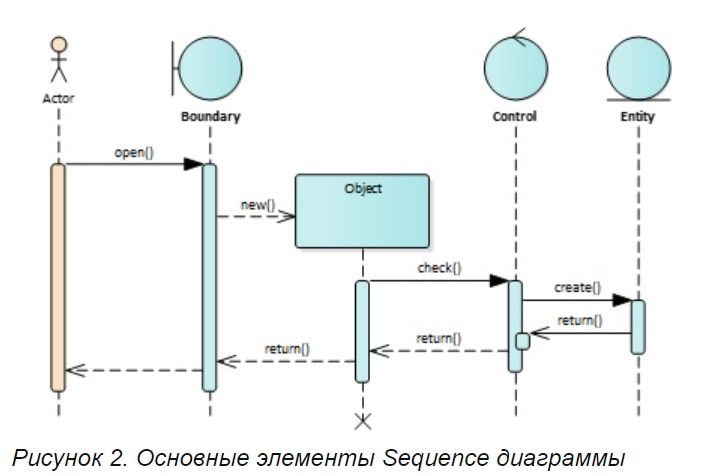
8. Что такое ассоциация?

Аналогично связям, соединяющим объекты, ассоциации соединяют классы. Для того, чтобы между объектами была связь, между ними должна быть ассоциация.  
  
Если предположить, что у нас есть два класса, которые взаимодействуют друг с другом, между ними должна быть проведена непрерывная соединительная линия, обозначающая на схеме ассоциацию. Часто мы также можем увидеть глагол, передающий ее смысл.  
  
Помимо этого, мы также можем указать кратность, то есть число объектов, которые могут принимать участие в отношениях. Кратность задается в виде разделенного запятыми списка интервалов, в котором каждый интервал представлен в виде минимум-максимум.  
  
Например, один студент может учиться у множества преподавателей.  
Но и преподаватель может учить множество студентов.



9. Какие обозначения используют на диаграмме последовательности?

Объекты расположены в горизонтальной последовательности, между ними передаются сообщения. Ось времени ориентирована сверху вниз.  
Элемент Actor может использоваться для представления пользователя, инициирующего поток событий.  
Каждый объект имеет пунктирную линию, называемую "линией жизни", где этот элемент существует и потенциально принимает участие во взаимодействиях. Фокус управления обозначается прямоугольником на линии жизни объекта.  
Сообщения, которыми обмениваются объекты, могут быть нескольких типов, сообщения также могут быть настроены для отражения операций и свойств исходного и целевого элементов.  
Стереотипные элементы, такие как границы (Boundary), элементы управления (Control) и сущности (Entity), могут использоваться для моделирования пользовательского интерфейса (GUI), контроллеров и элементов базы данных, соответственно.  
Повторяющийся поток обмена сообщениями может быть обозначен как фрагмент с типом "loop".



10. Для чего нужна диаграмма последовательности?

Диаграмма последовательности – диаграмма взаимодействия, в которой основной акцент сделан на упорядочении сообщений во времени. Диаграмма последовательности (sequence diagram) — это способ описания поведения системы "на примерах". Диаграмма последовательности (sequence diagram) — диаграмма, предназначенная для представления взаимодействия между элементами модели программной системы в терминологии линий жизни и сообщений между ними.

Т.е. для того, чтобы определить:

* с какими компонентами интерфейса будет взаимодействовать наш пользователь;
* какие управляющие компоненты нам понадобятся;
* что мы будем хранить;
* какими сообщениями будут обмениваться пользователь и компоненты системы для выполнения функции.

Основными элементами диаграммы Sequence являются взаимодействующие объекты с различными стереотипами и связи между ними — взаимодействующие объекты обмениваются между собой некоторой информацией (Рисунок 2).

11. Каково назначение диаграммы использования, пакетов и активности?

**Диаграмма пакетов** предназначена для представления размещения элементов модели в пакетах и спецификации зависимостей между пакетами и их элементами. Как правило, основными предметами языка UML, изображаемыми на этой диаграмме, являются **классы и пакеты**.

**Пакет** – элемент модели, используемый для группировки других элементов модели. Элементы модели, входящие в состав некоторого пакета, называются его членами. Элементами пакета называются элементы модели, которые входят в пространство имен этого пакета. К элементам пакета относят как члены этого пакета, так и члены других импортируемых пакетов.

В UML для этого существуют **диаграммы деятельности(активности)**, являющиеся частным случаем диаграмм состояний. Диаграммы деятельности удобно применять для визуализации алгоритмов, по которым работают операции классов.

Для того чтобы описать различные группы пользователей и их возможности в будущей программе, создаётся так называемая **диаграмма вариантов использования**. Диаграмма вариантов использования. Диаграмма вариантов использования (англ. use-case diagram) – диаграмма, описывающая, какой функционал разрабатываемой программной системы доступен каждой группе пользователей.