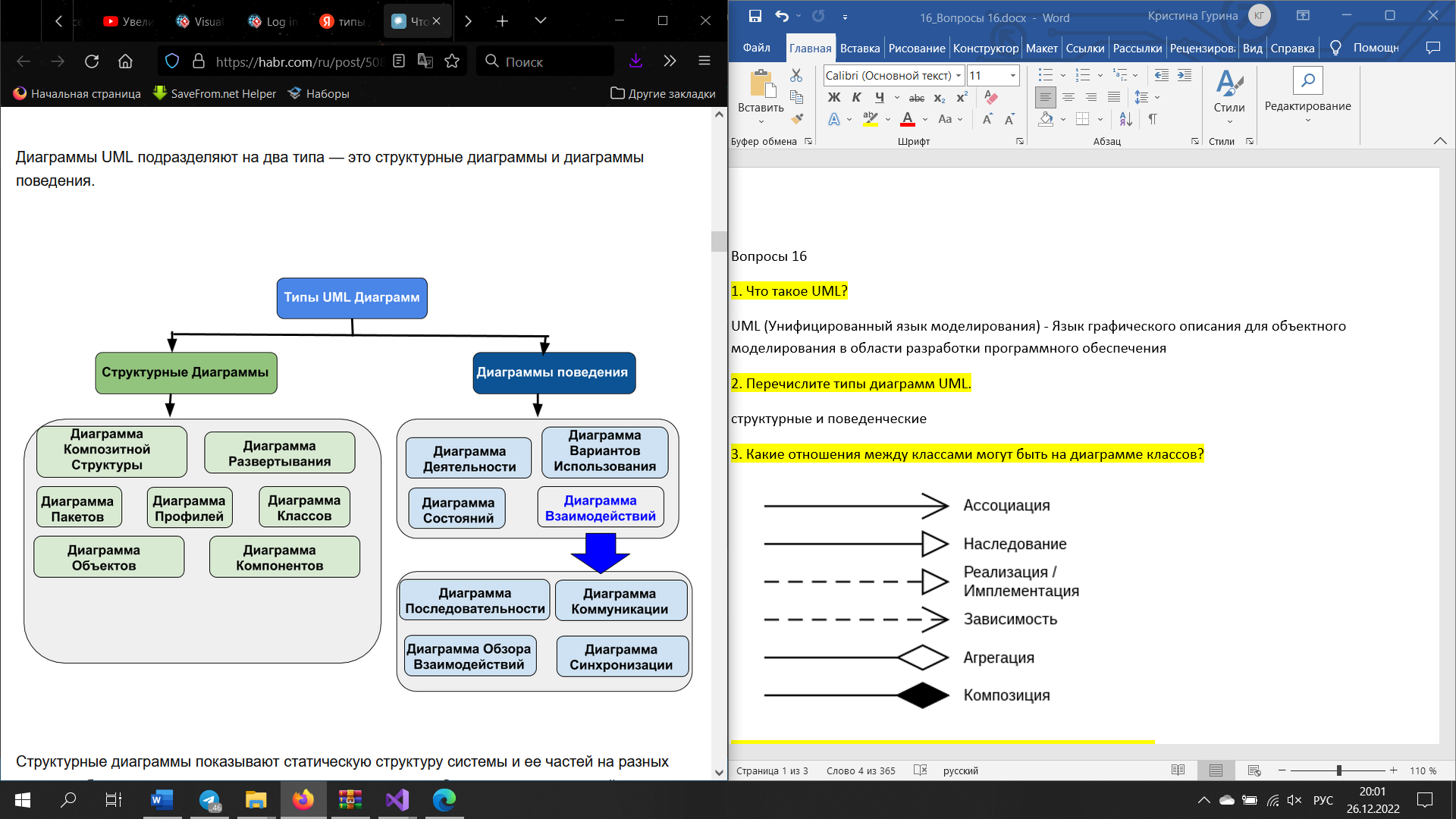
Вопросы 16

1. Что такое UML?

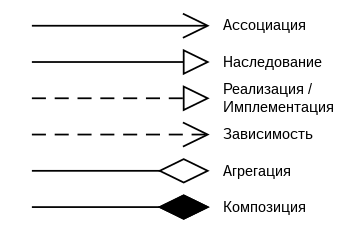
UML (Унифицированный язык моделирования) - Язык графического описания для объектного моделирования в области разработки программного обеспечения

2. Перечислите типы диаграмм UML.

структурные и поведенческие



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



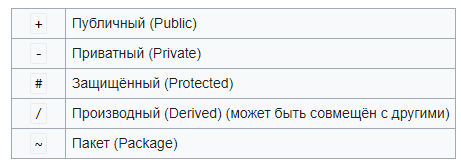
4. Как обозначаются абстрактные классы на диаграмме классов?

полужирным курсивом.

5. Как обозначаются интерфейсы на диаграмме классов?

кружком

6. Как отображается доступность членов класса на диаграмме классов?



7. Что такое агрегация? Как обозначается?

[*Агрегация*](https://ru.wikipedia.org/wiki/%D0%90%D0%B3%D1%80%D0%B5%D0%B3%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)) — это разновидность ассоциации при отношении между целым и его частями. Как тип ассоциации агрегация может быть именованной. Одно отношение агрегации не может включать более двух классов (контейнер и содержимое).

При агрегации реализуется слабая связь, то есть в данном случае объекты Car и Engine будут равноправны. В конструктор Car передается ссылка на уже имеющийся объект Engine. И, как правило, определяется ссылка не на конкретный класс, а на абстрактный класс или интерфейс, что увеличивает гибкость программы.

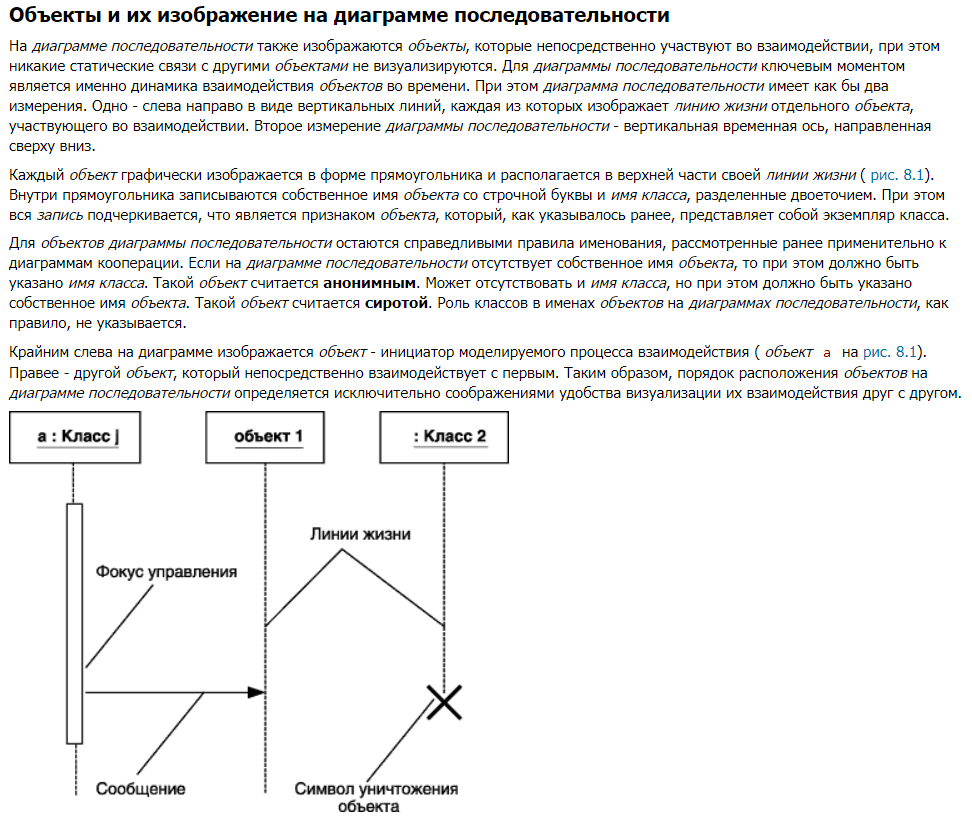
8. Что такое ассоциация?

[*Ассоциация*](https://ru.wikipedia.org/wiki/%D0%90%D1%81%D1%81%D0%BE%D1%86%D0%B8%D0%B0%D1%86%D0%B8%D1%8F_(%D0%BE%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)) показывает, что объекты одной сущности (класса) связаны с объектами другой сущности таким образом, что можно перемещаться от объектов одного класса к другому. Является общим случаем композиции и агрегации.

***Ассоциация*** - это отношение, при котором объекты одного типа неким образом связаны с объектами другого типа. Например, объект одного типа содержит или использует объект другого типа

9. Какие обозначения используют на диаграмме последовательности?

<https://creately.com/blog/ru/%D0%B4%D0%B8%D0%B0%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B0/%D1%83%D1%87%D0%B5%D0%B1%D0%BD%D0%BE%D0%B5-%D0%BF%D0%BE%D1%81%D0%BE%D0%B1%D0%B8%D0%B5-%D0%BF%D0%BE-%D0%BF%D0%BE%D1%81%D0%BB%D0%B5%D0%B4%D0%BE%D0%B2%D0%B0%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D0%BE%D0%B9/>



10.Для чего нужна диаграмма последовательности?

Диаграмма последовательности моделирует взаимодействие объектов на основе временной последовательности. Она показывает, как одни объекты взаимодействуют с другими в конкретном прецеденте.

позволяет визуализировать временные отношения между передаваемыми *сообщениями*. С помощью *диаграммы последовательности* можно представить взаимодействие элементов модели как своеобразный временной *график* "жизни" всей совокупности *объектов*, связанных между собой для реализации варианта использования программной системы, достижения бизнес-цели или выполнения какой-либо задачи.

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**.

Диаграмма пакетов — это **структурная схема UML, которая показывает пакеты и зависимости между ними**.

Она позволяет отображать различные виды системы, например, легко смоделировать многоуровневое приложение.

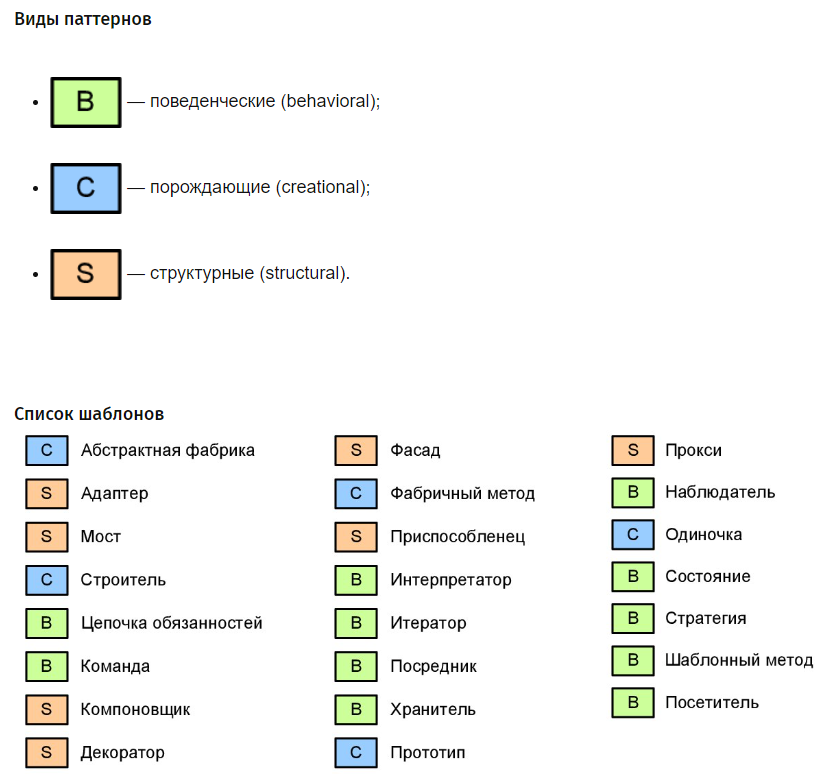
Для того чтобы описать различные группы пользователей и их возможности в будущей программе, создаётся так называемая ***диаграмма вариантов использования***. Диаграмма вариантов использования. Диаграмма вариантов использования (англ. use-case diagram) **– диаграмма, описывающая, какой функционал** разрабатываемой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

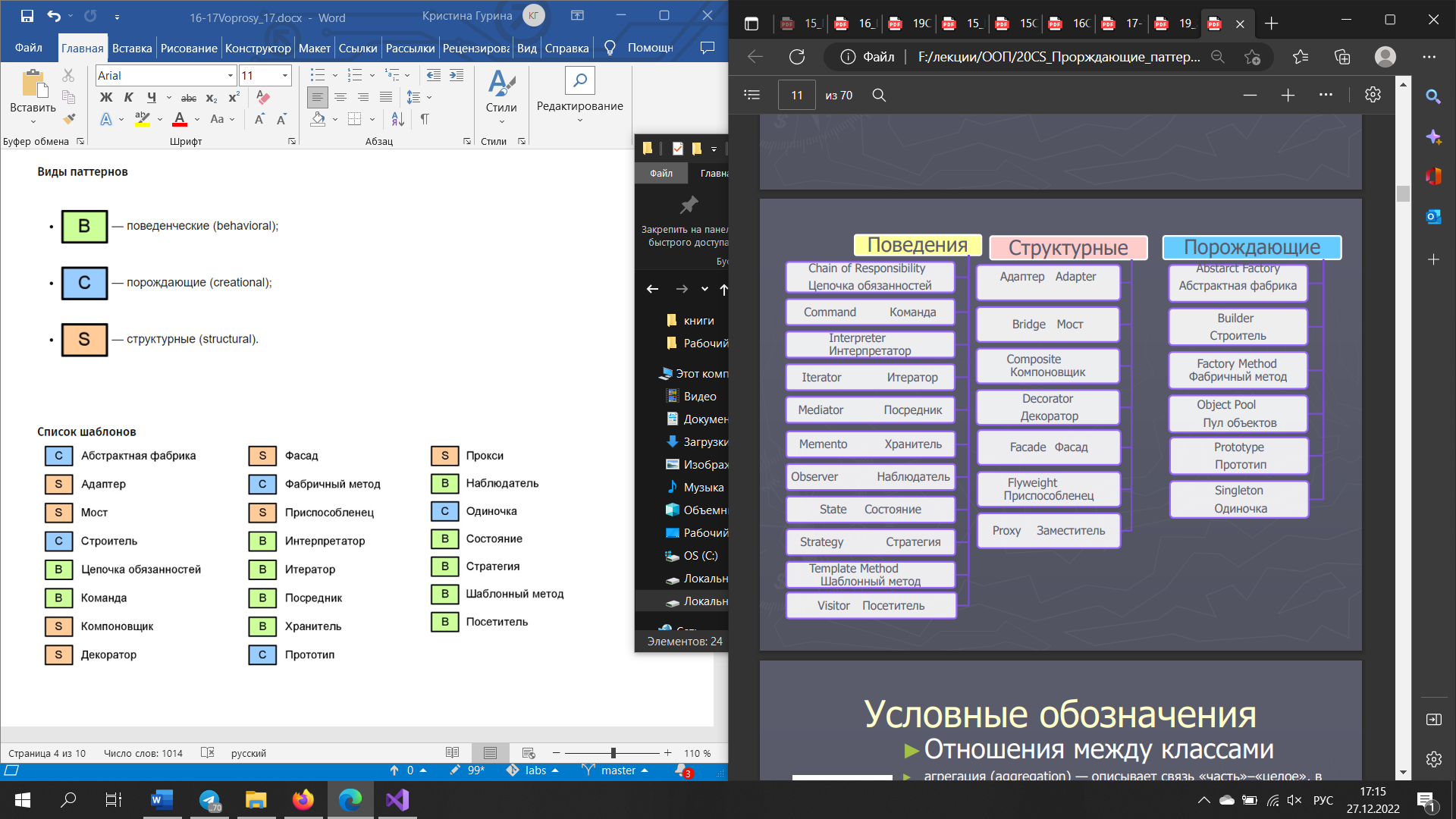
***Диаграмма деятельности (диаграмма активности***) позволяет более детально визуализировать конкретный случай использования.

позволяет моделировать последовательности бизнес-процессов или действий, реализуемых методами классов. Указанные последовательности могут представлять собой альтернативные ветви процесса обработки данных или ветви, которые могут выполняться параллельно.

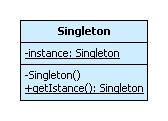
**Вопросы 17-18**

1. Какие типы паттернов бывают?





2. Нарисуете диаграмму классов и поясните принцип работы паттерна Singleton. Назовите условия применения.

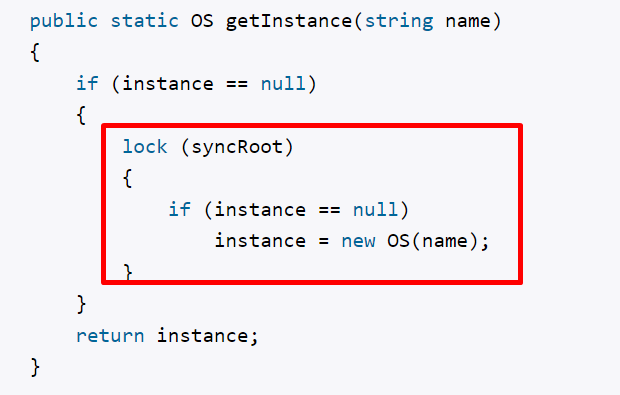
Одиночка (Singleton, Синглтон) - порождающий паттерн, который гарантирует, что для определенного класса будет создан только один объект, а также предоставит к этому объекту точку доступа.

Когда надо использовать Синглтон? Когда необходимо, чтобы для класса существовал только один экземпляр

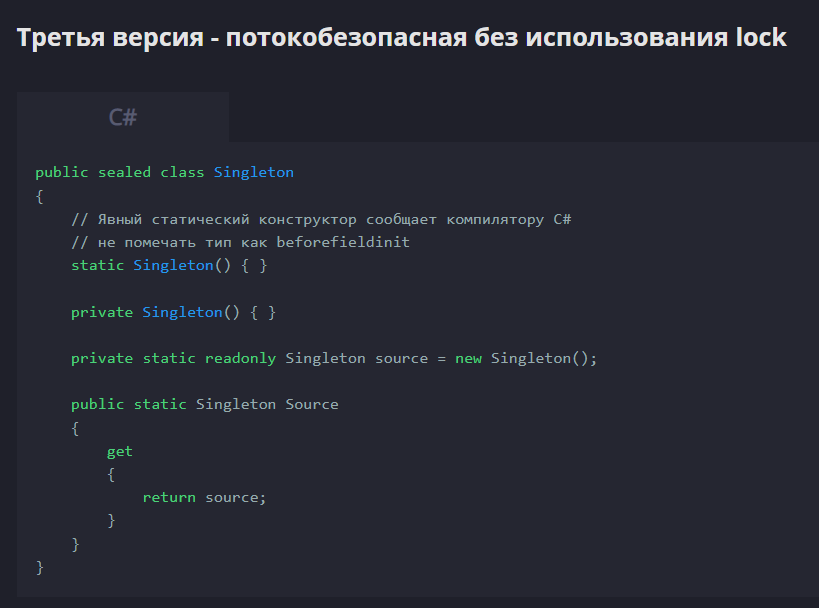
Синглтон позволяет создать объект только при его необходимости. Если объект не нужен, то он не будет создан. В этом отличие синглтона от глобальных переменных.

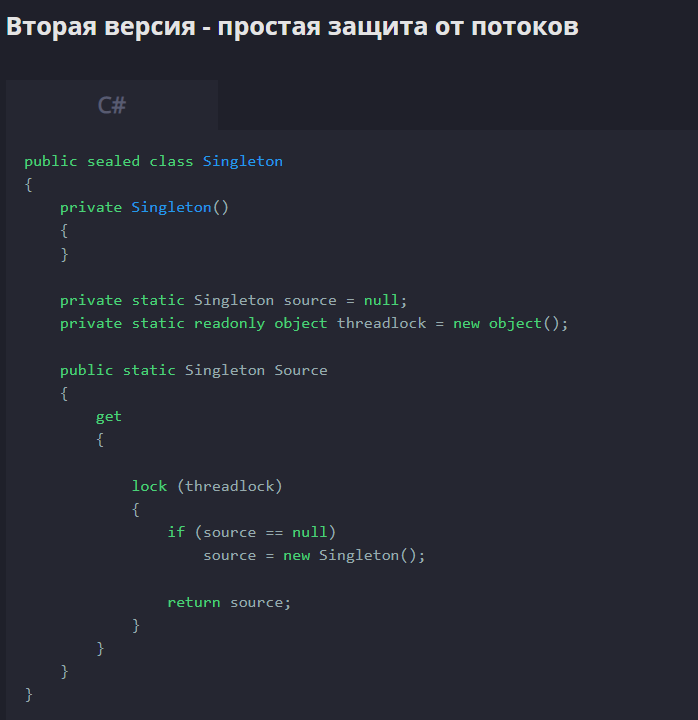
3. Как сделать потокобезопасную, с отложенной реализацией версию Singleton.

Чтобы избежать одновременного доступа к коду из разных потоков критическая секция заключается в блок **lock**.



Определение объекта синглтона в виде статического поля класса открывает нам дорогу к созданию **Lazy-реализации** паттерна Синглтон, то есть такой реализации, где данные будут инициализироваться только перед непосредственным использованием.





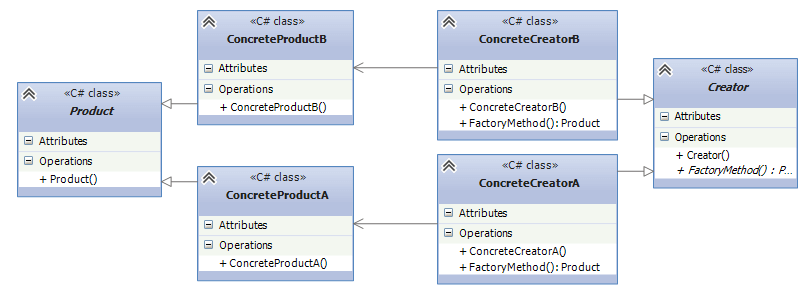
4. Нарисуете диаграмму классов и поясните принцип работы паттерна Factory Method. Назовите условия применения

Фабричный метод (Factory Method) - это паттерн, который определяет интерфейс для создания объектов некоторого класса, но непосредственное решение о том, объект какого класса создавать происходит в подклассах. То есть паттерн предполагает, что базовый класс делегирует создание объектов классам-наследникам.

### Когда надо применять паттерн

* Когда заранее неизвестно, объекты каких типов необходимо создавать
* Когда система должна быть независимой от процесса создания новых объектов и расширяемой: в нее можно легко вводить новые классы, объекты которых система должна создавать.
* Когда создание новых объектов необходимо делегировать из базового класса классам наследникам

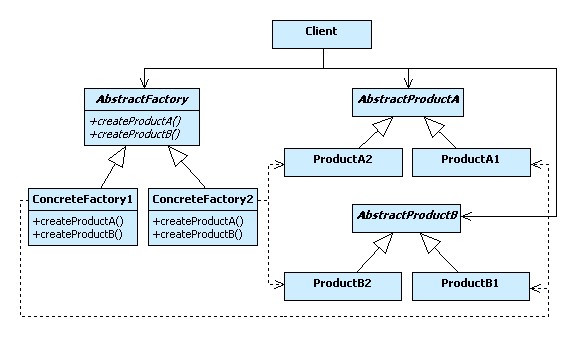
На языке UML паттерн можно описать следующим образом:

5. Нарисуете диаграмму классов и поясните принцип работы паттерна Abstract Factory. Назовите условия применения

Паттерн "Абстрактная фабрика" (Abstract Factory) предоставляет интерфейс для создания семейств взаимосвязанных объектов с определенными интерфейсами без указания конкретных типов данных объектов.

**Когда использовать абстрактную фабрику**

* Когда система не должна зависеть от способа создания и компоновки новых объектов
* Когда создаваемые объекты должны использоваться вместе и являются взаимосвязанными

С помощью UML абстрактную фабрику можно представить следующим образо

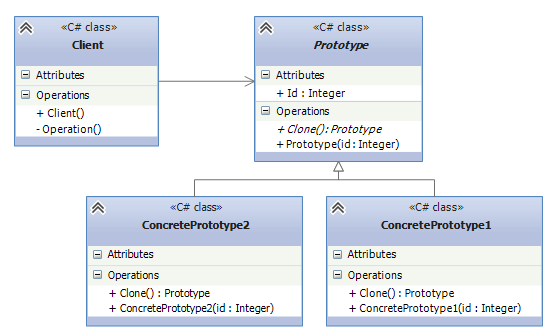
6. Нарисуете диаграмму классов и поясните принцип работы паттерна Prototype. Назовите условия применения

Паттерн Прототип (Prototype) позволяет создавать объекты на основе уже ранее созданных объектов-прототипов. То есть по сути данный паттерн предлагает технику клонирования объектов.

Когда использовать Прототип?

* Когда конкретный тип создаваемого объекта должен определяться динамически во время выполнения
* Когда нежелательно создание отдельной иерархии классов фабрик для создания объектов-продуктов из параллельной иерархии классов (как это делается, например, при использовании паттерна Абстрактная фабрика)
* Когда клонирование объекта является более предпочтительным вариантом нежели его создание и инициализация с помощью конструктора. Особенно когда известно, что объект может принимать небольшое ограниченное число возможных состояний.

На языке UML отношения между классами при применении данного паттерна можно описать следующим образом:



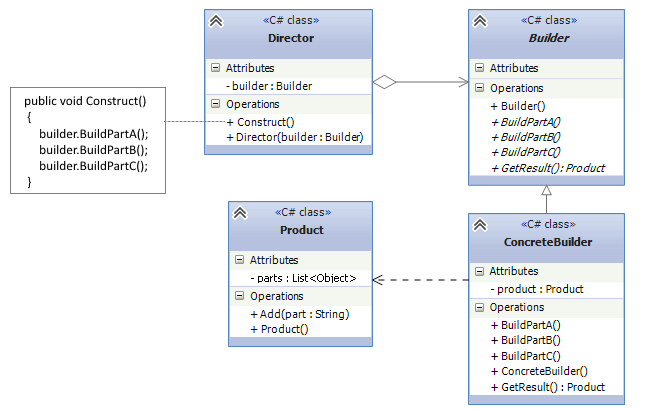
7. Нарисуете диаграмму классов и поясните принцип работы паттерна Builder. Назовите условия применения

Строитель (Builder) - шаблон проектирования, который инкапсулирует создание объекта и позволяет разделить его на различные этапы.

### Когда использовать паттерн Строитель?

* Когда процесс создания нового объекта не должен зависеть от того, из каких частей этот объект состоит и как эти части связаны между собой
* Когда необходимо обеспечить получение различных вариаций объекта в процессе его создания

Формально в UML паттерн мог бы выглядеть следующим образом:



8. Что такое Пул объектов (Object pool), в чем суть паттерна?

Объектный пул (англ. object pool) — порождающий шаблон проектирования, набор инициализированных и готовых к использованию объектов. Когда системе требуется объект, он не создаётся, а берётся из пула. Когда объект больше не нужен, он не уничтожается, а возвращается в пул.

9. Какое основное назначение паттерна отложенная инициализация (Lazy initialization)?

Отложенная инициализация объекта означает, что его создание откладывается до первого использования. (В этом разделе термины «ленивая» инициализация и «ленивое» создание экземпляра являются синонимами.) Отложенная инициализация в основном используется для повышения производительности, предотвращения непроизводительна вычислений и сокращения требований к памяти программы.