1. Перечислите и поясните принципы ***SOLID***.

**S: Single Responsibility Principle (Принцип единственной ответственности).**

Класс должен быть ответственен лишь за что-то одно.

**O: Open-Closed Principle (Принцип открытости-закрытости).**

Программные сущности (классы, модули, функции) должны быть открыты для расширения, но не для модификации.

**L: Liskov Substitution Principle (Принцип подстановки Барбары Лисков).**

Необходимо, чтобы подклассы могли бы служить заменой для своих суперклассов.

Цель этого принципа заключаются в том, чтобы классы-наследники могли бы использоваться вместо родительских классов, от которых они образованы, не нарушая работу программы. Если оказывается, что в коде проверяется тип класса, значит принцип подстановки нарушается.

**I: Interface Segregation Principle (Принцип разделения интерфейса).**

Создавайте узкоспециализированные интерфейсы, предназначенные для конкретного клиента.

Клиенты не должны зависеть от интерфейсов, которые они не используют. Этот принцип направлен на устранение недостатков, связанных с реализацией больших интерфейсов.

**D: Dependency Inversion Principle (Принцип инверсии зависимостей).**

*Объектом зависимости должна быть абстракция, а не что-то конкретное.*

1. Модули верхних уровней не должны зависеть от модулей нижних уровней. Оба типа модулей должны зависеть от абстракций.
2. Абстракции не должны зависеть от деталей. Детали должны зависеть от абстракций.
3. Поясните детально, каким образом ***Dependency***

***Inversion Principle*** применен в реализации класса ***Employee***.

С помощью интерфейса IBonus создается промежуточная абстракция с помощью которой мы выносим из

1. Поясните суть паттерна ***Abstract Factory***, что дает его применение.

Абстрактная фабрика — это порождающий паттерн проектирования, который позволяет создавать семейства связанных объектов, не привязываясь к конкретным классам создаваемых объектов.

1. Сравните два паттерна ***Factory Method*** и ***Abstract Factory.***

Определения

В момент определяем наследника(т.е. в процессе создания)  
в абстракт сначала реализуем а потом ток чет добавляем

1. Как (какие классы, методы и как) необходимо доработать библиотеку ***Lec03LibN:***

*-*если появится новый тип вознаграждения *D(x,y,z)= (wH+a)\*cH\*x+y\*z, действующий, только для уровня 2;*

*Создаем класс и в уровень 2 добавляем*

*- если появится новый уровень вознаграждения 4(a,b,c)= (wH+a\*c)\*(cH+b), действующий, только для уровня 1;*

Только в уровне 1 создаем класс

*- если перестанет применяться уровень вознаграждения типа B уровня 3.*

Будет ругаться программа, заглушка, public, можно исключения вызывать