**Inheritance:**

Inheritance in Java is a mechanism in which one object acquires all the properties and behaviors of a parent object. ... The idea behind inheritance in Java is that you can create new classes that are built upon existing classes. When you inherit from an existing class, you can reuse methods and fields of the parent class.

**inheritance** в Java - это механизм, в котором один объект получает все свойства и поведение родительского объекта. ... Идея inheritance в Java заключается в том, что вы можете создавать новые классы, основанные на существующих классах. Когда вы наследуете существующий класс, вы можете повторно использовать методы и поля родительского класса.

**Single Inheritance :** In single inheritance, subclasses inherit the features of one superclass.

Одиночное наследование: при одинарном наследовании подклассы наследуют свойства одного суперкласса.

**Multiple Inheritance (Through Interfaces) :** In Multiple inheritance ,one class can have more than one superclass and inherit features from all parent classes. Java does not support multiple inheritance with classes. In java, we can achieve multiple inheritance only through Interfaces.

**Множественное наследование (через интерфейсы**): при множественном наследовании один класс может иметь более одного суперкласса и наследовать функции от всех родительских классов. Java не поддерживает множественное наследование с классами. В java мы можем добиться множественного наследования только через интерфейсы.

An **abstract class** is a class that is declared abstract —it may or may not include abstract methods. Abstract classes cannot be instantiated, but they can be subclassed.

**Абстрактный класс** - это класс, который объявлен абстрактным - он может включать или не включать абстрактные методы. Абстрактные классы не могут быть созданы, но могут быть по subclassed

**Modifiers of abstruct**

A Java class containing an abstract class must be declared as abstract class. An abstract method can only set a visibility modifier, one of public or protected. That is, an abstract method cannot add static or final modifier to the declarationдклассами.

**Modifers of abstruct:**

Класс Java, содержащий абстрактный класс, должен быть объявлен как абстрактный класс. Абстрактный метод может устанавливать только модификатор видимости: открытый или защищенный. То есть абстрактный метод не может добавлять к объявлению модификатор static или final.

Enum

we use enum keyword anytime we want to reperesent a fixed set of constands for example days of the week or month of the years or so on

мы используем ключевое слово enum каждый раз, когда хотим представить фиксированный набор констант, например дни недели или месяцы лет или так далее.

public enum Day {

SUNDAY, MONDAY, TUESDAY, WEDNESDAY,

THURSDAY, FRIDAY, SATURDAY

}

Static and final

**static** means there is only one copy of the variable in memory shared by all instances of the class. The final keyword just means the value can't be changed. Without **final** , any object can change the value of the variable.

**static** означает, что в памяти есть только одна копия переменной, совместно используемая всеми экземплярами класса. **final** ключевое слово просто означает, что значение нельзя изменить. Без final любой объект может изменить значение переменной.

Overloading

Method overloading allows the method to have the same name which differs on the basis of arguments or the argument types. It can be related to compile-time polymorphism. Following are a few pointers that we have to keep in mind while overloading methods in Java

**Overloading** метода позволяет методу иметь одно и то же имя, которое различается в зависимости от аргументов или типов аргументов. Это может быть связано с полиморфизмом времени компиляции. Ниже приведены несколько указателей, которые необходимо учитывать при перегрузке методов в Java.

**Overriding:**

Inheritance in java involves a relationship between parent and child classes. Whenever both the classes contain methods with the same name and arguments or parameters it is certain that one of the methods will override the other method during execution.

Overriding в java включает отношения между родительскими и дочерними классами. Если оба класса содержат методы с одинаковым именем и аргументами или параметрами, несомненно, что один из методов переопределит другой метод во время выполнения.

Tostring():

A toString() is an in-built method in Java that returns the value given to it in string format. Hence, any object that this method is applied on, will then be returned as a string object.

ToString () - это встроенный метод в Java, который возвращает значение, заданное ему в строковом формате. Следовательно, любой объект, к которому применяется этот метод, будет возвращен как строковый объект.

Hashcode():

The **hashCode** method is an inbuilt method that returns the integer hashed value of the input value. Here are a few key concepts to remember: Multiple invocations of the hashCode must return the same integer value within the execution of a program unless the Object used within the equals method changes.

Метод **hashCode** - это встроенный метод, который возвращает целочисленное хешированное значение входного значения. Вот несколько ключевых концепций, которые следует запомнить. Несколько вызовов hashCode должны возвращать одно и то же целочисленное значение во время выполнения программы, если только объект, используемый в методе equals, не изменится.

Equal();

The java string equals() method compares the two given strings based on the content of the string. If any character is not matched, it returns false. If all characters are matched, it returns true.

Метод java string equals () сравнивает две заданные строки на основе содержимого строки. Если какой-либо символ не найден, возвращается false. Если все символы совпадают, возвращается истина.

Solid principle:

1: Single Responsibility Principle¶

Single Responsibility Principle or SRP states that every class should have a single responsibility. There should never be more than one reason for a class to change.

Single responsibility principle или SRP гласит, что каждый класс должен иметь единственную ответственность. У класса никогда не должно быть более одной причины для изменения.

2-Open/Closed Principle¶

Open/Closed Principle or OCP states that software entities should be open for extension, but closed for modification.

Принцип открытости / закрытости или OCP гласит, что программные объекты должны быть открыты для расширения, но закрыты для модификации.

3-Liskov Substitution Principle¶

Liskov Substitution Principle or LSP states that objects in a program should be replaceable with instances of their subtypes without altering the correctness of the program.

Liskov Substitution Principle или LSP гласит, что объекты в программе должны быть заменены экземплярами их подтипов без изменения правильности программы.

**4-Interface Segregation Principle¶**

Interface Segregation Principle or ISP states that many client-specific interfaces are better than one general-purpose interface. In other words, you should not have to implement methods that you don’t use.

**Interface segregation principle** или ISP утверждает, что многие клиентские интерфейсы лучше, чем один интерфейс общего назначения. Другими словами, вам не нужно реализовывать методы, которые вы не используете.

5-Dependency Inversion Principle¶

Dependency Inversion Principle or DIP has two key points:

* Abstractions should not depend upon details;
* Details should depend upon abstractions.

This principle could be rephrased as use the same level of abstraction at a given level.

Dependency inversion principle или DIP имеет два ключевых момента:

* Абстракции не должны зависеть от деталей;
* Детали должны зависеть от абстракций.

Этот принцип можно перефразировать как использование одного и того же уровня абстракции на данном уровне.