**0.Overview**

- OOP is a programming paradigm based on the concept of object

- paradigm (начин на мислене/ начин на създаване)  
- Class – шаблон , по който се създават обектите.  
- new – дава инстанция или нов обект

- Object   
 - properties  
 - methods

**- Encapsulation** (цялата информация за класа трябва да остава скрита за него)

- Class data should be hidden.

- Accessed only through public methods

**- Inheritance:**

- Process where one class acquires the properties of another  
 - Information is made manageable in a hierarchical order

**- Polymorphism** (един обект може да приема различни форми)

- Ability of an object to take on many forms  
 - Parent class reference is used to refer to a child class object

**- Cohesion and Coupling(Strong cohesion and loose coupling)**

- Coupling

- How related are two modules/classes

- A class/module cannot be used without another class  
- A change in a class will affect all classes it is coupled to

**1.Defining Classes**

- Abstract Data Type (Hide Details from the Client)

- Defining Smple Classes

- Class is a concrete implementation of an ADT  
- Classes provide structure for describing and creating objects

- Class Members

- Class is made up of state and behavior (Класовете дефинират дадено състояние и поведение, а обектите ги съдържат(имат) )

- Field store states

- Methods describe behaviours

- Creating an Object  
 - A class can have many instances(objects)

- Classes vs Objects

- Classes provide structure for describing and creating objects

- An object is single instance of a class

- Modifiers  
 - Classes and class members have modifiers

- Modifiers define visibility  
- Getters and Setters

- Used to create accessors and mutators(-Getters and Setters)

- Constructors (връщат инстанция на дадения клас)

- Special method, executed during object creation

- Static Members (неща, които не са закачени към конкретната инстанция, а към класа)

- static members are shared class - wide

**2.Encapsulation (Hiding Implementation)**

- Process of wrapping code and data together into a single uni

- Object fields must be private

- Use getters and setters for data access  
- Keyword this  
 - this is reference to the current object  
 - this can refer current class instance variable  
 - this can invoke current class method  
- Private

- object hides data from the outside world  
- classes and interfaces cannot be private  
- Data can be accessed only within the declared class itself

- Protected  
 - Grand access to subclasses in other package

- Default  
 - Do not explicitly declare an access modifier  
 - Available to any other class in the same package  
- Public

- Grand access to any class belonging to the Java Universe  
- Import a package if you need to use a class  
- The main() method of an application has to be public

- Validation

- Data validation happens in setters

- Printing with System.out couples your class  
 - Client can handle class exception  
 - Constructours use private setter with validation logic  
 - Guarantees valid state of object in its creation

- Guarantees valid state for public setters

- Immutable Objects

- Immutable == value cannot be changed

- Mutable Objects

- You can change state of objects by their reference

- Mutable Fields  
- Keyword final

- final class can’t be extended

- final method can’t be overridden

Encapsulation – Benefits  
 - Reduces complexity

- Structural changes remain local

- Allows validations and data binding

**3. Inheritance (Extending Classes)**

- Superclass

– Parent class, Base Class

- The class giving its members to its child class

- Subclass – Child class, Derived Class

- The class taking members from its base class

- Inheritance in Java  
 - Java supports inheritance through extends keyword

- Using Inherited Members

-You can access inherited members as usual

- Reusing Constructors

- Constructors are not inherited

- Constructors can be reused by the child classes

- Inheritance has a transitive relation

- може да наследим само един клас

- Multiple Inheritance

- In Java there is no multiple inheritance

- Only multiple interfaces can be implemented

- Access to Base Class Members

- Use the super keyword  
- Inheritance and Access Modifiers

- Derived classes can access all public and protected members

- Derived classes can access default members if in same package

- Private fields aren’t inherited in subclasses (can’t be accessed)

- Shadowing Variables

- Derived classes can hide superclass variables

- Overriding Derived Methods

- A child class can redefine existing methods

- Inheritance Benefits – Abstraction

-Types of Class Reuse

-Extension  
 - Duplicate code is error prone

- Reuse classes throught extension

- Sometimes the only way

- Composition (клас , в който има други класове)/(обект, в който има други обекти)

- Using classes to define classes

- Delegation

Class Laptop {

Monitor monitor;

Void incrBrightness(){

Monitor.brighten();

}

}

**4.Polymorphism**

- This is something similar to word having several different meanings depending on the context.

- Polymorphism

- Ability of an object to take on many forms.

- Reference Type and Object Type

- List<String>(referenceType) = new ArrayList<>();(ObjectType)

- Variables are saved in reference type

- You can use only reference methods

- If you need object method you need to cast it or override it

- Types of Polymorphism

- Runtime polymorphism

- using of override method

- Compiletime polymorphism (also know as Static Polymorphism)

- Rules for Overloading Method

- Overloading can take place in the same class or in its sub – class

- Constructor in Java can be overloaded

- Overloaded methods must have a different argument list

- Overloaded methods should always be the part of the same class (can also take place in sub class), with same name but different parameters

- They may have the same or different return types

- Rules for Overriding Method

- Overriding can take place sub – class.

- Argument list must be the same as that of the parent method

- The overriding method must have same return type

- Access modifier cannot be more restrictive

- Private, static and final methods can NOT be overridden

- The overriding method must not throw new or broader checked exceptions.

- Abstract Classes

- Abstract class can NOT be instantiated // Compile time error

- An abstract class may or may not include abstract methods

- If it has at least one abstract method, it must be declared abstract

- To use abstract class, you need to extend it