TABLE OF CONTENTS - NCR TR

NCR PREPARATION

Note

You can find some of the implementations in the following github repository please check it out.

Static Binding

- To identify the method executed for a particular method call is known as binding
- Binding which can be resolved at compile time by compiler
- Static binding also called as early binding
- Ex: method overloading

Dynamic Binding

- When compiler is not able to resolve the binding at compile time
- It is also called as late binding
- Ex:method overriding

Object Oriented Programming

- OOP's is a methodology or paradigm to design a program using classes and objects
- Objects are real world entities
- Classes are blueprints of an object that contains properties (Attributes) or behaviours (Activities)
- Diff objects of a class have same prop with different values

OOP's	Procedural
Bottom up Approach	Top down Approach
Has Access modifiers , so it has much security	Doesn't have access modifiers
Supports Overloading	Doesn't support overloading
Divided into objects	Divided into functions

1. Encapsulation
2. Inheritance
3. Abstraction
4. Polymorphism

Methods:

- We can’t call a non-static method from a static method
- Each method has a signature i.e the number of parameters and type of that parameters
- Method overloading
  - When a class has same method name with different parameters is known as method overloading

Pass by value vs Pass by reference

- The method parameters are copied to another variable and then the copied object is passed this is known as pass by value
- In alias a reference to the actual parameter is passed to the method, that’s why it is called as Pass by reference
- **JAVA IS ALWAYS PASS BY VALUE**

Constructor

- Constructor is defined as a method and there is no return type and name should be same as Class name
- Whenever an object is created the constructor is called
  - If a constructor doesn’t accept arguments it is called as No-argument constructor
  - If we don’t define a Constructor in class , Java compiler defines a default constructor
  - If we want to customise the attributes depending on the object we use a parameterised constructor
  - Constructor overloading is possible
  - After we create a constructor we can’t call a default constructor (if we define a default constructor then it is possible)

Packages and Access Modifiers

- There are many built-in packages like util,lang,javax,swing,io etc
- Packages are used to categorize classes and interfaces
- Provides access protection
- We can’t have same Class names in same package

Access Modifiers

- Private,Protected,Public and default
- We can change the access level of a method,constructor and class by applying access modifier on it
- If we keep private access modifier we can access it only within the class
- If we don’t specify any access modifier it can be accessed within the same package
- Protected keyword - those can be accessed within the class and it’s children only

Encapsulation

- Encapsulation is a process of wrapping code and data together into a single unit. The data in a class is hidden from other classes so it is known as data hiding.
- Related fields and methods should be together
- Make the fields private
- Set the setter and getter methods as public

Static Keyword

- Public static void main -- we have been using this in every program
- Static keyword is related to class but not Object
- Static keyword is applicable to methods , variables , blocks and nested class
- In a non static inner type we can’t declare a static variable

- We cannot declare a top level class as static but we can declare a nested class as static
- Nested classes are defined as static because we don't need to create an object of top level class, if not mentioned as static we need to create an object of top level and then we can create nested class objects.
- Static Block is executed as soon as Class is called i.e executed before main

Inheritance

- Inheritance represents a IS-A relationship
- Inheritance is property of an object to acquire properties of its parent object
- Reusing the code

Protected:

- Protected is an access modifier and we can make methods and fields as protected
- Protected members are accessible within the class
- In the same package
- Within its subclasses

Multiple inheritance is not possible in java

Method Overriding

- Same name and same arguments is known as method overriding
- Method overriding bcz it is able to specify the behaviour of the subclass type
- We can't override static methods because
    - Static methods are bonded at compile time
    - The subclass hides the static method but not override them

Super Keyword

- In java super keyword is used to refer the immediate super class of a child

Why no Multiple Inheritance in Java

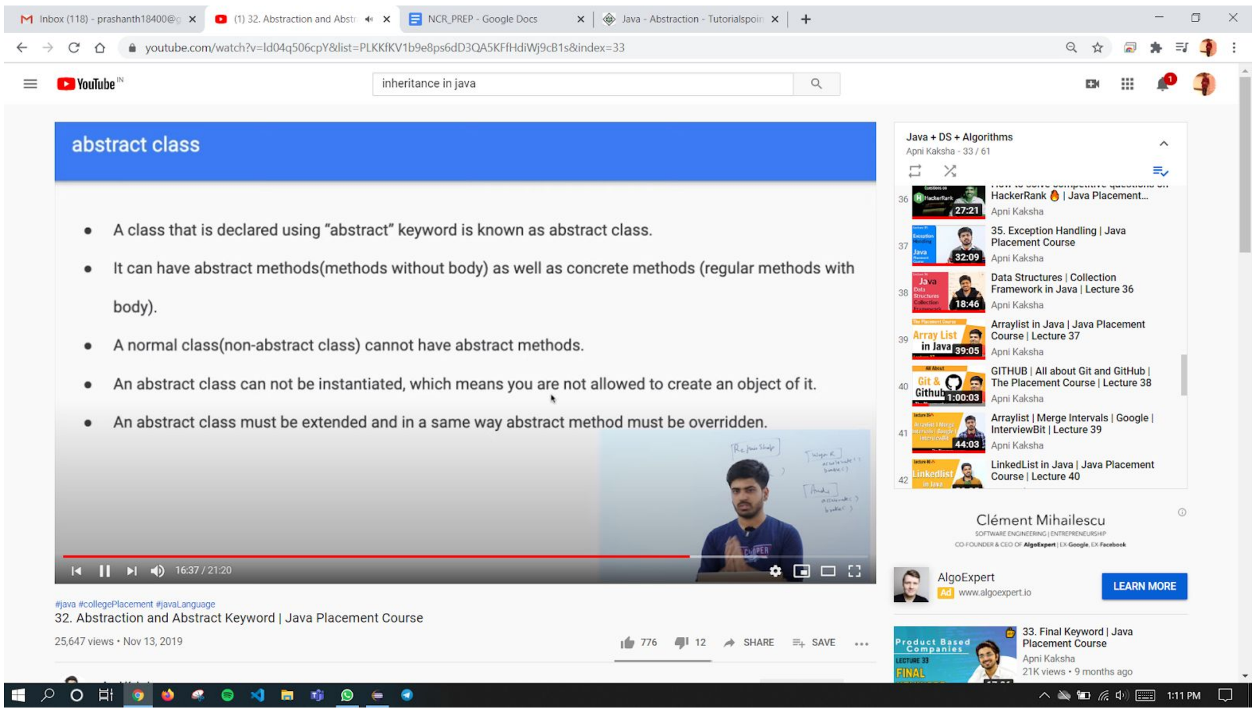
- One subclass can’t inherit two or more parent classes
- Because two classes can have different ways of doing same thing
- And subclass cannot pick one and this results in **Diamond Pattern**

Polymorphism

- Existing in more than one form
- Method overriding is runtime polymorphism
- Late binding or dynamic binding is also known as Runtime Polymorphism
- Early binding or Static binding is also known as Compile time Polymorphism
- Compile time polymorphism is faster than Runtime polymorphism
- Variables can’t be override they can only be hided

Abstraction

- Reduce the complexity
- Abstraction can be achieved by Abstract classes and Interfaces
- Abstraction is about hiding unwanted info and giving most essential information



abstract class

- A class that is declared using "abstract" keyword is known as abstract class.
- It can have abstract methods(methods without body) as well as concrete methods (regular methods with body).
- A normal class(non-abstract class) cannot have abstract methods.
- An abstract class can not be instantiated, which means you are not allowed to create an object of it.
- An abstract class must be extended and in a same way abstract method must be overridden.

#java #collegePlacement #javaLanguage
32. Abstraction and Abstract Keyword | Java Placement Course
25,647 views • Nov 13, 2019

youtube.com/watch?v=Id04q506cpY&list=PLKKfKV1b9e8ps6dD3QA5KFfHdiWj9cB1s&index=33
inheritance in java
16:37 / 21:20
776
12
SHARE
SAVE
Java + DS + Algorithms
Apni Kaksha - 33 / 61
HackerRank | Java Placement...
35. Exception Handling | Java Placement Course
Data Structures | Collection Framework in Java | Lecture 36
Arraylist in Java | Java Placement Course | Lecture 37
GITHUB | All about Git and GitHub | The Placement Course | Lecture 38
Arraylist | Merge Intervals | Google | InterviewBit | Lecture 39
LinkedList in Java | Java Placement Course | Lecture 40
AlgoExpert
LEARN MORE
33. Final Keyword | Java Placement Course
1:11 PM

Final Keyword in Java

- We can assign variable, methods and classes
- A final method can’t be overloaded
- A final class cannot be extended
- A final variable can only be changed
- We must initialise a final variable else we get a compiler error
- It is a good practice to make all final variable in CAPS

Interface

- An interface gives a set of specification that are to be overridden by the sub classes
- Multiple inheritance is not there in Java
- We use implements keyword to Implement a Interface
- Why interface
    - Class can implement multiple interfaces
    - This allows us to get the functionality of multiple inheritance
- We can’t instantiate an interface in java
- Interfaces can’t have a method with body
- By default all interfaces methods are static final and abstract,final

Exception Handling

- Java being a object oriented programming language whenever an exception occurs it creates an exception object
- JRE tries to find someone that can handle the raised exception

Collections

- Conversion of primitives to Objects is known as Auto-Boxing
- Class is defined for a specific purpose
- Whereas Interface is a collection of abstract methods
- In a treeset the values are stored in sorted order
- So in a treeset we can’t store heterogeneous data types
- If we try to store heterogenous objects in treeset it throws an ClassCastException
- Enumeration can only do read operations
- So Iterator has been introduced which can do read and delete operations
- As iterators can’t replace and add new objects we use Listiterator
- List iterator can traverse in both directions
- Adding generics will give compile time safety
- Maps are used to represent key value pairs

Dynamic memory allocations

Malloc()	Allocates required size of byte and returns a pointer to the first byte of allocated space
Calloc()	Allocates space for an array of elements and are initialised to zero and then returns a pointer to the first byte
Free()	Used to free the previously allocated memory
realloc()	Realloc is used to change the size of previously allocated space

Java Class Loader

The **Java ClassLoader** is a part of the **Java Runtime Environment** that dynamically loads Java classes into the **Java Virtual Machine**. The Java run time system does not need to know about files and file systems because of classloaders.

Data Structure

- Data structure is a particular way of organising data so that it can be used efficiently

Stack

- Stack is a data structure in which insertion and deletion are done at same end
- Stack follows LIFO order ie Last in First out
- For efficient implementation of stack using Linked List we perform insertion and deletion and front end only
- Stack is used for
    - Converting infix to postfix
    - Evaluating postfix expression
    - Implementation of recursion

Tree

- A tree is a set of nodes in which one node is defined as root and remaining are classified into disjoint sets
- In a full binary tree all levels except the last are completely filled and each node has 0 or 2 children
- In a complete binary tree all levels except the last are completely filled and the children should be as left as possible