Basics of OOP – object-oriented programming – Java

Class – blueprint / template of the object.

Student, Teacher, ….

Id, name, dob, address, email, gender, grade, mobile…….

Teacher

Student

**Class fields/attributes/columns/properties of class**

Id, name, age, address, email, mobile, gender, qualification, salary, ….

Objects – instance of a class/real world entity

Classname objname = new Classname();

Student student1 = new Student();

Teacher teacher1 = new Teacher();

Before assigning values to objects(stores default values)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Id =0 | Name=null | Address=null | Dob=null | Email=null | Mobile=null | Grade=null |

student1 – 6d8h39ajj39

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Id =0 | Name=null | Address=null | Dob=null | Email=null | Mobile=null | Grade=null |

student2 – 6d8h3jsiw39

After assigning the values to objects

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Id =123 | Name=Ajay | Address=hdheiwksjnsjkdd | Dob=12/12/2008 | Email=ajay123@gmail.com | Mobile+91-898989899 | Grade=6th grade |

student1 – 6d8h39ajj39

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Id =124 | Name=Sanajay | Address=eejekkekdk | Dob=…. | Email=…. | Mobile=… | Grade=.. |

student2 – 6d8h3jsiw39

packages – collection of classes – folder structure of the project.

Access specifiers/modifiers – gives scope(visibility) of class, variables, methods.

1. Public – everyone can access.
2. Private – no one can access, only that class.
3. Protected – sub classes or child classes.
4. Default(no specifiers) – access within the same package.

Keywords :

Static - – it can be directly accessed without the creation of the objects. (it always executes first).

Final - it cannot be changed(it will have fixed values like constants). (final class, final methods, final variables)

This – it refers to the current object values.

Principles/pillars/concepts/features of OOP:

1. Encapsulation- what? – binding the data member variable and code together.(How ? – hiding the data variables and access them using getters and setters).
2. Polymorphism – many + forms

A method/function taking multiple/many forms.

1. Method overloading(compile time polymorphism)
2. Method overriding(run time polymorphism)
3. Inheritance - deriving/inheriting the features of parent(base or super) class to child(sub or derived) class.

Single inheritance :

Parent class

A

Child class

B

Multilevel inheritance

A

B

C

D

E

Hierarchical Inheritance

A

D

C

B

Multiple inheritance : (directly not possible between classes)

Sum() sum()

B

A

C

Which sum() should be used?

I am confusing…. ambiguity

Hybrid inheritance : (directly not possible between classes)

A

C

B

D

|  |  |  |
| --- | --- | --- |
| Vehicle(parent class) | Child classes | Specific to child |
| Start | Air vehicle | Travel on air |
| Stop | Water vehicle | Travel on water |
| Speed/fuel | Road vehicle | Travel on road |

Extends – is used for deriving/inheriting the features of parent class to child class.(class-to-class, interface-to-interface).

Method overriding – process of changing/overriding the implementation in child class which is already present in parent class.

Super – used to refer parent class variables and methods.

Final with inheritance :

1. Final with methods – cannot override the methods.
2. Final with class – cannot perform inheritance.

4.Abstraction- hiding the internal implementation and providing only the external essential information.

Abstract methods – methods which does not contains the body or the implementation.

Abstract class – class which contains atleast one abstract method and any number of concrete methods.

Concrete methods - methods which does contain the body or the implementation.