Q1) Explain OOPS ?

Ans > OOPS stands for Object Oriented Programming System.

This a programming paradigm which structures the program into reusable piess pieces of code blueprints called CLASS.

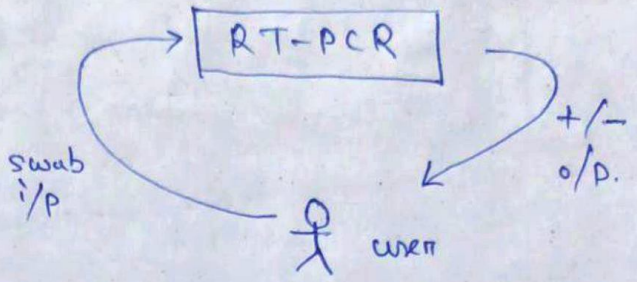
From this reusable blueprints called CLASS, we can create multiple templates of them called OBJECTS.

Class are logical combination of variables and functions but they can be accessed only with objects.

Q2) Explain abstraction? Real life examples.

Ans ⇒ Abstraction means hiding the internal details of an application from the user and presenting before them what should be done with the application.

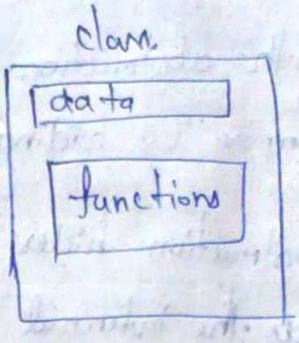
Eg,




The user does not know how the RT-PCR is working but still the user is able to use it.

Q8) What is encapsulation ? Explain with example.

Ans) Encapsulation is defined as binding the data and function into a single unit.




Generally, the class is such designed, that the data cannot be accessed from outside. In order to access the data, one has to go through the functions.

This provides data security to program.

Q4) Explain the relationship between abstraction and encapsulation?

Ans) Both abstraction and encapsulation are for security purpose by hiding/controlling access to the codes.
Abstraction hides the code complexity and encapsulation hides the internal working.

Q5) Explain polymorphism?

Ans) Polymorphism means performing one action in different ways.
In OOP, Inheritance lets us inherit the attributes and methods from another class.
Polymorphism uses those methods to perform different tasks.

Eg

```
class Animal
{
    method AnimalSound ()
    {
    }
}

class Dog extends Animal
// Implementation of method AnimalSound is different //

class Cat extends Animal
// Implementation of method AnimalSound is different. //
```

Q6) Explain Inheritance ?

Ans) It is the property of OOP where one class can inherit the attributes and behaviour from its parent class.

It helps us with code reusuability.

The parent class is called Superclass. / Base class

The child class is called Subclass. / Derived class

Q7) How composition is better than inheritance?

Ans) i) Since Java does not support Multiple Inheritance, in that scenarion composition is better.

ii) Although both composition and inheritance allow to reuse code, but inheritance weakens encapsulation.

When subclass depends on superclass for its functions, the subclass becomes fragile as any change in superclass can affect the functionality of subclass without actually making any change in subclass.

Q8) Which OOPS concept is used as a reuse mechanism?

Ans) Inheritance.

Q9) Which OOPS concept exposes only necessary information to calling function?

Ans) Abstraction.

Q10) Explain a class? Create a class.

Ans) A class is a blueprint for creation of an object. It contains data members and member functions but these are not allocated any physical memory.

A class is created using keyword "class"

```
class <class_name>
{
  Data member;
  Member_function();
  Constructor();
}
```

Eg

```
class Animal
{
  void AnimalSound()
  {
    ==
  }
}
```

Qss) Using above created class explain abstraction and encapsulation?

Ans)
```
class Animal
{
abstract void Sound();
}

class Dog extends Animal
{
    void sound()
    {
    System.out.println("The animal barks");
    }
}

class Lion extends Animal
{
    void sound()
    {
     System.out.println("The animal roars");
    }
}.
```

Inheritance : Class Dog and class Lion inherit all the attributes and behaviour of class Animal.

Abstraction : Method Sound() is not implemented in class Animal. But its implementation is different for class Dog and class Lion.

Q12) Explain difference among class and object?

Ans) The class is a blueprint of an object. They have the structure but they donot have any physical memory allocation.

Object is an instance of a class which acquire physical memory similar to the structure of class.

Q13) Define access modifier?

Ans) The access modifiers are certain keywords in OOP that controls the accessibility of classes, methods and other members.

They are used facilitate the encapsulation of components.

The types are i) Default ii) Public

iii) Private iv) Protected.

Q14) Explain object ? Create an object of a class.

Ans) An object is an instance of a class which allocate physical memory same as the structure defined by the class.

Eg
```
class Animal
{
    void sound()
    {
    }
}

public static void main (String args[])
{
    Animal a1 = new Animal();
}
```

a1 is an object of class Animal.

Q15) Give real life examples of object.

Ans) Class is a logical entity whereas object is physical entity

Eg Animal is a class

but dog, cat etc are objects.

Q16) Explain a constructor?

Ans) A constructor is a special member function of a class which initializes objects of a class.

It donot have any return type and its name is same as class name.

→ Whenever a constructor is called, memorey is allocated for the object.

→ A constructor cannot be abstract, static, final and synchronized.

Q17) Define various types of constructors?

Ans) i) Default Constructor: Whenever no constructor is written, JVM assumes a default constructor and initialize objects with default values.

ii) Parameterized Constructor: Whenever we want to initalize the objects with certain values, we can create parameterized constructor and initialize the object with the values passed in parameter.

Q18) Whether static method can use non-static members?

Ans) Static method can directly access only static members. However we can use object to access non-static members

Q19) Explain Destructor?

Ans) Destructor is an instance member function which is invoked automatically whenever an object is going to be destroyed. JAVA does not have destructor.

Q20) Explain in inline function?

Ans) When an inline function is called, the whole code of the inline functions gets inserted at the point of inline function call.

Q21) Explain virtual function?

Ans) A virtual function is a member function which is declared within a base class and is re-defined (overridden) by a derived class.

Q22) Explain a friend function?

Ans) A friend function of a class is a non-member function defined outside the scope of the class but it has the right to access all private and protected members of the class.

Q23) Explain function overloading?

Ans) It is a feature of OOP where 2 or more functions can have the same name but different parameters.

→ It is a polymorphism where a particular function name is overloaded with different jobs.

Q24) Explain base class, sub-class and super class?

Ans) In OOP, the property of inheritance allows one class to inherit the properties of another class.

The class which inherits are called sub-class and the class whose properties are inherited are called base class / super-class.

Q26) Explain an abstract class?

Ans) In JAVA, an abstract class is a class whose objects cannot be formed.
It is declared using keyword 'abstract'.

Q27) Explain operator overloading?

Ans) Operator overloading gives the ability to use same operator to do various operations.
The function of operator is declared using "operator" keyword

Q28) Define the different types of arguments? (Call by value/ Call by reference).

Ans) When a function pass argument by value, it sends a copy of the arguments. Any change in the value of argument inside function is not reflected in calling function. In call by reference, the reference address of the variables is passed. So any change is reflected in main function

Q29) Explain super keyword?

Ans) The super keyword is used to refer the immediate parent class instance variables and methods. and constructor

Q30) Explain method overriding?

Ans) When the function in subclass and superclass have the same method name, same return type and the same parameter list.

→ We cannot override a method declared as static and final.

→ We should always override an abstract method.

Q31) Difference among overloading and overriding?

Ans) Two functions are said to be overloaded when they are within the same class and when the functions are in different classes [inherited class], it is called method overriding.

Q35) Explain an interface?

Ans) An interface is a blueprint of class. It has static constants and abstract methods.

→ It is a mechanism to achieve abstraction.

→ Interface can have abstract method and abstract variables. It cannot have a method body.

→ Interface represents Is-A Relationship.

Q36) Explain Exception Handling?

Ans) Sometime during a program execution, we may get runtime errors/exception due to which the remaining portion of our program will not execute.

Exception handling is a mechanism to handle the runtime errors so that the normal flow of the application can be maintained.

→ Exception is an object which is thrown at runtime and disrupts the normal flow of the program.

Q37) Explain the difference among structure and class?

Ans) Both structure and class are user-defined data types which can contain a group of different data-types.
But there are some differences between them :→

| Structure | Class |
|---|---|
| i) Structure stores the data values and hence called value types | i) Objects in a class stores the reference to data values and are called reference types. |
| ii) The value types are allocated on stack memory | ii) Reference types are allocated to heap memory. |
| iii) Class can inherit from another class. | iii) Structure cannot inherit from another structure. |
| iv) Datamember of a structure cannot be protected | iv) Datamember of a class can be protected. |
| v) Function member cannot be abstract. | v) Function member can be virtual or abstract. |

Q38) Explain the default access modifier ?

Ans) Access modifiers specifies the accessibility or scope of a field (data member), method, constructor or class.

When the access modifier is default, it means it can be accessed only within the package. It cannot be accessed from outside the package.

When we donot write any keyword for access modifier, it is considered as "default."

Q39) Explain a pure virtual function?

Ans) A virtual function is a member function which is declared in parent class and is re-defined (overriden) in child class.

A pure virtual function (or sometimes called abstract function) is a virtual function which can only be declared in parent class but it cannot have any implementation in parent class.

Its implementation is in child classes only.

Q40) Explain dynamic or runtime polymorphism?

Ans) Runtime polymorphism / Dynamic polymorphism / Dynamic Method Dispatch is a process to resolve the call ambiguity of an overridden method at runtime rather than compile-time

In this process, the overridden method is called through the reference of the parent class. The determination of the method to be called is based on the object being referred to by the reference variable.

Q41) Do we require a parameter for constructor?

Ans) It is not compulsory, but optional

If we donot use any parameters, the data member will be assigned to default values.

Q42) Explain static and dynamic binding?

Ans) Binding is a mechanism for creating a link between a method call and method actual implementation.

→ Private, final and static members (data and functions) use static binding while virtual functions use dynamic binding.

→ Overloaded methods are resolved using static binding whereas overridden methods are resolved using dynamic binding.

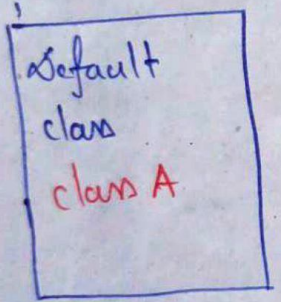
In static binding, binding is resolved at compile time while the same is resolved at runtime for dynamic binding.

Q43) How many instances can be created for an abstract class?

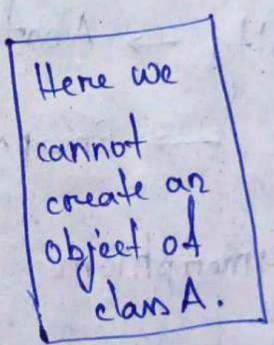
Ans) None.
We cannot create any instance of an abstract class.

Q44) Explain default access specifier in class definition?

Ans) When the specifier to a class is default, then we cannot access the class from a class of any other package.




Package-1




Package-2.

Q45) Which OOPS concept is used in reuse mechanism?

Ans) Inheritance.

Q46) Define the benefits of OOP ?

Ans) i) Large program can be divided into smaller class which makes it easy to debug.

ii) Code reusability → Inheritance.

iii) Function Security → Abstraction.

iv) Data Security → Encapsulation.

v) Flexibility → Polymorphism

Q47) What is method overloading ?

Ans) When more than one method have same name but different arguments types or number of arguments.

Q49) Explain difference among early binding and late binding?

Ans) Early Binding / Static binding : The binding which can be resolved at compile time by compiler.

Late Binding / Dynamic Binding : The binding is resolved in run-time.

Q50) Explain loose coupling and tight coupling ?

Ans) Tight coupling is when a group of classes are highly dependent on one another. It reduces code flexibility and reusability.

Loose coupling is when the dependency of classes are low. It allows user to make change in the code easily.

Q52) Write in breif the abstract class.

Ans) It is a restricted class which cannot have any object or we cannot create object of this class.

Q53) Define the benefits of OOP over POP?

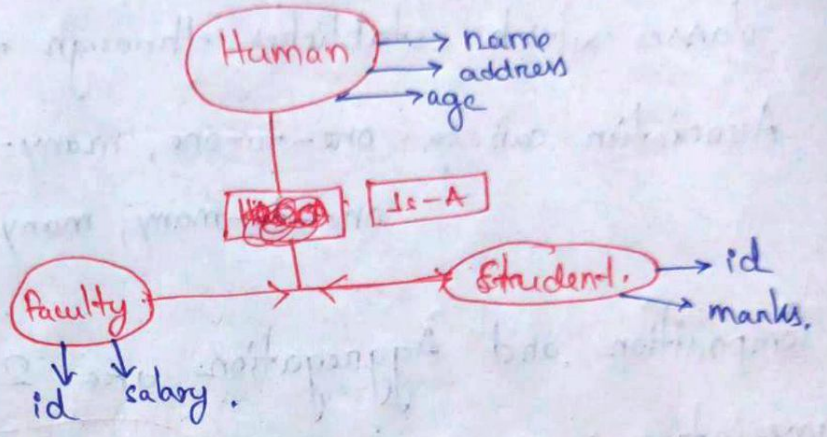
Ans) i) OOP make development and maintenance easier.

ii) It has more security features like abstraction and encapsulation.

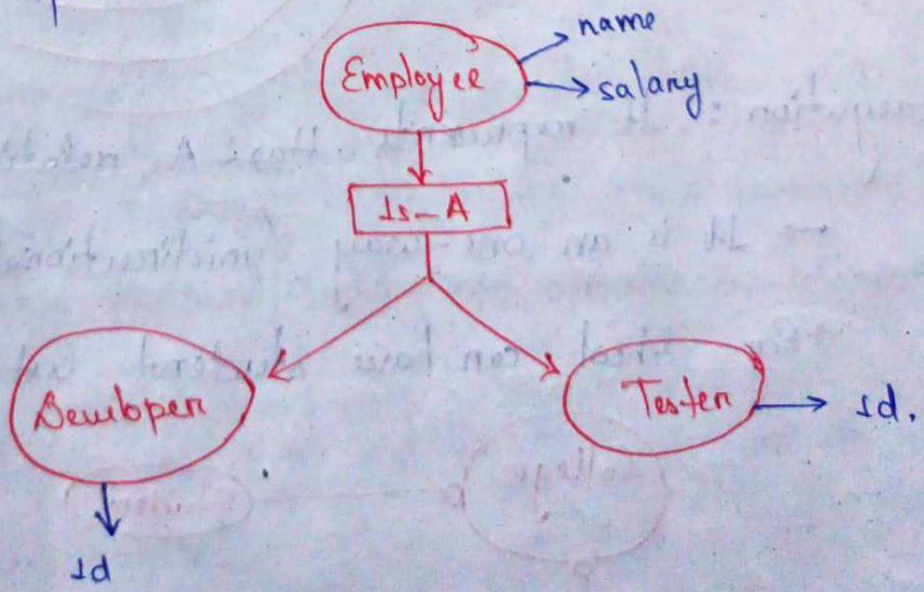
iii) It has the ability to simulate real world objects much more effectively.

Q54) Explain Generalization and Specialization

Ans) Generalization : It is a bottom up approach in which two or more entities can be generalized to a higher level entity.



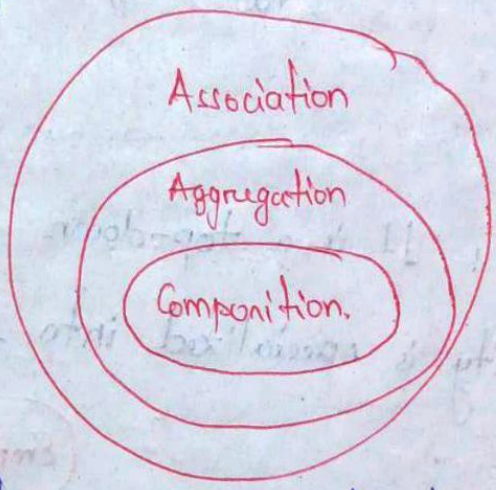

Specialization : It is a top-down approach where higher level entity is specialized into 2 or more lower level entities.



Q55) Write in brief Association, Aggregation and Comporition?

Ans) Association is a relation between 2 seperate classes which establiches through their objects. Association can be one-to-one, many-to-one, one-to-many, many-to-many.

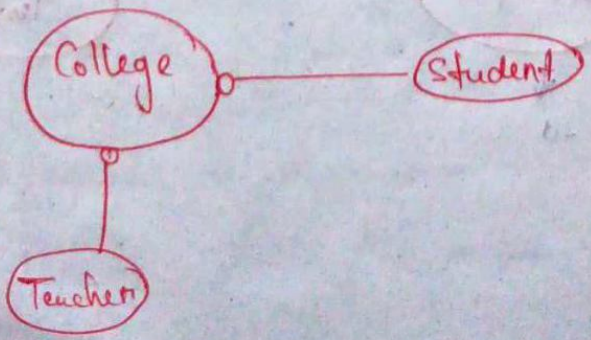
Composition and Aggregation are 2 forms of association.




Aggregation: It represents Has-A relationship.

→ It is an one-way / unidirectional relationship

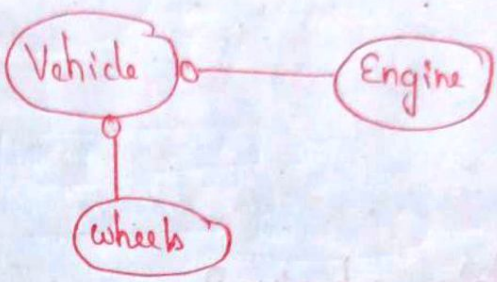
→ Eg School can have student but not vice-versa.



Composition is a restricted form of ~~Association~~ Aggregation in which 2 entities are highly dependent on each other.

→ It represents part-of relationship.

→ In composition, both entities are dependent on each other.




Q5) Write in brief Object Composition Vs Inheritance.

Ans) Establishing Associations between classes is one method that OOP programming uses to encourage code reusability.

There are 2 primary ways to ~~use~~ create these relationships

i) Inheritance : The ~~process~~ process with which an object can take on the properties of one or more other objects.

ii) Composition : The process of utilization of one thing with another object.

Q57) Explain cohesion?

Ans) Cohesion is an OOP principle most closely associated with making sure that a class is designed with a single, well-focused purpose.

Q58) What is Black-Box reuse Vs White-Box reuse?

Ans) Black-Box reuse means using the component without knowing its internals.

White-Box reuse means how the component is implemented. Usually White-Box reuse means class Inheritance.

Q59) Explain "this"?

Ans) "this" is a keyword used to point to any data or method of that class where it is written.

Q60) Write in brief static member and member functions.

Ans) The static member functions are special functions used to access the static data members or other static member functions. It is defined using static keyword.

→ To access the static member function, we use class name or class's objects

→ If any static member function tray to access any non-static data member or non-static member function, it throws an error.

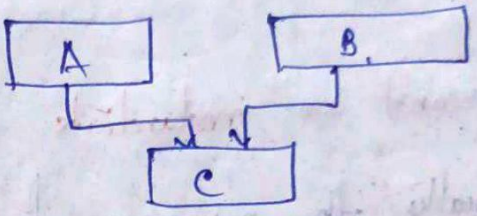
Static Data Member/Static member: The static data members are those attributes of the class which you can access without instantiating the class i.e without object of the class.

Q62) Explain the diamond problem in Java.

Ans) It is related to multiple Inheritance.

Sometimes it is also called as the Deadly Diamond Problem (OR) Deadly Diamond of Death.

Since JAVA does not support multiple inheritance, it doesnot face this Diamond problem. But other OOP Language has this problem.




When A and B both have a same method, and we access that method with object of C, there is an ambiguity and hence compiler shows error.

This ambiguity is diamond problem

Q63) Explain the solution of Diamond Problem.

Ans) The solution of Diamond Problem is ~~i) Abstract method~~

i) Default Method ii) Interface.

Q64) Explain the need of abstract class?

Ans) Although we cannot create any object of abstract class, but we can achieve abstraction with the help of abstract class.

The abstract class can work as base class for the child classes and hence implementing full abstraction.

Q65) Why can't we instantiate abstract class?

Ans) Generally, the purpose of making a class as abstract is that it donot have complete implementation. The implementation of abstract class is in its child classes. Since the class can be incomplete, it is prevented from having object or else it may create any problem.

Q66) Can abstract class have constructors?

Ans) Yes, an abstract class always have a constructor.
If we donot define our own constructor, the compiler will give a default constructor to the abstract class.
Above holds true for all classes - nested, abstract, anonymous etc.

Q67) How many instances can be created for an abstract class?

Ans) None.

Q68) Which keyword is used for overloading?

Q69) Explain the default access specifier in class definition?

Ans) Any member of a class mentioned without any access specifier is considered as default class.
The default class will act as public within same package i,e can be accessed by other classes and will act as private outside the same package.

Q70) Define the operators that cannot be overloaded.

Ans) In C++, the operators that cannot be overloaded are:

i) Scope Resolution Operator (::),

ii) Ternary or conditional Operator (?:)

iii) Dot operator (.)

iv) Pointer to member operator (.*)

v) Object size (sizeof)

vi) Object type (typeid)

~~vii) Static cast~~

vii) static_cast

viii) const_cast

ix) reinterpret_cast

x) dynamic_cast.

Q71, 72, 73, 75 → Repeat.

Q74) Explain ternary operator?

Ans) Ternary operator is a conditional operator

It consists of 3 parts.

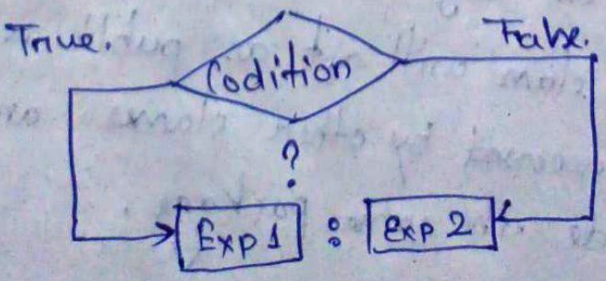
Variable = (condition) ? exp1 : exp2

true

false

Variable = exp1

variable = exp2.



Q76) Explain sealed modifiers?

Ans) There modifiers prevents from inheritance and make them concrete.

Q77) Explain the difference between new and override?

Ans) Override allows child classes to redefine behaviour. and the behaviour is redefined using 'new'.

Q79) Repeat.

Q80) Define Manipulators?

Ans) In C++, manipulators are simply an instruction used to format the output data in various ways.

Q81) Can you give some examples of token in JAVA?

Ans) Tokens are smallest elements of a language

A JAVA program is written using tokens, white spaces and syntax of the language.

The JAVA has following tokens

i) Keywords ii) Identifiers iii) Literal

iv) Operator v) Seperator.

A compiler ignores white space unless written inside String.

Q82) Explain structured programming and its disadvantages?

Ans) It is a programming approach in which program is is made in single structure. It means the code will run instruction by instruction one after another.

It doesnot support the possibility of jumping from one instruction to another.

Eg C, C++, JAVA, .... supports structured programming.

Assembly languages like Microprocessor 8085 etc donot get executed in structured manner.

Disadvantages :→

i) It is machine independent. So, takes time to convert into machine code.

ii) Usually development in this approach takes longer time.

Q83) When to use interface over abstract class?

Ans) When we require that child class inherits the properties from more than one [multiple inheritance], at that time we prefer interface over abstract class because JAVA doesnot support multiple inheritance.

Q84) Explain private constructor? Where will you use it?

Ans) When a class has only ~~one or more~~ private constructors (one or more) and no public constructors, then we cannot instantiate it i,e cannot create object of this class and also it cannot be inherited by other classes.

Private constructor is used when the class contains static members only.

But we can create object within the same class.

Q85) Can you override private virtual methods?

Ans) No.

We cannot override private as well as static methods.

Q86) Can you allow class to be inherited, but prevent from being overriden?

Ans) Using 'sealed' keyword in C++, we can do that.

Q87) Why can't we specify access modifiers for methods inside interface?

Ans) Upto JAVA 7, we could have only public, abstract as modifiers for interface methods.

From JAVA 8 onwards, interface allows default and static methods.

~~Q88) Can we use static members~~

Q88) Can static methods use non-static methods? Give reasons

Ans) Directly we cannot.

But with object and (.) operator, we can.

Q88) Can static members/functions be used inside non-static methods? Give reason.

Ans) Yes we can.

Q89) Define different ways of method Overloading?

Ans) i) Number of parameters in two methods are different.

ii) Data types of parameters " " " " .

iii) The order of the parameters " " " " " .

Q90) Can we have an abstract class without having any abstract methods?

Ans) Yes we can declare an abstract class with no abstract methods in JAVA.

An abstract class can have abstract as well as non abstract methods.

Q91) Explain the default access modifier of a class?

Ans) When a class is having default access modifier, that means the class can be accessed by other classes within the same package but cannot be accessed by classes outside the same package.

Q92) Can function overriding be explained in same class?

Ans) No.

There must be a parent-child relationship.

Q93) Does function overloading depends on return type?

Ans) No.

Overloading comes under compile-time polymorphism. The compiler only checks method signatures and not return types