CHAPTER 7: CONCEPT OF INHERITANCE

Inheritance: It is the capability of one class to derive properties from another class.

Need for Inheritance:

1. It ensures closeness with real-world models.
2. Reusability
3. Transitive nature of inheritance

Subclass & Super Class: The class being inherited is called super class or base class and the inheriting class is called sub class or inherited class. Thus subclass derives some features (data members and methods) from its super class.

Forms of Inheritance

- **Single Inheritance:** A Subclass inherits from only one base class.
- **Multiple Inheritance:** A subclass inherits from multiple base classes **(not supported by Java)**.
- **Hierarchical Inheritance:** Many subclasses inherit from a single base class.
- **Multilevel Inheritance:** A subclass inherits from a class that itself inherits from another class. This shows transitive nature of inheritance.
- **Hybrid Inheritance:** Here a sub class inherits from multiple base classes and all of its base classes inherit from a single base class.

Defining Derived class:

Class <sub class name> extends <super class name> {
: // members of sub class}

Function Overloading: A function name having several definitions in the same scope that are differentiable by the number or types of their arguments (i.e. same name but different signature), is said to be an overloaded function. Functions with same name and same signature but different return type are not allowed. Functions with same name and signature are treated as re-declaration of first.

Need for Function Overloading: To cope with the changing behavior in different situations.

Declaration and Definition:

```
double a =0.0, b = 5.4, c = 8.9;
int d = 0, e = 5, f = 8;
a = sum(b,c);
d = sum(e,f);
System.out.println(a + "," + d);
int sum( int a, int b){                    //func 1
return (a+b); }
double sum( double x, double y){           //func 2
return (x+y);   }
```

Example of Inheritance and constructors

```
package a;
class person {
int j = 4;
private int i  = 5;
```

```
protected String name;
public String address;
person (String name, String address) {
this.name = name ;
this.address = address ; }
}
class student extends person {   //can't access i
int rolno ; int j = 1;
student (String name, String address, int rolno){
super(name, address);
this. rolno  = rolno ; }
public void display () {System.out.println(j + ", " + super.j);} }
// j of class person is hidden

class employee extends person{
String dept;
employee (String name, String address, String dept) {
super(name, address);
this.dept = dept; }}

class professor extends employee{
String inst;
professor(String name, String address, String dept, String inst) {
super(name, address, dept);
this. institute = inst; }}
class p  { }     // can't access i,

package b;
class q  { }
```

Sample Questions:

1. What is inheritance? Discuss its various forms.
2. Define base class and derived class. How are these related?
3. How does the visibility mode control the access of members in the derived class? Explain with examples.