CHAPTER 5: INTRODUCING CLASSES AND OBJECTS

Object Oriented Programming: The object oriented programming style emphasizes upon classes and objects. It incorporates the features of Data Abstraction, Encapsulation, inheritance and Polymorphism. Java is a pure object oriented language.
Class is a blue print that represents a set of similar objects e.g. vehicle. Class is User Defined/Reference Data Types.
Polymorphism*:* Polymorphism is the ability for a message or data to be processed in more than one form. It is the ability to behave differently in different circumstances.
Object is an entity with unique identity, characteristics and behaviour.
Method is the action defined in the class which can be carried out on the data.
Purpose of using method:
(i) Cope with complexity (ii) Hide details (iii) Reuse.
Definition:
[access specifier] [modifier] return type method-name (parameter list)
{ body of the method}

- Access specifiers can be private, protected or public. Default is friendly.
- Modifiers can be final, static etc.
- Return type may be of any data type or void.
- Parameter list is a comma-separated list of variables also known as arguments.

Access Specifiers
It controls access to members of a class. It may be

- **private**: Can't be accessed from outside the class.
- **protected**: Public to subclasses in any package and classes of same package but private to all other class.
- **public** : Directly accessible from all other classes.
- **default access** : If no specifier is used then the class has default access which is friendly or package access. Members with package access are not available to the classes or subclasses of other package.

Types of member in a class:
➢ **Instance member**: Defined without static keyword. Instance variables and instance methods are collectively called instance members. Each object created from such class will have its own copy of instance member. It is called as <object name>.method().
➢ **Static member**: It is declared using static keyword. It belongs to class as a whole and not to a particular object. It is called as <class name>.method().

```
float x = 25;
float z = Math.sqrt(x);          // class method.
String s = "KVS";
String t = s.substring(0, 4);    //instance method as it is called through object s
```

Creating objects:

```
city metro1, metro2;
metro1 = new city();
Or city metro1 = new city();
```

Sample Questions:

1. Define Object Oriented Programming (OOP).
2. Define a class with respect to OOP.
3. Define data encapsulation with reference to OOP.