Concept of OOPS:

- **Class:** When we create a class we use class keyword. Class is a collection of objects. Class is an entity which contains attributes and methods. Classes are variables. Classes are always public. We take a example of class:
  Class Employee{
  }
- Object: Object is an identity which contains a state and behaviour. We take an example of dog.
- Identity: The name of the dog is the identity of the dog.
- Stat: The age, colour and breed stat of the dog.
- Behaviour: while the dog is eating or sleeping.
- Polymorphism: Polymorphism is derived from poly and morph. Polymorphism means many number of forms. It can be achieved in python by using two methods:
- Method Overloading: when we have two or more methods having the same name but different parameters. This concept is known as method overloading.

Method 1:

```
def product(a,b):
p=a*b
print(p)
product(2,4)
```

Method 2:

```
def product(a, b, c):
p=a*b*c
print(p)
product(3, 4, 5)
```

- Method Overriding: When we have two or more methods having the same name and same arguments. This concept is known as Method Overriding.

- 1st Method:

```
def product(a, b):
p=a*b
print(p)
product(2,4)
```

- 2nd Method:

```
def product(a, b)
p=a*b
print(p)
```

product(3,4)

- Inheritance: When a class inherits the members and properties of another class. This concept is known as Inheritance. There are different types of inheritance. It provides the reusability of code. There is no need to write the code again and again.
- Single Inheritance: When one derived inherits the members and properties of single super class.
- Multiple Inheritance: When a derived class inherits the members and properties of two super class.
- Multi level Inheritance: Features of derived class and base class inherited further to the new derived class.
- Let assume, We take an example of three derived classes:d1,d2,d3 When a derived class d2 inherits the members and properties of derived class d1 and derived class d3 inherits

the members and properties of derived class d1 and d2.

- Hybrid Inheritance: when two derived class inherits the members and properties of single super class.
- Hierarchical Inheritance: It is a combination of hybrid inheritance and multiple inheritance.
- Encapsulation: Encapsulation means binding the code and data together as a single unit. It encapsulates all the data member functions and variables. A python class is an example of encapsulation. Encapsulation is the concept of wrapping the data and code acting as a single unit. We take a example of encapsulation.
- Class Base1
  def_init_(self):
  self.p="javatutorial"
  self.__q="javatutorial"
- #derived class
  Class deived1(Base1)
  def_init_(self):

- #base class
  ```
  Base1_init_(self)
  print("we will call only the private member of the base class")
  print(self.__q)
  ```
- #derived class
  ```
  obj_1=base1()
  print(obj_1.p)
  ```
- Data Abstraction: Data Abstraction providing only the required details and hiding the implementation from this world. It can be achieved in python by using interfaces and abstract classes. We take a example of a car like users does not know the implementation of car. users can only use the features of car.
-