**Polymorphism in Java**

The process of representing one form in multiple forms is known as **Polymorphism**.

Polymorphism is derived from 2 greek words: **poly** and **morphs**. The word "poly" means many and "morphs" means forms. So polymorphism means many forms.

Polymorphism is not a programming concept but it is one of the principal of OOPs. For many objects oriented programming language polymorphism principle is common but whose implementations are varying from one objects oriented programming language to another object oriented programming language.

**Real life example of polymorphism**

Suppose if you are in class room that time you behave like a student, when you are in market at that time you behave like a customer, when you at your home at that time you behave like a son or daughter, Here one person present in different-different behaviors.

**How to achieve Polymorphism in Java ?**

In java programming the Polymorphism principal is implemented with method overriding concept of java.

Polymorphism principal is divided into two sub principal they are:

* Static or Compile time polymorphism
* Dynamic or Runtime polymorphism

**Note:**Java programming does not support static polymorphism because of its limitations and java always supports dynamic polymorphism.

**Let us consider the following diagram**

Here original form or original method always resides in base class and multiple forms represents overridden method which resides in derived classes.

In the above diagram the sum method which is present in BC class is called original form and the sum() method which are present in DC1 and DC2 are called overridden form hence Sum() method is originally available in only one form and it is further implemented in multiple forms. Hence Sum() method is one of the polymorphism method.

**Example of runtime polymorphism.**

In below example we create two class Person an Employee, Employee class extends Person class feature and override walk() method. We are calling the walk() method by the reference variable of Parent class. Since it refers to the subclass object and subclass method overrides the Parent class method, subclass method is invoked at runtime. Here method invocation is determined by the JVM not compiler, So it is known as runtime polymorphism.

**Example of Polymorphism in Java**

**class** Person

{

**void** walk()

{

System.**out**.println("Can Run....");

}

}

**class** Employee **extends** Person

{

**void** walk()

{

System.**out**.println("Running Fast...");

}

**public** **static** **void** main(String arg[])

{

Person p=**new** Employee(); //upcasting

p.walk();

}

}

**Output**

Running fast...

**Dynamic Binding**

**Dynamic binding** always says create an object of base class but do not create the object of derived classes. Dynamic binding principal is always used for executing polymorphic applications.

The process of binding appropriate versions (overridden method) of derived classes which are inherited from base class with base class object is known as dynamic binding.

Advantages of dynamic binding along with polymorphism with method overriding are.

* Less memory space
* Less execution time
* More performance

**Static polymorphism**

The process of binding the overloaded method within object at compile time is known as **Static polymorphism** due to static polymorphism utilization of resources (main memory space) is poor because for each and every overloaded method a memory space is created at compile time when it binds with an object. In C++ environment the above problem can be solve by using dynamic polymorphism by implementing with virtual and pure virtual function so most of the C++ developer in real worlds follows only dynamic polymorphism.

**Dynamic polymorphism**

In dynamic polymorphism method of the program binds with an object at runtime the advantage of dynamic polymorphism is allocating the memory space for the method (either for overloaded method or for override method) at run time.

**Conclusion**

The advantage of dynamic polymorphism is effective utilization of the resources, so java always use dynamic polymorphism. Java does not support static polymorphism because of its limitation.