Polymorphism:

Date: 28th Mar 2024

Different types of JVM memories:
1. Heap area--> non-static method declaration
2. Static pool area--> static method declaration
3. method area --> static & non-static method definition
4. stack --> main()--> method execution flow

Polymorphism:

It is one of the OOPs principles where one object showing different behaviour at different stages of life cycle.

Polymorphism is an Latin word where poly stand for many & morphism stands for forms.

In java Polymorphism is classified into 2 types:

1. Compile time Polymorphism
2. Runtime Polymorphism

1. Compile time Polymorphism:

In Compile time Polymorphism method declaration is going to get binded to its definition at compilation time, based on argument/input/parameter is known as compile time Polymorphism.

As binding takes during compilation time only, so it is also known as early binding.

//once binding is done, again rebinding can't be done, so it is called static binding.

Method overloading is an example of compile time Polymorphism.

2. Runtime Polymorphism:

In Runtime Polymorphism method declaration is going to get binded to its definition at Runtime/execution time, based on object creation is known as runtime Polymorphism.

As binding takes during Runtime/execution time, so it is also known as late binding.

//once binding is done, again rebinding can be done, so it is called dynamic binding.

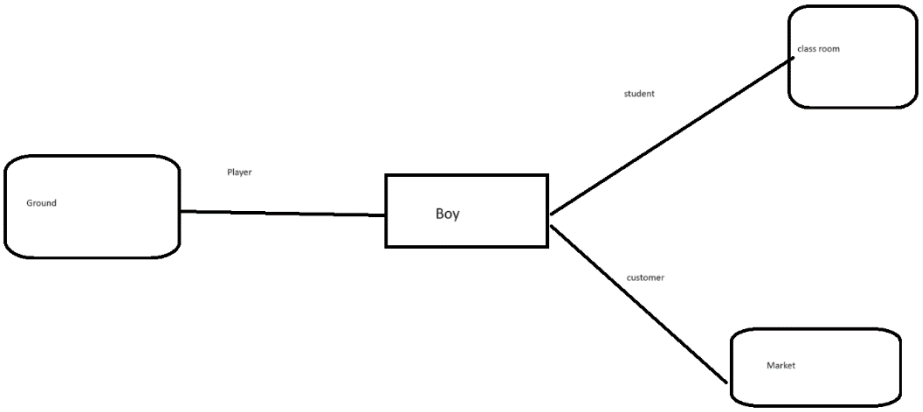
Method overriding is an example of Runtime Polymorphism.

Method overloading:

Declaring multiple method with same method name but with different argument/parameter/inputs in a same class is called method overloading

Method overriding:

Acquiring super class method into sub class with the help of extends keyword & changing implementation/definition according to subclass specification is called method overriding



```java
package PolyMorphism;
public class Sample1
{
	//method overloading

	//add method with 2 int parameter
	public void add(int a, int b)
	{
		System.out.println(a+b);
	}

	public void add(int a, int b, int c)
	{
		System.out.println(a+b+c);
	}
}
```

```java
package PolyMorphism;
public class TestOverloading
{
	public static void main(String[] args)
	{
		Sample1 s1=new Sample1();
		s1.add(10, 20, 30);
	}
}
```