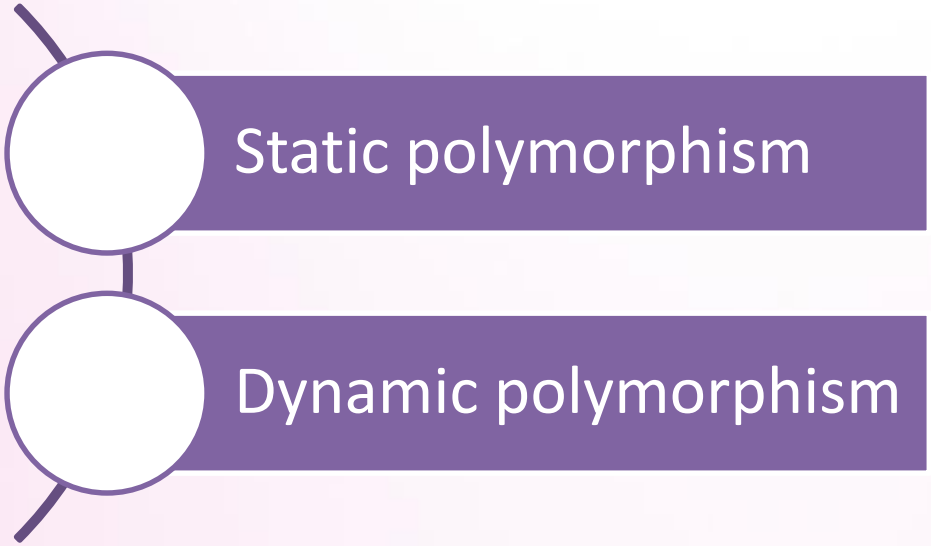
Implementing Polymorphism

- Polymorphism is an OOP feature that enables an entity to exist in multiple forms.
- In Java, polymorphism has the following two types:



Static Polymorphism

- In case of static polymorphism, an entity, such as a method, can exist in multiple forms.
- In static polymorphism, one or more methods can exist with the same name but with a different argument list.
- In the static polymorphism, when exhibited by methods, is known as method overloading.
- For example, a method named calculate can be overloaded by using the following code snippet:

```
calculate( int x, int y)
{
/* Some code to be added */
}
```

Static Polymorphism (Contd.)

```
calculate (float x, int y, int z)
{
/* Some code to be added */
}

calculate (int x, float y)
{
/* Some code to be added */
}

calculate (float y, int x)
{
/* Some code to be added */
}
```

Static Polymorphism (Contd.)

- While implementing method overloading, it is important to consider the following points about overloaded methods:
  - They differ in the type and/or number of their arguments.
  - They differ in the sequence of their parameters.
  - They differ in the data types of their parameters.