Tính đa hình (Polymorphism)

Polymorphism là từ được cấu thành từ 2 từ Hy Lạp là poly và morhps, nếu hiểu theo nghĩa tiếng anh thì từ này có nghĩa là **one thing in many form**, nghe thì có vẻ khó hiểu nhưng nếu nhìn vào các ví dụ bên dưới thì bạn sẽ hình dung ra được thế nào là **one thing in many form**. Tính đa hình được thể hiện trong code dưới 3 hình thức: nạp chồng phương thức, ghi đè phương thức và đa hình thông qua các đối tượng đa hình (polymorphic objects).

Đa hình với nạp chồng phương thức

Ví dụ: phương thức cộng sẽ có các "forms" là cộng 2 số nguyên, cộng 2 số thực, cộng 3 số nguyên, v/v. Có thể thấy cùng là phương thức cộng nhưng lại có nhiều "forms" (**one thing in many form**) nên đây chính là biểu hiện của tính đa hình. Ví dụ về đa hình với nạp chồng phương thức:

```
package OOP;

class Calculator {
	public int add(int a, int b) {
		return a + b;
	}

	public double add(double a, double b) {
		return a + b;
	}

	public int add(int a, int b, int c) {
		return a + b + c;
	}
}

public class Entry {
	public static void main(String[] args) {
		Calculator calculator = new Calculator();
		System.out.println(calculator.add(1, 2));
		System.out.println(calculator.add(3.3, 4.2));
		System.out.println(calculator.add(1, 2, 3));
	}
```

```
}
```

Kết quả khi chạy chương trình:

```
3
7.5
6
```

Đa hình với ghi đè phương thức

Ví dụ phương thức `getSalary()` dùng để tính lương sẽ có các "forms" là tính lương cho quản lý, tính lương cho nhân viên:

```
class Employee {
        private String name;
        private int salary;

        public Employee(String name, int salary) {
                super();
                this.name = name;
                this.salary = salary;
        }

        public String getName() {
                return name;
        }

        public void setName(String name) {
                this.name = name;
        }

        public int getSalary() {
                return salary;
        }
```

```
        public void setSalary(int salary) {
                this.salary = salary;
        }

        public void display() {
                System.out.println("Name: " + getName());
                System.out.println("Salary: " + getSalary());
        }
}

class Manager extends Employee {
        private int bonus;

        public Manager(String name, int salary, int bonus) {
                super(name, salary);
                this.bonus = bonus;
        }

        public int getBonus() {
                return bonus;
        }

        public void setBonus(int bonus) {
                this.bonus = bonus;
        }

        @Override
        public int getSalary() {
                return super.getSalary() + bonus;
        }
}
```

Đều là tính lương nhưng với mỗi đối tượng lại có một cách tính khác nhau, đó chính là tính đa hình.

Đa hình thông qua các đối tượng đa hình (polymorphic objects)

Như đã học ở bài up-casting, biến thuộc lớp cha có thể tham chiếu tới đối tượng của các lớp con, vậy biến thuộc lớp cha cũng có nhiều "forms" nên đây cũng là đa hình. Ví dụ:

```
package OOP;

class Animal {
    public void sound() {
        System.out.println("some sound");
    }
}

class Dog extends Animal {
    public void sound() {
        System.out.println("bow wow");
    }
}

class Cat extends Animal {
    public void sound() {
        System.out.println("meow meow");
    }
}

class Duck extends Animal {
    public void sound() {
        System.out.println("quack quack");
    }
}

public class Entry {
        public static void main(String[] args) {
```

```
                Animal animal = new Animal();
                animal.sound();
                animal = new Dog();
                animal.sound();
                animal = new Duck();
                animal.sound();
                animal = new Cat();
                animal.sound();
        }
}
```

Kết quả khi chạy chương trình:

```
some sound
bow wow
quack quack
meow meow
```

Có thể thấy trong ví dụ trên, biến `animal` có thể đại diện cho các đối tượng của các lớp `Animal, Cat, Dog`. Đây cũng là một ví dụ điển hình của tính đa hình.