**PHƯƠNG THỨC**

1. **TRUYỀN THAM SỐ**

3 thuật ngữ truyền tham số

* **Pass by value:** truyền tham trị
* Là cách truyền tham số khi kiểu dữ liệu là kiểu cơ bản.
* Pass by variable: truyền tham biến
* **Pass by reference:** truyền tham chiếu
* Là cách truyền tham số khi kiểu dữ liệu là Object

1. **Overloading**

* Nạp chồng phương thức (các phương thức trong cùng một class cùng tên, cùng kiểu, khác tham số)
* Giúp tăng linh hoạt cho phương thức khi được gọi

1. **Phạm vị của biến**

* Không có biến gọi là toàn cục
* Nếu “biến” khai báo trước tất cả các phương thức của class thì ta gọi là “thuộc tính”.
* Nếu biến khai báo bên trong hàm thì là biến địa phương. Phạm vi của biến địa phương được xác định bên trong khối lệnh (block {}) mà nó được khai báo.

1. **STATIC**

|  |  |  |
| --- | --- | --- |
|  | **static** | **non-static** |
| **Cách sử dụng** | Gọi bằng class | Gọi bằng object |
| **Phạm vi** | Phương thức chung của cả class | Phương thức riêng của đối tượng |
| **Tài nguyên sử dụng để thực thi phương thức** | Không cần tài nguyên riêng, tất cả mọi thứ mà phương thức cần đều ở trong tham số. | Cần tài nguyên riêng (dữ liệu của đối tượng) để thực thi. |
| **Phương pháp** | Hàm chức năng  *Hướng cấu trúc cơ bản* | Hành vi của đối tượng  *Hướng đối tượng* |
| **Khi nào sử dụng?** | Thực hiện các chức năng xử lý trực tiếp trên tham số, không liên quan đến dữ liệu của đối tượng. | Khi cần sử dụng không gian nhớ của đối tượng và thực hiện hành vi trên nó. |
| **Nên dùng cái nào?** | Cái nào cũng đúng!!!  Bạn nghĩ kiểu static hay là non-static  Point p1 = new Point(1,2);  Point p2 = new Point(2,3);  Ví dụ: cần viết hàm tính khoảng cách 2 điểm:  static: Point.distance(p1,p2);  non-static: p1.distance(p2)  *không nên tư duy kiểu: p1.distance(p1,p2);* | |