# Polymorphism

* refactor switch statements to simple one liners

# Encapsulation

* reduce complexity,
* combine related data together

# Inheritance

* Increase reusability

# Abstraction

Reduces the breaks due to changes by hiding inner details

# Objects

## Factory Function

* Create object through a function with return statement

## Constructor

* Uses new keyword, first letter of constructor shall be Capital,
* assign the properties uses this keyword to assign properties,
* and no return is required

## Constructor Property

* Every object has a constructor property; this property refers to the function this is used to create the object
* **Functions itself are objects in JavaScript (and their constructor is Function method present in JavaScript engine)**

## Primitive vs Reference Types

* Value type copy by values
* Reference types are copied by reference

|  |  |
| --- | --- |
| Primitive Types | Reference Types |
| Number | Objects |
| String | Functions |
| Boolean | Array |
| Undefined |  |
| null |  |
| Symbol (Available in ES6) |  |

## Adding Removing Properties

* Add new property with dot or bracket notation
* Remove it with delete property (delete circle.radius)

## Looping Through Properties

* We can use for in loop to enumerate the properties