1）1，2，3是构造函数三要素：

1.构造函数的首字母必须大写，用于来区分普通函数

2.构造函数内部使用的this对象，来指向即将要生成的实例对象

3.使用new运算符调用

2）构造函数有返回值，没有return语句也有返回值的，返回this，即实例对象，是object类型。

如果写return语句，且返回值为基本数据类型，构造函数会忽略return的值，依然返回this对象。有return语句，且返回值为引用数据类型，构造函数会返回return的值。

3）prototype：1.只有函数才有prototype属性

2.当创建一个函数，系统会给函数生成prototype属性，这个属性的值是一个对象，而prototype中有constructor属性，指向创建的函数（指向其空间）即fun===fun.prototype.constructor

3.可以通过prototype属性向对象添加属性和方法（与constructor在一起，在同一块）\_\_proto\_\_属性：每个实例对象都有一个属性\_\_proto\_\_指向它的构造函数的原型对象（prototype），所以fun.prototype===fun1.\_\_proto\_\_

//fun1是创建的实例对象。

//实例对象可以访问这个原型对象（prototype）中的属性，就像本身拥有该属性一样。

function Person(){

}

//这样用{}来添加属性，就是相当于重新定义属性方法，令Person.prototype只有{}中出现的属性方法。

Person.prototype={

fun1:function(){},

fun2:function(){}

}

var boy=new Person();

console.log(boy.\_\_proto\_\_===Person.prototype);//true

console.log(Person===Person.prototype.constructor);//false

4）静态属性：给函数添加的属性

原型属性添加在函数prototype上的属性

实例属性：添加在实例对象上的属性。

实例对象可以访问本身独享上的是属性，也可以访问原型属性，但不能访问构造函数上的静态属性。

5）new运算符作用：1.创建一个空的简单的JS对象（即{}）

2.该对象会被执行[[prototype]]连接

3.如果构造函数没有返回引用类型，则返回this指向的实例对象

4.将该对象作为this的上下文，并执行构造函数中的代码。