instanceof 原理是什么，用代码实现

`instanceof` 左侧是实例，右侧是类。如 `[] instanceof Array`，如果左侧操作数是右侧操作数的实例，则返回 true，否则返回 false。

原理是：

- 查找左侧操作数的隐士原型 **proto** 是否和后侧操作数的原型相等，如果相等就返回 true。否则，就往下查找隐士原型，再来比较是否和右侧操作数的原型是否相等，一直这样找到 null，如果还不相等就证明左侧操作数不是右侧操作数的实例，返回 false。

```
function myInstanceof(instance: any; Foo: any) {
    if (instance === null || instance === undefined) return false;

    const type = typeof instance; // 获取实例类型，判断是不是非引用类型
    if (type !== 'object' && type !== 'function') {
        return false;  // 左侧操作数如果是值类型(number, string, boolean)，就不是右侧操作数的实例
    }

    let temInstance = instance;
    while(temInstance) {
        if (temInstance.__proto__ === Foo.prototype) {
            return true;
        } else {
            temInstance = temInstance.__proto__;  // 顺着原型链，往上找
        }
    }
}
```

每个 class 都有显示原型 prototype，每个实例都有隐士原型 proto，且实例的 proto 指向 class 的显示原型