//第三天小作业

// 1.bind方法使用

// 1)绑定函数(即创建一个函数，使得这个函数不管怎么调用都有相同的this值,eg:

this.num = 9;

const mymodule = {

num: 81,

getNum() { return this.num; }

};

const a = mymodule.getNum(); // 81

const getNum = mymodule.getNum; // 9, 在这个例子中，this指向全局对象

const boundGetNum = getNum.bind(mymodule);//81,this绑定到mymodule的函数

// 2)偏函数,使用bind()我们设定函数的预定义参数，然后调用的时候传入其他参数即可

function list() {

return Array.prototype.slice.call(arguments);

}

const list1 = list(1, 2, 3); // [1, 2, 3]

const leadingThirtysevenList = list.bind(undefined, 37);// 预定义参数37

const list2 = leadingThirtysevenList(); // [37]

const list3 = leadingThirtysevenList(1, 2, 3); // [37, 1, 2, 3]

2.require含义

当 Node 遇到 require(X) 时，按下面的顺序处理。

（1）如果 X 是内置模块（比如 require('http'）)

　　a. 返回该模块。

　　b. 不再继续执行。

（2）如果 X 以 "./" 或者 "/" 或者 "../" 开头

　　a. 根据 X 所在的父模块，确定 X 的绝对路径。

　　b. 将 X 当成文件，依次查找下面文件，只要其中有一个存在，就返回该文件，不再继续执行。

X

X.js

X.json

X.node

　　c. 将 X 当成目录，依次查找下面文件，只要其中有一个存在，就返回该文件，不再继续执行。

X/package.json（main字段）

X/index.js

X/index.json

X/index.node

（3）如果 X 不带路径

　　a. 根据 X 所在的父模块，确定 X 可能的安装目录。

　　b. 依次在每个目录中，将 X 当成文件名或目录名加载。

（4） 抛出 "not found"