let 与 var 的区别

1： 不存在变量提升,和存在暂时性死区，如果使用`let`，声明的变量仅在块级作用域内有效

所以在声明之前，都属于`x`的“死区”，只要用到该变量就会报错。因此，`typeof`运行时就会抛出一个`ReferenceError`。

2: 只要块级作用域内存在`let`命令，它所声明的变量就“绑定”（binding）这个区域，不再受外部的影响。

3:`let`不允许在相同作用域内，重复声明同一个变量。

ES6 的块级作用域

每一层都是一个单独的作用域。

第四层作用域无法读取第五层作用域的内部变量。

内层作用域可以定义外层作用域的同名变量。

块级作用域的出现，实际上使得获得广泛应用的匿名立即执行函数表达式（匿名 IIFE）不再必要了

const

改变常量的值会报错。

`const`声明的变量不得改变值，这意味着，`const`一旦声明变量，就必须立即初始化，不能留到以后赋值。

`const`命令声明的常量也是不提升，同样存在暂时性死区，只能在声明的位置后面使用。

`const`的作用域与`let`命令相同：只在声明所在的块级作用域内有效。

块级作用域与函数声明

ES5 规定，函数只能在顶层作用域和函数作用域之中声明，不能在块级作用域声明。

ES6 引入了块级作用域，明确允许在块级作用域之中声明函数。ES6 规定，块级作用域之中，函数声明语句的行为类似于`let`，在块级作用域之外不可引用。