1.栈内存（stack）【存储的值大小固定，由系统自动分配内存空间，空间小，运行效率高】堆内存（heap）【存储的值大小不定，动态调整，代码由程序员调控】

2.基本数据类型的变量存放在栈区（基本类型的值不可变）

3.引用数据类型的值是同时保存在栈内存和堆内存中的对象，堆内存中存真正数据，栈内存中存放地址

4.变量操作：

复制变量：

基本类型复制：拷贝栈内存中内容，是变量所关联的值，所以拷贝的是值。相互独立互不影响

引用类型复制：拷贝栈内存的内容，是复制地址，所以指向的是同一地址指向的堆内存，是同一块堆内存空间

比较变量：

//3.比较变量

Var a= 'hello';

a.toUpperCase();

console.log(a);

//引用他类型是判断所指向的内存空间（地址）是否相同

Var x = {

k:10,

h: 20

};//x相关联的空间1，存的地址01指向的堆内存中的空间1

Var y ={

k: 10,

h: 20

};//y相关联的空间2，存的地址02指向的堆内存中的空间2

console.log(x === y); //所指向的内存空间（地址）不相同

X = y; //x拷贝y的栈内存的所存的地址，令x和y指向的堆内存的空间一样

console.log(x===y);

参数传递：

隐式转换：非纯数字字符串转Number时时NAN（无意义）。其他类型转Number用1\*【NAN不等于自身】。undefined转Number时是NAN。声明但未赋值变量是undefined类型，比较大小是转换为Number类型比较。

5.空字符串，NAN,undefined，false，null，0是假值，除此外都是真值

6.存取【字符串，数字或布尔值】得属性时创建的【临时对象】（临时对象指一旦属性引用结束，包装对象就会销毁，即堆内存中存储内容销毁）

7.var str="hello";

var stt=str.toUpperCase();//stt是基本类型，创建和String对应的包装对象，临时的。

//stt=new String(str).toUpperCase();这样创建stt时个引用类型，是个对象

str.name="STR";//创建包装对象，堆内存中开辟一块空间，存STR，但当这行代码运行完毕，堆内存中内容销毁。

console.log(str.name);//又重新创建了一个包装对象，str.name没有赋值，结果为undefined

8.包装类型和引用类型的区别主要是：对象的生存周期