1. 全局搜索关键字VSCode

Edit->find in files

1. 正则表达式：描述符合某些复杂规则的字符串的代码
2. JIT：运行时编译

支持动态链接，编译占用运行时资源

AOT：运行前编译

避免运行时的编译性能消耗和内存消耗，加快程序启动

使程序安装的时间增加，牺牲Java的一致性

1. &按位与

&&短路运算

1. 自动装箱

Integer a = 3; //将3自动装箱成Integer类型

1. String s = “constant”;//s是常量池中”constant”的引用

String s2 = new String(“constant”);//s2无法在编译器确定，是运行时创建的新对象” constant”的引用

s2.intern();//查找常量池，没有则新增一个

1. Array可以包含基本类型和对象类型，大小固定

ArrayList只能包含对象类型，大小动态变化

8． 值传递：对基本数据类型而言，传递的是变量的一个副本，改变副本不影响原变量

引用传递：对于对象型变量而言，传递的是对象地址的一个副本

9. 4.0-3.6=0.4000001？二进制的小数无法精确地表达十进制小数，计算机在计算十进制小数的过程中要先转换为二进制进行运算，这个过程中出现了误差

10. 十进制的数在内存中以二进制补码的形式存储，最高位是符号位

正数的反码是其本身

负数的反码是在其原码的基础上, 符号位不变，其余各个位取反.

正数的补码就是其本身

负数的补码是在其原码的基础上, 符号位不变, 其余各位取反, 最后+1. (即在反码的基础上+1)

11. Lambda表达式：允许把函数作为一个方法的参数

// 5. 接受一个 string 对象,并在控制台打印,不返回任何值(看起来像是返回void)

(String s) -> System.out.print(s)

12. hashcode的主要作用是用于查找的快捷性（哈希算法，将数据依特定算法直接指定到一个地址上，当集合要添加新元素时，先调用这个元素的hashcode方法，就能定位到它的物理位置）

1）如果两个对象相同，equals返回true

2）两个对象的hashcode 相同，并不一定表示两个对象就相同，只能说明这两个对象在一个散列存储结构中

、

13. Map四个实现类

①HashMap:根据键的hashcode值存储数据，根据键可以直接获取它的值，遍历时，取得数据的顺序是随机的，遍历顺序与容量有关,不支持线程同步，最多只允许一条记录的键为NULL，允许多条记录的值为NULL

②HashTable：键和值不能为空,支持线程同步（即任一时刻只能有一个线程写入HashTable）

③LinkedHashMap

④TreeMap：能把它保存的值根据键排序

14. 保证任一时刻最多只有一个线程执行该段代码

Synchronized是关键字：发生异常时，会自动释放线程占有的锁，因此不会导致死锁的发生

Lock是接口：发生异常时，得通过在finally块中用unLock()释放锁

15. volatile关键**字：被volatile修饰的变量**保证每个线程能够获取该变量的最新值

16．重载（Overloading）发生在同一个类里面多个方法的方法名相同但是参数不同的情况

无法以返回型别作为重载函数的区分标准

覆盖（overriding）子类重新定义父类的方法，可重新定义访问限制

17.

抽象：从多个事物中将共性的，本质的内容抽象出来，忽略一个主题中与当前目标无关的那些方面，以便更充分地注意与当前目标有关的方面

抽象类：可以包含抽象和非抽象方法

接口：特殊的抽象类，接口更强调功能

继承：新类继承原始类的特性

封装：把过程和数据包围起来成为一个对象，这些对象通过一个受保护的接口访问其他对象

多态

18. Comparator

19. 反射：允许允许中的Java程序获取自身的信息，并且可以操作类或对象的内部属性

所有类在加载后，JVM会为其在堆中创建一个Class<类名称>的对象，并且每个类只会有一个Class对象，这个类的所有对象都要通过Class<类名称>来进行实例化。

可通过Class.newInstance()方法创建Class对象对应类的实例（调用的类要有无参构造函数）

Class对象的getDeclaredMethods()方法返回类或接口声明的所有方法、

getMethods()方法返回某个类的所有公用（public）方法

getDeclaredConstructors()返回构造函数

通过method1.invoke()方法调用method1()

20. static静态修饰符

静态：被static修饰的成员在编译后所分配的内存会一直存在，知道程序退出内存才会释放这个空间

不依赖类特定的实例，被类的所有实例共享；static对象可以在它的任何对象创建之前访问，无须引用任何对象

static final 修饰的常量 ->全局常量、

修饰的容器类型变量->所装的值可变

static静态代码块在项目启动的时候就执行，自动执行

静态方法里只能直接调用同类中其他的静态成员，而不能直接访问类中的非静态成员，因为非静态方法和变量只有在创建完类的实例对象后才可使用

静态方法内部无法使用this,super，因为静态方法不属于某个实例对象

21. 上界<? extend Fruit> ，表示所有继承Fruit的子类（上界是fruit）

编译器支持向上转型

22. 泛型：“参数化”类型，将类型由原来的具体的类型参数化，在使用/调用时传入具体的类型  
23. StringBuffer比StringBuilder线程安全，多了个Synchronized修饰符

24. Object

-clone()//创建并返回此对象的一个副本

-equals()//比较两个对象是否相等

-finalize()//当垃圾回收器确定不存在对该对象的等多引用时，由对象的垃圾回收器调用此方法

-getClass()//返回一个对象的运行时类

-hashcode()//返回该对象的哈希码

-wait()//导致当前的线程等待

-wait(long timeout, int nanos) 导致当前的线程等待，直到其他线程调用此对象的 notify() 方法或 notifyAll() 方法，或者其他某个线程中断当前线程，或者已超过某个实际时间量

-notify()//唤醒在此对象监视器上等待的单个线程

-notify()//唤醒在此对象监视器上等待的所有线程

25. 类是对某一类事物的描述，是抽象的

对象是实实在在的个体，是类的一个实例

26. String在JDK中被声明为final类(final char[])，所以不可变

为什么HashMap中的键往往使用String：不可变，hashcode一致性，处理速度快，安全，多线程安全