面试真题

【腾讯】编译型语言和解释型语言的区别？

* 编译型语言C++

源代码被编译成机器码，生成可执行文件执行

适合高性能、低延迟的场景，比如操作系统

* 解释型语言Python

源代码被解释器逐行解释执行

适合快速开发、跨平台的场景，比如数据分析

* 编译和解释共存语言Java

源代码被编译成字节码，由JVM解释执行

【腾讯】动态数组的实现：ArrayList 和 Vector 的比较

* ArrayList：非线程安全、**扩容为**1.5**倍**
* Vector：线程安全、**扩容为**2**倍**

【腾讯】HashMap 的扩容条件是什么？

* Java8扩容**条件**：**已有元素的个数达到阈值**
* Java7扩容**条件**：**已有元素的个数达到阈值** && 发生哈希冲突
* Java7 HashMap扩容存在极端情况：  
  默认大小16，负载因子0.75，阈值12，最多存26个值还不触发扩容

前11个值不达阈值+冲突，存在同一位置

后15个值达到阈值+不冲突， 存剩下15个位置

面向简历

熟悉Java集合

* Collection接口：存储对象
  + Set接口：不许重复
    - HashSet
    - LinkedHashSet：基于链表和哈希表，维护插入顺序
    - TreeSet：基于红黑树，元素有序
  + Queue接口**：**
    - LinkedList：FIFO队列
    - PriorityQueue：优先级队列
  + List接口**：**
    - ArrayList：动态数组、查询快、扩容1.5倍
    - Vector：动态数组、线程安全开销大、扩容2倍
    - LinkedList：双向链表、插删快
* Map接口：存储键值对
  + HashMap
  + LinkedHashMap
  + TreeMap
  + HashTable：线程安全、不许null
  + ConcurrentHashMap：线程安全、、适合高并发、不许null

I/O流

字节流和字符流的区别

* 字节流：继承InputStream和OutputStream，以字节为单位，适用于文图音
* 字符流：继承Reader和Writer，以字符为单位，仅适用于文本

缓冲流和对象流

* 缓冲流BufferedXXX通过内存缓冲区提升IO效率
* 对象流ObjectXXX允许Java对象序列化保存或传输

New I/O与传统I/O的区别

NIO通过缓冲区和通道传输数据提升IO效率

NIO支持异步I/O和非阻塞I/O，支持高并发

I/O异常处理

* 使用try-catch-finally块进行异常处理
* Java 7引入try-with-resources，自动关闭流、避免资源泄漏

Java反射

Java反射：运行时根据类名调用类、字段、方法，实现依赖注入、动态代理

优点：灵活、常用于框架实现

缺点：开销大、不安全、可读性低

核心类：

1. Class类：通过forName来动态加载类
   * Class<?> clazz = Class.forName("com.example.MyClass");
2. Constructor类：通过getConstructor获取构造器，newInstance创建对象
   * Constructor<?> constructor = clazz.getConstructor(String.class, int.class);
   * Object obj = constructor.newInstance("John", 25);
3. Field类：通过getDeclaredField获取字段，setAccessible(true)访问私有字段
   * Field field = clazz.getDeclaredField("name");
   * field.setAccessible(true);
4. Method类：通过getDeclaredMethod获取方法，使用invoke调用
   * Method method = clazz.getDeclaredMethod("setName", String.class);
   * method.invoke(obj, "NewName");

HashMap及其扩容机制

负载因子

初始容量为 16、负载因子为 0.75（默认）

扩容阈值16\*0.75=12、当存入第13个元素时扩容16\*2=32

Rehashing

Rehashing扩容重新计算哈希值：

* Java7每个元素rehash后依次搬迁
* Java8优化后只需部分移动

基础：

1. 数组长度是2的次方，且扩容为2倍。
2. 位运算代替取模提高效率、保证hash均匀分布：hash & (数组长度-1)

举例：

旧长16即010000，16-1=15即001111

新长32即100000，32-1=31即011111

源码：

相当于rehashing时用旧长16即010000作为一个Mask看对应该位是否为1

源码if e.hash & oldCap==0（0接在loTail后面、1接在hiTail后面）

知识框架