practice makes perfect.

Java 基础

horaoen@gmail.com | https://github.com/horaoen

一、Java 和 C++主要区别有哪些？各有哪些优缺点？

Java 和 C++分别代表了两种类型的语言：

1. C++是编译型语言（首先将源代码编译生成机器语言，再由机器运行机器码），执行速度快、效率高；依赖编译器、跨平台性差些。
2. Java 是解释型语言（源代码不是直接翻译成机器语言，而是先翻译成中间代码，再由解释器对中间代码进行解释运行。），执行速度慢、效率低；依赖解释器、跨平台性好。

PS：也有人说 Java 是半编译、半解释型语言。Java 编译器(javac)先将 java 源程序编译成 Java 字节码(.class)，JVM 负责解释执行字节码文件。

二者更多的主要区别如下：

1. C++是平台相关的，Java 是平台无关的。
2. C++对所有的数字类型有标准的范围限制，但字节长度是跟具体实现相关的，同一个类型在不同操作系统可能长度不一样。Java 在所有平台上对所有的基本类型都有标准的范围限制和字节长度。
3. C++除了一些比较少见的情况之外和 C 语言兼容 。Java 没有对任何之前的语言向前兼容。但在语法上受 C/C++ 的影响很大
4. C++允许直接调用本地的系统库 。 Java 要通过 JNI 调用，或者 JNA
5. C++允许过程式程序设计和面向对象程序设计 。Java 必须使用面向对象的程序设计方式
6. C++支持指针，引用，传值调用 。Java 只有值传递。
7. C++需要显式的内存管理，但有第三方的框架可以提供垃圾搜集的支持。支持析构函数。 Java 是自动垃圾收集的。没有析构函数的概念。
8. C++支持多重继承，包括虚拟继承 。Java 只允许单继承，需要多继承的情况要使用接口。

Java 与 C 的参数方法有什么区别?

C 语言是通过指针的引用传递

```
void swap(int *i, int *j) {
    int temp = *i;
    *i = *j;
    *j = temp;
}
```

Java 会拷贝当前栈中的值传递过去

```
public class Test {
    private int num;
    public Test(int num) {
        this.num = num;
    }
```

practice makes perfect.

```
    @Override
    public String toString() {
        return "Test{" +
                "num=" + num +
                '}';
    }

    public static void main(String[] args) {
        int i = 10;
        int j = 20;
        swap(i,j);
        //10
        System.out.println(i);
        //20
        System.out.println(j);
        Test ii = new Test(i);
        Test jj = new Test(j);

        swapInstance(ii, jj);
        //Test{num=10}
        System.out.println(ii);
        //Test{num=20}
        System.out.println(jj);
    }

    private static void swap(int i, int j) {
        int temp = i;
        i = j;
        j = temp;
    }

    private static void swapInstance(Test i, Test j) {
        Test temp = i;
        i = j;
        j = temp;
    }
}
```

编程语言中需要进行方法间的参数传递, 这个传递的策略叫做求值策略。在程序设计中, 求值策略有很多种，比较常见的就是值传递和引用传递。还有一种值传递的特例——共享对象传递。

值传递和引用传递最大的区别是传递的过程中有没有复制出一个副本来，如果是传递副本，那就是值传递，否则就是引用传递。

Java 对象的传递，是通过复制的方式把引用关系传递了，因为有复制的过程，所以是值传递，只不过对于 Java 对象的传递，传递的内容是对象的引用。