编程语言

一、C和C++

C语言面向**过程**，重点在于**算法和数据结构。**C程序的设计首先考虑的是如何通过一个**过程**，对输入（或环境条件）进行**运算处理**得到**输出**（或实现过程（事务）控 制）。

C++语言是面向**对象**语言，首先考虑的是如何构造一个**对象模型**，让这个模型能够**契合**与之对应的 **问题域**，这样就可以通过**获取对象的状态信息**得到**输出或实现过程**（事务）控制。 它在C的基础上添加了面向对象、模板等现在程序设计语言的特性。拓展了面向对象设计的内容，如类、继承、虚函数、模板和容器类等等，使之更加符合现代程序设计的需要。

C++是C语言的继承，它既可以进行**C语言的过程化**程序设计，又可以进行以**抽象数据类型**为特点的**基于对象**的程序设计（泛型编程），还可以进行以**继承和多态**为特点的**面向对象**的程序设计（面向对象编程），支持类、封装、继承、多态等特性。

面向对象编程：面向对象是一种对**现实世界理解和抽象**的方法、思想，通过将**需求要素**转化为**对象**进行问题处理的一种思想。

类：类定义了事物的**属性**和它可以做到的（它的**行为**）。一个类的**方法和属性**被称为"**成员**"。类是在**对象**之上的抽象，**对象**则是类的具体化，是**类的实例。**

封装性：封装使**数据**和加工该数据的方法（**函数**）封装为一个整体，把对象的**设计者**和对象的**使用者**分开，使用者不必知晓行为实现的细节，可以增加安全性。

继承性：继承性是**子类共享父类**之间**数据和方法**的机制。一个类**直接继承**其它类的**全部描述**，同时可**修改和扩充**。可以增强**代码的复用性。**

多态性：对象根据所**接收的消息**而做出**动作。同一消息为不同的对象接收时可产生完全不同的行动**，这种现象称为多态性。使具有**不同内部结构**的对象**共享相同**的**外部接口。**可以增加扩展性。

1、指针和变量的自增自减有什么不同?

答：**变量**的自增自减是改变变量的**值**，**指针**的自增自减是改变指针的**指向地址。**

2、宏定义与操作符的区别?

答：**宏定义**是C++的**预处理**命令之一，它是一个替**换操作**，**不做计算和表达式求解**，不占内存和编译时间。

3、虚函数与纯虚函数的特点?

答：**虚函数**必须是基类的**非静态成员函数**，其访问权限可以是protected或public；**纯虚函数**是虚函数的一个子集，含有纯虚函数的类就是抽象类，它**不能生成对象。**

4、如何使用纯虚函数?

答：纯虚函数用来定义**没有意义的实现**，用于抽象类中需要交给**派生类**具体实现的方法。

5、引用与值传递的区别?

答：**值传递**传递的是一个**值的副本**，函数对形参的操作不会影响实参的值；**引用传递**传递的是引用对象的**内存地址**，函数对形参的操作会影响实参的值，实参的值会随着形参的值的改变而改变。

6、指针与引用的区别?

答：（1）引用无需解引用，指针需要解引用；

（2）引用在定义时被初始化一次，之后不可变，指针可变；

（3）引用不能为空，指针可以为空；

（4）程序为指针变量分配内存区域，而引用不需要分配内存区域，所以指针自增操作是指针变量的自增，引用自增操作是变量值的自增。

7、面向对象与面向过程的区别?

答：**面向过程**是一种以**过程为中心**的编程思想，以**算法**进行驱动；**面向对象**是一种以**对象为中心**的编程思想，以**消息**进行驱动。面向过程编程语言的组成：**程序=算法+数据**；面向对象编程语言的组成：**程序=对象+消息。**

8、面向对象的特征是什么?

答：面对对象的3个要素：**封装，继承，多态。**面向对象中所有对象都可以**归属为一个类。**

9、类与结构体有什么区别?

答：（1）**结构体**存储在**栈**中，**类的实例化**可以存储在**栈**中，也可以存储在**堆**中；

（2）结构体的**执行效率**比类要高；

（3）结构体没有**析构函数**，类有析构函数；

（4）结构体不可以**继承**，类可以继承。

10、C++覆盖与隐藏概述?

答：（1）**覆盖**指的是在**子类和父类**中，**存在函数名、参数均相同的函数**，并且**父类的该函数为虚函数**；

（2）**隐藏**指的是在**子类与父类**中，**存在函数名相同、参数不同的函数**，此时无论父类函数是否为虚函数，父类函数都会被被**隐藏**，或者存在函数名、参数均相同的函数，此时只有当父类函数不为虚函数时，父类函数才会被隐藏。

11、C++支持参数个数不确定的函数吗?

答：C++可以通过**隐藏参数机制**支持参数不确定的函数。

12、什么是内联函数?

答：在类声明的**内部**声明或定义的**成员函数**叫做**内联（inline）函数**，在内联函数内**不允许有循环语句和switch语句。**

13、什么是静态函数？如何使用静态函数?

答：**静态函数**是用**static**修饰符修饰的函数，静态函数**没有this指针**，**只能访问静态变量。**类中如果函数调用的结果**不会访问或者修改任何对象数据成员**，这样的成员声明为静态成员函数比较好。

14、函数重载及作用域?

答：函数重载是指在**相同作用域**下，具有**相同名称**而**不同参数列表**的**多个函数。**

15、什么是函数模板?

答：函数模板技术是指使用了**模板技术**定义了**参数化类型**的**非成员函数**，这使得程序能够使用**不同的参数类型调用相同的函数。**

16、什么是类模板?

答:类模板是使用**模板技术**的类，描述了能够**管理**其他**数据类型**的**通用数据类型**。类模板技术通常用于建立包含其他类型的**容器类**（队列、链表、堆栈等）。

17、什么是泛型编程?

答：泛型编程就是以**独立于特定类**实现的方式编写代码，针对**不同的类型**提供**通用**的实现。

18、C++如何实现泛型编程?

答：C++中泛型编程的实现是使用C++中的**模板技术**来实现的，主要是**设计函数模板和类模板**。

二、C++和JAVA

首先应该清楚，Java 是由 C++发展而来的，保留了 C++的大部分内容，其编程方式类似于 C++。但 Java 的句法更清晰、规模更小、更易学。Sun 公司曾对多种程序设计语言进行分析研究，取其精华去其糟粕，最终推出了 Java。Java从根本上解决了C++的固有缺陷，形成了新一代面向对象的程序设计语言。

1、解释对编译

Java是一种**解释性**（或者说半解释半编译）语言，意味着其在**执行时会被"翻译"为二进制形式**，也就是java跑得时候必须有人（jvm）去解释它（现在的Java语言，其执行方式已经不仅仅是解释执行方式了，即时编译器（JITC、just-in-time compiler）技术和原型编译技术的出现大大提高了JAVA的运行效率）。而C++则是编译语言，意味着程序**只能在特定操作系统上编译并在特定系统上运行**，也就是说C++一步到位成机器语言的。

2、指针

Java 没有指针的概念。在 C/C++中，指针操作内存时，经常会出现错误。而在Java中是没有指针这一概念的，因此也有效地防止了一系列由指针引起的操作层失误（如指针悬空所造成的系统崩溃），更有利于 Java 程序的安全。

3、内存安全

Java是一种**内存安全型语言**，意味着大家可以为**给定数组分配任意参数**，即使超出范围也只会返回错误提示。C++更为灵活，但代价是一旦分配的参数超出资源范围，则会引起错误甚至严重崩溃。

4、多重继承

Java不支持多重继承。多重继承，它允许多父类派生一个子类。也就是说，**一个类允许继承多个父类。**尽管多重继承功能很强，但使用复杂，而且会引起许多麻烦，编译程序实现它也很不容易。所以 **Java 不支持多重继承，但允许一个类实现多个接口。**可见，Java 既保留了 C++多重继承的功能，又避免了 C++的许多缺陷。

5、数据类型

Java 是完全面向对象的语言，所有方法和数据都必须是类的一部分。除了基本数据类型之外，其余类型的数据都作为对象型数据。例如，对象型数据包括字符串和数组。类将数据和方法结合起来，把它们封装在其中，这样每个对象都可实现具有自己特点的行为。而 **C++将函数和变量定义为全局的，然后再来调用这些函数和变量**，从而增加了程序的负担。此外，Java 还取消了 C/C++中的结构和联合，使编译程序更加简洁。

6、自动内存管理

Java 自动进行无用内存回收操作，不再需要程序员进行手动删除。Java 程序中所有的对象都是用 new 操作符建立在堆栈上的，这个操作符类似于 C++的“new”操作符。当 Java 中一个对象不再被用到时，无须使用内存回收器，只需要给它添加删除标签，无用内存的回收器便利用空闲时间在后台运行。而 C++中必须由程序释放内存资源，这就增加了程序员的负担。

7、操作符重载

Java 不支持操作符重载，操作符重载被认为是 C++的突出特征。操作符重载，就是把**操作符(比如'+，-，*，/'这些运算符)赋于新的意义**， 来完成更为细致具体的运算等功能。要实现操作符重载，就要使用**操作符重载函数**，而运用函数就肯定会存在各种限制条件以及特殊情况。特殊情况就需特殊处理，因此操作符重载还是比较繁琐的。Java语言是走“简洁风”的，因此为了保持Java语言的简洁性，便毅然抛弃了操作符重载这一功能，但是为了避免舍本逐末的情况，**Java语言还是可以通过类来实现操作符重载所具有的功能的。**

8、预处理功能

C/C++在编译过程中都有一个**预编译阶段，即预处理器。**预处理器为开发人员提供了方便，但增加了编译的复杂性。**Java 允许预处理，但不支持预处理器功能**，因为 Java 没有预处理器，所以为了实现预处理，它**提供了引入语句（import），但它与 C++预处理器的功能类似。**

9、缺省参数函数

Java 不支持缺省参数函数，而 C++支持 。在 C 语言中，代码组织在函数中，函数可以访问程序的全局变量。后来C++增加了类，提供了类算法，该算法是与类相连的函数，C++类方法与 Java 类方法十分相似。由于 C++仍然支持 C语言，所以 **C++程序中仍然可以使用 C 的函数，结果导致函数和方法混合使用**，使得 C++程序混乱，而**Java 没有函数(准确说叫方法)。**作为一个比 C++更纯的面向对象的语言，**Java 强迫开发人员把所有例行程序包括在类中。**事实上，用方法实现例行程序可激励开发人员更好地组织编码。

10、字符串

C 和 C++不支持字符串变量，在 C 和 C++程序中使用**“Null”终止符代表字符串的结束。在** Java 中字符串是用类对象（String 和 StringBuffer）来实现的，**在整个系统中建立**字符串和访问字符串元素的方法是一致**的。**Java 字符串类是作为 Java 语言的一部分定义的，而不是作为外加的延伸部分。此外，**Java 还可以对字符串用“+”进行连接操作。**

11、goto 语句

“可怕”的 goto 语句是 C 和 C++的“遗物”，它是该语言技术上的合法部分。goto语句也称为**无条件转移语句**，通常**与条件语句配合使用，用来实现条件转移， 构成循环**，跳出循环体等功能。但是，在结构化程序设计中一般不主张使用goto语句， 以免造成程序流程的混乱，使程序的可读性变差,增加程序调试的难度。 **Java 不提供 goto 语句，虽然Java指定 goto 作为关键字，但不支持它的使用**，这使程序更简洁易读。

12、类型转换

在 C 和 C++中，有时会出现数据类型的**隐含转换**，这就涉及了**自动强制类型转换**问题。例如，在 C++中可将一个浮点值赋予整型变量，并去掉其尾数。**Java 不支持 C++中的自动强制类型转换，如果需要，必须由程序显式进行强制类型转换。**

应用场景：

java侧重于**大型企业级**应用开发，**C++**侧重于**底层**应用开发。Java是Android开发领域的王者，因此移动开发者无疑应该选择它作为项目基础。另外，Java也常见于Web及桌面应用乃至服务器端应用。

C++更接近机器语言，因此其软件运行速度更快且能够直接与计算机内存、磁盘、CPU或者其它设备进行协作。另外，C++也能为游戏提供良好的运行性能。