以Java和C++为例，说明OOPL中单继承和多继承的差异和优缺点。

差异：单继承的每个子类只有一个超类，多继承的每个子类可以有多个超类。

单继承的缺点：由于每个子类都只有一个超类，重用只能发生在一个继承路径上，无法重用来自不同继承路径上的代码。

单继承的优点：不用考虑多继承中名字冲突和重复继承等问题。

多继承的缺点：需要处理来自不同超类的名字冲突：当两个或多个不同超类对接口中的某些元素使用相同的名字时，就可能发生名字冲突，如同名的实例变量或方法。名字冲突可能导致多继承的子类的二义性行为；多继承还会产生重复继承，比如菱形继承结构。

多继承的优点：多继承可以让子类复用多个父类代码，并且所有父类型都可以接收子类示例。

Java禁止多继承，但通过“单继承加多接口”的方式模拟多继承，通过接口实现外部功能契约的继承，但和多继承体现的多复用有区别。接口继承需要父类对象参数的函数，可以接受子类对象参数，并且可以通过父类的接口使用子类对象的方法。Java可以通过委派实现多重继承，把从接口调用的方法，都明确地委派给实现共同功能的对象，这相当于手动实现了多重继承，这使得子类可以直接复用父类的代码。在Java中也可以不用委派实现多继承，而是把实现共同功能的对象作为成员变量使用。这样操作对象并不需要直接实现接口，而只是作为属性保存一个实现共通功能的对象，在程序中直接调用该属性的方法。

C++允许多继承。C++中的接口类仅含有声明，除了析构函数外，接口类中都是纯虚函数。C++并不像Ruby语言一样提供Mix-in支持，但是用户可以在具体使用多继承时根据自己的理解进行设计。C++中的继承会保留基类的所有数据成员的副本，虚函数则会使用虚函数表保存。在解决重复继承问题时，C++提供了虚基类。虚基类的初始化工作只会交给最后的派生类进行。但是这增加了用户使用C++多继承的复杂性。在名字冲突问题上，C++使用基类名限定有冲突的函数名：using ClassName::function. 在函数选择的优先级上，处于继承层次的子类方向上的优先级更高。