1. 什么是抽象，什么是封装？什么是数据抽象和数据封装？

抽象：
抽象(Abstraction)是简化复杂的现实问题的途径，它可以为具体问题找到最恰当的类定义，并且可以在最恰当的继承级别解释问题。它可以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。抽象并不打算了解全部问题，而只是选择其中的一部分，暂时不用部分细节。抽象包括两个方面，一是过程抽象，二是数据抽象。它侧重于相关的细节和忽略不相关的细节。抽象作为识别基本行为和消除不相关的和繁琐的细节的过程，允许设计师专注于解决一个问题的考虑有关细节而不考虑不相关的较低级别的细节。
封装：
封装，即隐藏对象的属性和实现细节，仅对外公开接口，控制在程序中属性的读和修改的访问级别；将抽象得到的数据和行为（或功能）相结合，形成一个有机的整体，也就是将数据与操作数据的源代码进行有机的结合，形成“类”，其中数据和函数都是类的成员。

数据抽象：
数据的使用者只需要知道对数据所能实施的操作以及这些操作之间的关系，而不必知道数据的具体表示形式。
数据封装：
指把数据及其操作作为一个整体来进行描述。
数据的具体表示对使用者是不可见的（封装），对数据的访问（使用）只能通过封装体所提供的对外接口（操作）来完成。

2. 什么是类，什么是对象？

对象是由数据及能对其实施的操作所构成的封装体，它属于值的范畴。
类描述了对象的特征（包含哪些数据和操作），它属于类型的范畴（对象的类型）。
数据：数据成员、成员变量、成员函数的局部变量等
操作：成员函数、方法、消息处理过程等

3. 面向对象程序设计与面向过程的程序设计有什么区别？

面向过程编程是一种以过程为中心的编程思想,分析出解决问题的步骤，然后用函数把这些步骤一步一步实现。面向过程编程，数据和对数据的操作是分离的。
面向对象编程是将事物对象化，通过对象通信来解决问题。面向对象编程，数据和对数据的操作是绑定在一起的。
面向对象的三大基本特征：
封装：把客观事物封装成抽象的类，隐藏属性和方法的实现细节，仅对外公开接口。

继承：子类可以使用父类的所有功能，并且对这些功能进行扩展。继承的过程，就是从一般到特殊的过程。
多态：接口的多种不同的实现方式即为多态。同一操作作用于不同的对象，产生不同的执行结果。在运行时，通过指向基类的指针或引用来调用派生类中的虚函数来实现多态。
封装可以隐藏实现细节，使得代码模块化；继承可以扩展已存在的类。它们的目的都是为了---代码重用。
而多态则是为了实现另一个目的---接口重用。
面向对象的代码更加支持重用，能降低软件开发和维护的成本，提高软件的质量。

4．什么是类成员的访问控制？C++中提供的类成员的访问控制机制有哪些？

对类成员访问权限的控制，是通过设置成员的访问控制属性实现的。访问控制属性有以下三种：public,private 和 protected。
公有类型成员用 public 关键字声明，任何一个来自类外部的访问都必须通过这种类型的成员来访问（“对象.公有成员”）。公有类型声明了类的外部接口。

私有类型成员用 private 声明（若私有类型成员紧接着类名称，可省略关键字），私有类型的成员只允许本类的成员函数来访问，而类外部的任何访问都是非法的。这样完成了私有成员的隐蔽。在不考虑继承的情况下，

保护类型（protected）的性质和私有类型的性质一致。即保护类型和私有类型的性质相似，其差别在于继承过程中对产生的新类影响不同。
若在类的内部访问时，所有成员之间都可以通过成员名直接访问，这就实现了对访问的有效控制。