指针（pointer）和 int，char 类似，是一种独立的数据类型。区别在于，当我们说“指针”时，其实是说一系列的数据类型（泛指），就像我们说“数值型数据类型”，也是泛指（包括 int, float, double 等等）。当我们具体说 “int 型指针”时，我们是说 int *(而不是 float* 或者 double*），就像是指出了数值类型中的某一个具体类型（例如 int）。

下面以 int *a = &b; 为例，并假设这行代码的运行环境下，int 为 4 字节长。

1，指针是地址吗？ 不是。上面已经说了，指针本身具有独立的数据类型，我们可以用这个类型声明一个变量并赋值，只不过指针所对应的变量用途比较特殊，是用来保存某个内存地址的。也就是说，指针本身占用了一小块内存空间（它本身不是地址，地址就是一串数字，完全不搭边儿），而这块内存空间是用来写入一个地址数据的。

2，既然指针是用来保存地址的，也就是保存一串数字而已，为什么还要有类型？ 类型系统有两个重要作用，一个是用作编译时类型检查，让编译器帮助我们避免犯错，比如上面的例子，我声明的是一个 int 型指针，但如果赋值时使用的 b 不是 int 型（可能酒后写代码不小心写错了），如果没有指针类型检查的话，这个错误可能只有在程序运行的时候才能发现。 类型的另外一个作用是，指明一个内存地址所保存的二进制数据，应该怎样被解释。想想看在没有指针类型的情况下，虽然我知道了 a 中保存的地址，但当我想解引用指针（*a），*读出这块地址中的一个 int 型数据时就有问题了，编译器怎样知道你从这个位置开始要读出一个字节，还是 4 个字节？而在有 int* 这个类型的指导下，编译器知道在解引用的时候读出 4 个字节（int* 对应的指针偏移量是 4 字节），甚至，你也可以通过强制类型转换读出一个 char 类型数据来玩。