指针-对于一个类型T，T\* 就是指向T的指针类型，也即一个T\*类型的变量能够保存一个T对象的地址，而类型T是可以加一些限定词的，如const、volatile等等。  
引用-引用是一个对象的别名，主要用于函数参数和返回值类型，符号X&表示X类型的引用。

* + 不同点：

本质： 指针指向一块内存，它的内容是所指内存的地址；而引用则是某块内存的别名，引用不改变指向。

* + 1. 引用不可以为空，但指针可以为空。定义一个引用的时候，必须初始化。因此使用指针之前必须做判空操作，而引用就不必。
    2. 引用不可以改变指向，对一个对象"至死不渝"；但是指针可以改变指向，而指向其它对象。
    3. 引用的大小是所指向的变量的大小，因为引用只是一个别名而已；指针是指针本身的大小，4个字节（32位）。
    4. const int\* p-> 指向常量的指针，int \* const p->本身是常量的指针.后者需要在定义的时候初始化。对于引用来讲int const & p=i;和 const int &p=i;没什么区别，都是指指向的对象是常量。
    5. 引用和指针的++自增运算符意义不同，指针的++表示的地址的变化，一般是向下4个字节的大小（一个只针的大小），引用的++就是对应元素的++操作。
    6. 指针传递和引用传递
       - 指针传递参数本质上是值传递的方式，它所传递的是一个地址值。值传递过程中，被调函数的形式参数作为被调函数的局部变量处理，即在栈中开辟了内存空间以存放由主调函数放进来的实参的值，从而成为了实参的一个副本。值传递的特点是被调函数对形式参数的任何操作都是作为局部变量进行，不会影响主调函数的实参变量的值。
       - 引用传递过程中，被调函数的形式参数也作为局部变量在栈中开辟了内存空间，但是这时存放的是由主调函数放进来的实参变量的地址。被调函数对形参的任何操作都被处理成间接寻址，即通过栈中存放的地址访问主调函数中的实参变量。正因为如此，被调函数对形参做的任何操作都影响了主调函数中的实参变量。