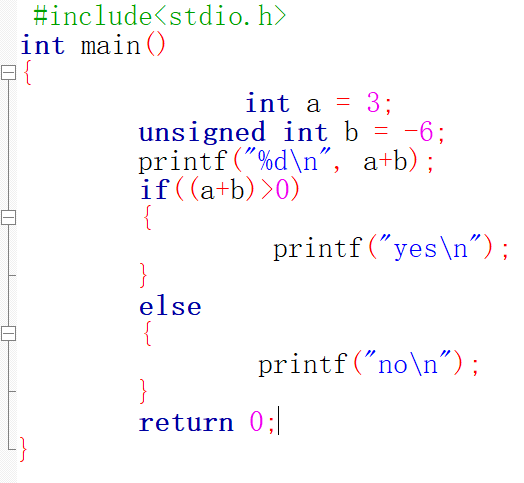
**1、int类型与unsigned int运算的问题**





知识点：

①int类型和unsigned int类型的差别。unsigned int类型比int类型的容量大是因为，unsigned int是无符号类型的，所以最高位不表示正负，而int类型的最高位是表示正负的，1表示负数，0表示正数。

②计算机中负数的存储方式——其补码。在计算机中，负数存储的方式都是以其其补码形式储存，例如：6在计算机的存储形式是0000 0000 0000 0110（32位计算机中），而-6是1000 0000 0000 0110 （32位机下）（最高位是符号位），补码也就是1111 1111 1111 1010（32位计算机中），补码=原码取反+1。

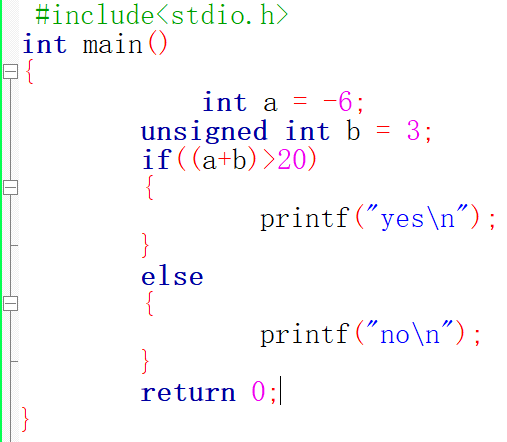
③赋值一个负数给unsigned int类型的变量，它的意思就是将-6这个数的补码形式（1111 1111 1111 1010（32位机下））形式存到一个叫b的空间中，下次我读取这个数的时候，是使用b这个名字，按照b这个名字的类型来读取的，所以当b是unsigned int类型的时候，就不考虑最高位是符号位，则b = 1111 1111 1111 1010（32位机下）（因为是无符号），当b是int类型的时候，考虑最高位是符号位（1为负数，0为正数），当发现是1的时候也就是负数的时候，计算机会把它转换成为原码从而读出是-6。所以一个变量的类型是决定读取这个变量存储的数的方式。

④隐式转换。当unsigned int类型与int类型运算时，如果没有指出转换的类型，它会进行“隐式转换”转换为unsigned int类型，也可以记住，不同类型进行运算时，没指出转换的类型时，它会往容量大的那总类型转换，因为这样可以更加容易避免溢出嘛。

原文链接：<https://blog.csdn.net/gyyu32g/article/details/79089475>

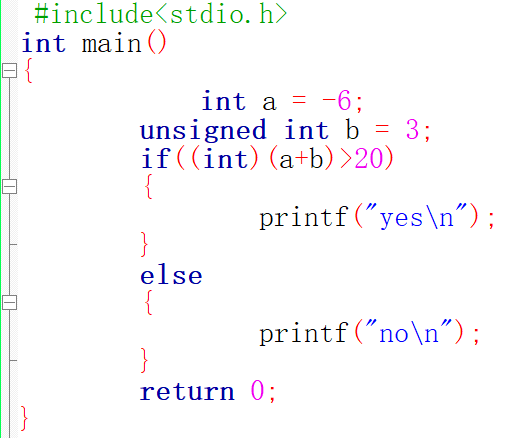
代码分析：

首先，定义了一个 int a = 3 。a的存储以3的二进制形式存储。 unsigned int b = -6 。b的存储是以其补码存储的，也就是1111 1111 1111 1010（32位机下）。当执行a+b的时候，printf函数要求指定打印出%d，所以，a以int（%d）类型读取出来，也就是3；b也是以int（%d）类型读取出来，也就是-6，所以3+（-6）=-3，发现打印结果是-3。而在后面的 if((a+b)>0) 判断中，(a+b)中并没有指定读取的类型，所以就进行隐式转换，a就转成unsigned int类型，也就是3；b转成unsigned int类型，因为b本来的存储是1111 1111 1111 1010（32位机下），转成unsigned int类型时不考虑符号位，所以就直接以补码的形式转化，也就是65530（32位机下），所以a+b=65533>0，输出yes。



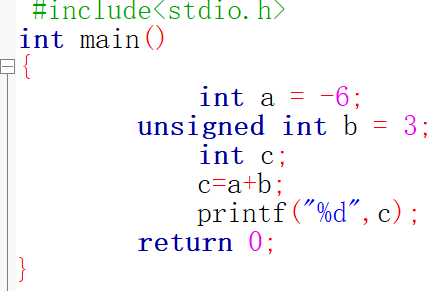


改正：





引申：





**明明的随机数（入门P1059）：**

使用bool类型

Bool A

If（A）

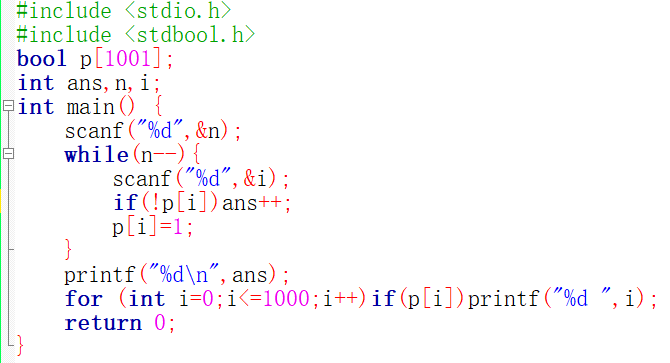
则A值为1

If（！A）

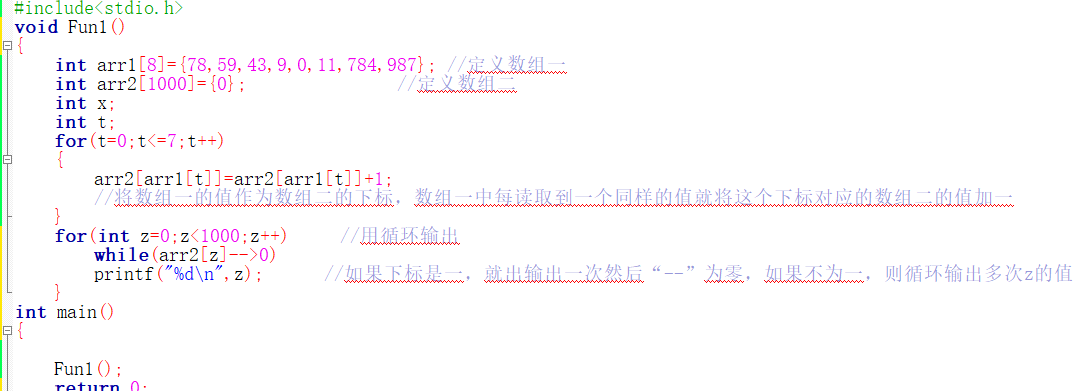
则A值为0

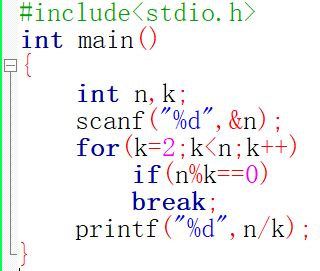
转换方式，灵活运用数组a[i]中的i。

i不仅仅可以是数组中的一个数，也可以本身为一个数



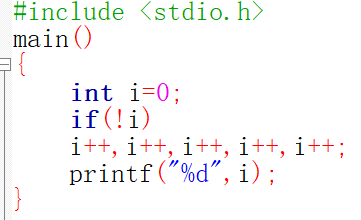
桶排序：

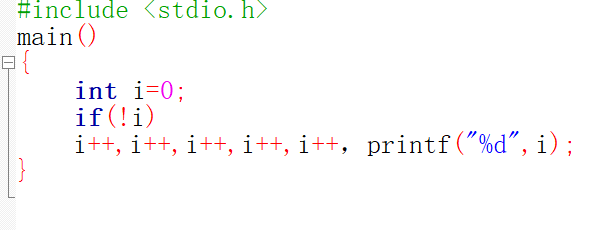


**唯一分解定理:一个数能且只能分解为一组质数的乘积。由此这几个质数很好找，排序也很简单。****两位质数代码**

**若为多位，可以设个函数来多次求直到最后一个数为质数。**

**If中的语句可以用逗号相连接（仅限同样的）**

**例：****5个语句均可执行；**

**报错**

**Fab(x) x的绝对值**

**&&与||是同级，看哪个在前就先算哪个。要说优先级，它们的优先级体现在结合上，即++b与++c结合起来做与运算&&，即(++a || (++b && ++c))**

**A？x:y**

一个条件表达式绝不会既计算x，又计算y。

**Int a=5；**

**Int \*d**

**d的类型为int\*型(指向int型的指针)**

**d=&a； &取得a的地址**

**d用来保存a的地址**

**\*d \*调用指针，若不加\*，则d值为a的地址**

**因此，也可以手动输入地址来使某个指针指向某个变量**

**#include <stdio.h>**

**#include <limits.h>**

**int main()**

**{**

**int a=5;**

**int \*d;**

**d=&a;**

**int \*c=6422028;**

**printf("%d",\*c);**

**} (a地址在十进制下为6422028)**

**&用于取址，若定义一个数组，因为数组名为一个指针，因此可直接**

**Printf(“%x”,a);获取其地址，但若再在a前边加一个&，输出结果仍然不变**

**int main()**

**{**

**int a[5]={1,2,3,4,9};**

**printf("%x",\*a);**

**}**

**输出结果为1.**

**原因：a定义为一个指向数组的指针，a用来保存地址，\*a即调用指针，输出数组中的第一个数。**

**若将\*a换为\*a+6，则输出7，因为\*优先级高于+，先调出1再＋6.**

**若改为\*（a+4），则输出3，（）优先级高于\*，调用第一个元素后的第四个元素，即9.**