1、注意区分“==”和“=”：

在if后的小括号里，特别容易发生把“==”写成“=”的错误。
这是非常非常非常常见的初学者的失误。
而且这种编译器检查不出来的错误，它很容易让人产生“今天我和代码一定至少坏掉了一个”的怀疑人生的想法。

比如你写了这样的代码：

```
char x;
cin >> x;
if (x='a') cout << "x is a";
else cout << "x is not a";
```

你想象中的代码应该要能实现的功能：程序能够判断我输入的东西是不是字母a。
但实际上，不管你输入什么，程序就像瞎了一样，只会告诉你，它们都是字母a。

因为在以上代码中，由于if的条件表达式会先执行将x赋值为'a'，再判断x是否为真，因此该条件恒为真，导致该if-else语句只会执行then子句。

所以，坏掉的不一定是你或者世界，只是疏漏了一个小小的等号而已。

2、表示是否为真/假的关系表达式：

表达式x==true可以简写为x（x为bool类型的变量，或结果为bool类型的表达式）。
表达式x==false可以简写为!x。

3、逻辑运算：

逻辑运算符：&& || ！【要记住这些东西叫逻辑运算符，不然以后如果听课听到逻辑运算，可能会短暂迷茫】
【逻辑运算的真假判断规则，你还记得吗？】
参加逻辑运算的数据如果不是bool类型，要先转换为bool类型：0为假，非0为真
【当bool类型参加算术运算时如何转换为整数？】

&&表达式和||表达式在执行时，先处理左边，如左边已能决定此逻辑表达式的结果，则右边不执行。【做选择题或者计算程序结果的时候，一定要记得这条规则】
为了提高程序执行效率，应该把false的概率比较大的表达式放在&&的左边，把true的概率比较大的表达式放在||的左边。

4、四大湖问题的逻辑表达：

四大湖问题中可以用关系表达式+算术表达式来表示每位同学都只说对了1个。
【具体怎样表示？尝试自己写一下】

5、if-else：

在if语句中，如果then子句或else子句不止一个语句，要加大括号（下图中红色）：

```
int x = 10, y;
char c = '0';
cin >> y;
if (y < x) {
        if (y >= 0)
        c += y;
cout << c << " is not " << y;
} else
cout << "y is no less than x";
```

```
int x = 10, y;
char c = '0';
cin >> y;
if (y < x) {
        if (y >= 0)
                c += y;
        cout << c << " is not " << y;
} else
        cout << "y is no less than x";
```

在写if…else…结构的代码时，要注意子句的缩进排列。
在上图中，左边的代码缩进不合理，右边的代码则缩进合理。【想想为什么】
【if(……)后面不要加分号，不然你可能又会怀疑程序或者你坏掉了】

6、常见条件判断：

判断闰年：四年一闰，百年不闰，四百年又闰。【怎样写成逻辑表达式？】
判断一个字符变量x的值是否为字母：'a'<= x && x <= 'z' || 'A' <= x && x <= 'Z'
【这两个之前期末都考过，有小部分同学做错，手写代码尤其不要忽视细节】

7、问号冒号表达式：

【问号冒号表达式的执行机制是怎样的？】
问号冒号表达式的优先级很低，**比输出运算符<<还低**，要注意加括号。【写代码一定要注意，凡是不确定优先级的地方，就加括号，不然会有奇怪的输出】
冒号两边的表达式的结果应当是相同的类型（或可以自动转换成相同的类型）。

8、switch-case：

switch后的小括号里可以是整型、字符型、布尔型或枚举类型，**但不能是浮点型**。
case 后面**也不能是浮点型，且必须是常量**，不能是一个包含变量的表达式。
比如 case x < 3: cout << "x is less than 3"; 是不合法的，因为x < 3 不是常量表达式。
但 case 5 < 3: cout << "5 is less than 3"; 是合法的，因为 5 < 3 是常量表达式，结果恒为false（相当于0），当switch 后的小括号里的表达式等于0 时，就会输出 5 is less than 3。
一个switch下的所有case语句（以及default语句）都包括在一个大括号里。

```
switch (x%3) {   // 这个大括号不要漏了
      case 0: cout << "x%3=0"; // case 后面的 0 是常量
      case 1: cout << "x%3=1"; // case 后面的 1 是常量
      case 2: cout << "x%3=2"; // case 后面的 2 是常量
}  // swtich 语句里可以没有 default
```

【switch的执行机制是怎样的？比如上图中的代码在x为4时会输出什么？】
在switch语句的执行过程中，如果遇到break 语句，则跳出switch的大括号。
【上图的代码中是否应该加入break？如果是，至少加入几个？】

附：使用switch语句时需要注意的问题↓

```
switch (x) {
    case0: cout << "x is 0";                 // case后面漏了空格
    case x=='1': cout << "x is 1";           // case后面只能是常量，不能包含变量
    case '2' || '3': cout << "x is 2 or 3"; // 等价于case true，跟'2'、'3'无关
    case '4': cout << "x is 4";              // 漏了break;
    case '5': cout << "x is 5";
}                                            // 没有default，不能处理其它情况
```

9、随机数：

生成随机数可使用<cstdlib>库中的函数rand()。【如果代码里要使用随机数函数，首先需要在预编译命令里包含<cstdlib>这个库。其实还有另一个库也需要包含进来，你还记得是什么吗？】
【如何将rand()产生的随机数映射到一个整数区间中，且使各个数的概率相同？】
如果不重置种子，那么每次生成的随机数都是基于上一个随机数计算出来的。
重置随机数种子的函数是srand(种子的值)。
种子的值通常使用当前系统时间time(NULL)，该函数属于<ctime>库。
在一个程序里只要设置一次随机数种子即可。
如果把调用srand的语句写在了循环语句里，可能会造成反复设置相同或近似的种子（因为计算机执行循环的速度很快，连续的几个循环周期所处的系统时间非常接近），使得随机数过于相近。