程序的逻辑结构

程序可以理解为一台状态机，每个变量的取值就是一个状态，程序就是一个状态的集合，也就是每一个变量在它自己的区间里面只能取一个值。那么赋值就可以理解为状态的迁移。从一个状态到另一个状态，如果有一个变量依据其他变量，就叫他条件变量，如果变量之间没有关系，就叫线性无关。

一般情况下，应该保证每个类里面的变量是线性无关的。但是有时候考虑到空间的利用效率，可能会引入中间变量，这样就导致了变量的相关性，就可能带来同步问题。导致代码难以维护。因为这个时候，相关的变量只要有一个变化，那么相应的变量都要同时更新。

所以在写代码的时候，可以先用效率较差的方法，保证变量的独立性，每次可以通过函数去计算上一层所需要的变量。或者保存一个局部变量。

但有时候虽然可能全部是独立变量，但是也可能超出变量的取值范围，这个时候就要在赋值的时候预先判断赋值的有效性。这个可能是引起bug的根源，而不是等到用该值是才判断。

变量的赋值都可以理解为一个函数，那么函数可以分几类，一类是赋值的函数，另一类是取值的函数，另一类是查询的函数。

可否找到某个不变的东西 或者 等价关系 什么的

一段逻辑代码有很多逻辑结构，根据需求的不同，每个人的编码方式也不一样，那么是不是有唯一的表达方式，

考虑成黑合的形式。外部不需要知道具体的细节。外部如果有公用的模块，那么就可以把它抽象出来。

能不能类似于 顺序，判断 ，循环 这样三种结构，搭建框架

是否能避免出现三角关系。就是三个模块相互牵扯在一起。但如果有些东西要贯穿始终呢

Cocos2d设计模式

1. QAbstractScene抽象类。主要控制场景切换，初始化，切换时的特效，是否需要loading
2. QAbstractLayer抽象类。主要控制载入资源，初始化，状态更新，逻辑管理
3. QUIBuild抽象类，主要根据xml文件生成相应的UI控件，如QSpriteBuild, QLabelBuild. 这里只适合静态的控件。
4. QStateMachine类，管理所有的全局状态，这里只限于bool状态，或者一定数量的枚举类型。不适合带数据类型的状态。区分一次性赋值或多次赋值。
5. QDataManage类，管理所有的全局数据，这里可以再考虑再细分，比如io类，web类，config类等等。尽量用较少的状态，比如只区分是哪一个layer,剩下的让layer自己区分。
6. 使用全局的update函数，以同步所有的状态。