# 一点小小的帮助

用来检验你的五级流水的每一个步骤是否出错，此时几乎不需要处理hazard问题

### 第一步 实现一个没有并行的五级流水

指令一：IF->ID->EXE->MEM->WB ->指令二 IF->……

Hint：只需要完成五个函数，然后在一个run()里把它们串联起来就可以了。你需要完成的其实就是你的一个模拟cpu，里面可以有内存、寄存器、pc等等。

*特别注意，每一个流水线步骤的连接处，需要保存上一个步骤完成后的指令状态。

### 第二步 修改连接，使之变成五级流水

指令1 WB->指令2 MEM->指令3 EXE->指令4 ID->指令5 IF->指令2 WB->……

这里开始就实现了伪并行了，每个stage对应的指令已经不一样了

思考：为什么倒着执行就可以实现伪并行了呢?

除了修改运行顺序，还需要处理hazard问题，如何检测?

当ID解码后（或EXE计算后）计算出需要跳转，应该怎么办?

当EXE需要用前一条指令的MEM结果时，应该怎么办?

### 第三步 加上分支预测

如果指令需要跳转，那么你会让cpu阻塞（或者塞进去一些NOP空指令），这时候就导致cpu的利用率下降。有一种好办法就是实现一个分支预测器，让你的模拟器在前面几次的跳转中吸取经验，预测下一次是否跳转。如果预测正确，那么不需要清空后面的buffer槽，否则还是和第二步中一样。

分支预测的方法有很多，最基本的就是2位计数器饱和预测。

00(不跳转)<–->01(不跳转)<–->10(跳转)<–->11(跳转)

### 第四步 （bonus）forwarding

前面已经提到，有可能指令1的MEM阻塞了指令2的EXE，那么你可以把MEM的结果传回EXE，EXE就可以不用从寄存器获取而是直接获取MEM/WB阶段的数据，可以加快流水，减少阻塞。