# readme

为什么写这个readme？

1. 实验报告语言描述比较拘谨，格式也要求比较严格，为了流畅自然地“把话讲清楚”，还是单独写一个readme

本文包含以下内容：

1. 项目介绍
2. 解释器的大概描述
3. 开发过程的大概描述
4. 描述一下接口

在这个项目中，我尝试写一个lua解释器。所谓解释器，就是执行一个语言的程序。lua解释器，就是执行lua代码的程序。至于解释器用什么语言写，可以用任何语言写，甚至可以用lua本身写。

解释器包含两个部分，前端是解析然后代码生成（生成汇编代码），后端是虚拟机，执行汇编代码。可以认为是一个编译器加上一个虚拟机。编译器不讲，虚拟机其实在机组实验就接触了，用Verilog写的mips cpu其实是一个虚拟机。一个虚拟机主要是一个虚拟机循环，在循环中取指，译码，然后执行。

这个项目我从三月初开始做，一开始看《游戏脚本高级编程》，写了一个汇编器+虚拟机，然后学编译原理，然后下定决心打算写一个带有编译器性质的东西，最后锁定在lua上，这中间有很多原因，不讲。我做这个项目的原因是学习lua源代码，顺便实践一下编译原理。

那么我是怎么学的呢？流程大概如下：

1. 学习如何写lua代码，lua的基本语法等
2. zhihu上有一个人实现了C#的lua解释器，并写了两篇文章，看掉；看他的解释器带啊没，没有任何注释和手册，卒
   1. 他的项目写了三年，而且已经商业上使用了
3. github上找C#的lua解释器，选择了Kopilua，看不懂，后来知道是lua源代码的直接翻译
4. 看lua源代码，看不懂
5. 找到一本书《lua设计与实现》，讲了一点实现，配合着看了一点lua源代码，看懂了一些部分
6. 阅读lua相关论文
7. 在这个时候，我觉得自己大概写不了一个前端的解析器，大作业开题展示时把项目目标定为后端虚拟机实现
8. 然而我知道了antlr，花一周学习
9. 花一周写一个一个前端，代码生成的时候卡住了，不知道该怎么写，lua源代码看不懂，而且用了antlr就要靠自己想了
10. 知道了栈式虚拟机比寄存器式简单，了解了一下，自己yy了一个代码生成，把前端和后端虚拟机接上了
11. 虚拟机怎么实现的呢？前面写的汇编器+虚拟机时，把框架保留重新写的（汇编器+虚拟机写的是xasm这个汇编语言的解释器）
12. 到此为止python实现能正确执行，这个时候是五月九号
13. 另外一条线的C#版本从四月十七号开始写data model，然后把antlr配置上去，五月十一号写完，包括实验报告
14. 这个项目就算结束了，因为花费了太长时间
15. 后面复盘了两次，修改了一些bug

解释器作为一个代码项目，和应用程序不同，没有业务逻辑，全是算法和架构设计（我偏向后者，目前没有深入某一具体算法）。如果要说有接口的话，仅有一个dofile(path)，它执行一个lua代码文件。我生成了一份文档，但是因为封装很困难，效果不是特别好。总之我不推荐阅读或者理解我的代码。

总结

1. 这个项目是非常难的，编译原理和lua源代码都很难
2. C#代码700行，但是实现了核心框架，个人认为这比堆一些没用的指令实现重要
3. 比代码重要的是对项目的理解，阅读文章和书占了80%以上的时间
4. 如果把目标定为完整实现lua（那么要实现部分标准库，至少要有标准库的框架auxlib），那么这个项目的进度是20%