

Lab03 Better Angels

[实验目的]

1. 根据收到的程序末四行的数据，计算出程序作者的学号
2. 优化收到的程序，使其完成操作所执行的指令数尽量少

[实验过程]

1. 猜出作者的学号

- 程序末四行的数据

```
a .FILL #930;
b .FILL #50;
c .FILL #1014;
d .FILL #470;
```

- 通过python求出0-16384对应的F(n)值，存入列表，对4段数据通过值检索列表中的下标，再合成学号，代码如下：

```
from icecream.icecream import ic
def find(a,b,c,d):
    list = []
    list.append(1)
    list.append(1)
    list.append(2)
    for i in range(3,16385):
        x = (list[i-1]+2*list[i-3])%1024
        list.append(x)
    aa = [k for k in range(len(list)) if (list[k] == a) and (k<22) and (k>17)]
    bb = [k for k in range(len(list)) if (list[k] == b) and (k<100)]
    cc = [k for k in range(len(list)) if (list[k] == c) and (k<30)]
    dd = [k for k in range(len(list)) if (list[k] == d) and (k<100)]
    for i in aa:
        for j in bb:
            for m in cc:
                for n in dd:
                    seg1 = str(i) if len(str(i)) == 2 else '0'+str(i)
                    seg2 = str(j) if len(str(j)) == 2 else '0'+str(j)
                    seg3 = str(m) if len(str(m)) == 2 else '0'+str(m)
                    seg4 = str(n) if len(str(n)) == 2 else '0'+str(n)
                    ic("PB"+seg1+seg2+seg3+seg4)
if __name__ == '__main__':
    find(930,50,1014,470)
```

- 结果如下:

```
PythonScripts 1.0\python.exe
aw_pictures
aw_pictures_pic
csFunc.py
oe2Png.py
oeFunc.py
ind.py
omepage.coe
omepage.png
ab.mp4
ab.txt
music.coe
music.png
ptimized.txt
iano.coe
Find ×
F:\Anaconda\envs\Lab\python.exe F:/PythonScripts/Find.py
ic| "PB"+seg1+seg2+seg3+seg4: 'PB20081519'
ic| "PB"+seg1+seg2+seg3+seg4: 'PB20081583'
ic| "PB"+seg1+seg2+seg3+seg4: 'PB20621519'
ic| "PB"+seg1+seg2+seg3+seg4: 'PB20621583'

进程已结束，退出代码为 0
```

考虑到我们学校学号的命名规则，只可能为PB20081519或PB20081583,在群中查找后确
定为PB20081583

2. 优化程序

- 思路:

原程序通过反复计算 $F(n)$ 的值，直至 n 达到所求的值，此过程中涉及多次比较与计算，而这些比较与计算在此思路下不可避免，如果可以避免比较直接获取所求的值就可以在 $O(1)$ 的时间内找到所求的值，对于避免计算可以采取预先求出所有可能的输入对应的 $F(n)$ 进行存储，而对于比较，最近在数据结构课上的一些查找算法中也涉及大量的比较，但后来介绍了哈希的思想，可以在 $O(1)$ 的时间内查找成功，再次思路下重写代码，将输入值与基本地址相加即得所求结果的地址，载入R7即可

由于这样相当于用大量内存换效率，代码也较长，故用python生成，代码如下：

```
def Fib():
    list = []
    list.append(1)
    list.append(1)
    list.append(2)
    for i in range(3, 20000):
        x = (list[i - 1] + 2 * list[i - 3]) % 1024
        list.append(x)
    with open("optimized.txt", "a") as f:
```

```

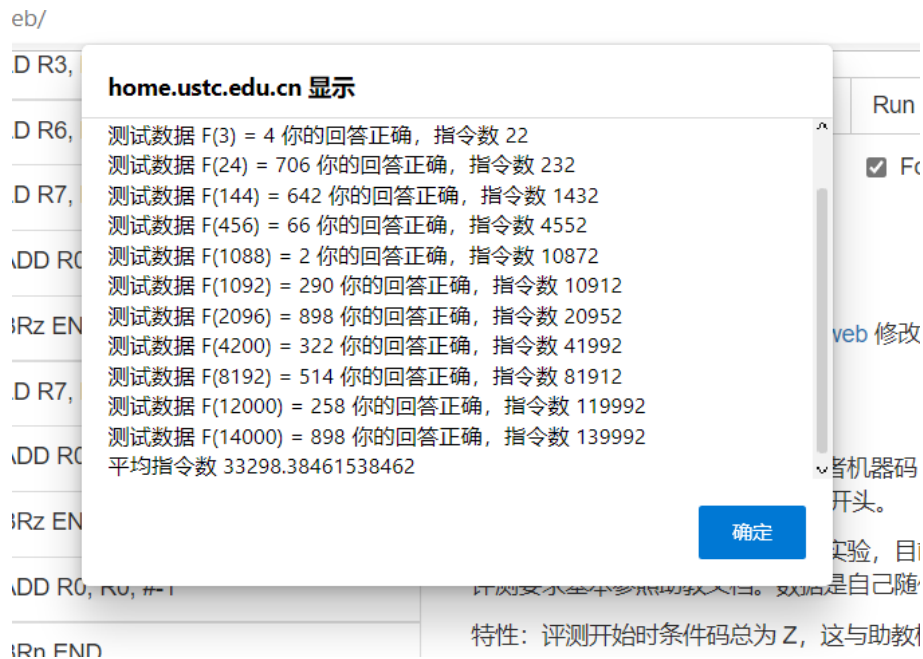
f.write(".ORIG x3000\n")
f.write("LD R1 base\n")
f.write("ADD R1 R1 R0\n")
f.write("LDR R7 R1 #0\n")
f.write("HALT\n")
f.write("base .FILL x3005\n")
for i in range(0,20000):
    f.write(".FILL #{}}\n".format(list[i]))
f.write(".end\n")
if __name__ == '__main__':
    Fib()

```

[实验结果]

优化前后分析：

- 优化前：



- 优化后：

ily0322/lc3web/

0	LDR F
5	HALT
5	ST R0
1	NOP
1	NOP
2	NOP
4	NOP
6	NOP
A	NOP
2	NOP
E	NOP

home.ustc.edu.cn 显示

测试数据 F(3) = 4 你的回答正确, 指令数 3
测试数据 F(24) = 706 你的回答正确, 指令数 3
测试数据 F(144) = 642 你的回答正确, 指令数 3
测试数据 F(456) = 66 你的回答正确, 指令数 3
测试数据 F(1088) = 2 你的回答正确, 指令数 3
测试数据 F(1092) = 290 你的回答正确, 指令数 3
测试数据 F(2096) = 898 你的回答正确, 指令数 3
测试数据 F(4200) = 322 你的回答正确, 指令数 3
测试数据 F(8192) = 514 你的回答正确, 指令数 3
测试数据 F(12000) = 258 你的回答正确, 指令数 3
测试数据 F(14000) = 898 你的回答正确, 指令数 3
平均指令数 3

确定

RunPauseC

☒ Follow PC

web 修改而来, 旨在满足

者机器码 (在左侧的 Asse

开头。

实验, 目前支持显示测试

是自己随便给的, 尽量全

特性: 评测开始时条件码总为 Z, 这与助教标准 (开始时为

选择测试题目

[实验总结]

从此次实验可以看出, 改进一个算法的表现中, 若想优化其执行时间, 可从其需要执行的主要操作入手, 若这些操作不可避免, 或优化空间、价值不大, 就必须改变思路, 另一方面, 本次改进采用空间换效率的思路, 有时效率与空间不可兼得, 必要时要权衡两者的重要性。