

编程作业1设计文档

作者：1853931-郑涵文

01 开发环境

本地环境：VMware Fusion 专业版 11.5.6 + Ubuntu 64位 20.04（需要配备**OpenSSH**工具）

远程服务器：阿里云ECS服务器

02 背景知识

- Linux Shell编程基本语法
 - Linux Vim文本编辑器操作
 - Shell管道方式实现命令
 - Shell带参数脚本编写
 - Linux虚拟机间通信方法
 - 另外用到一些具体的命令：
 - echo
 - uptime
 - sleep
 - cat
 - wc
 - sed
 - cut
 - date
 - bc
-

03 具体题目的分析与解决

题目1 质数求和

题目分析

作业第一题选取了一道简单的算法题，我认为这旨在让我们熟悉Linux下Shell编程的基本语法。另外尽快找到合适的编程环境也很重要，所幸由于我暑假的学习经历，我本身就保留有Ubuntu虚拟机环境，而且经过一暑假的历练我算是比较熟悉Linux的终端操作了，上手比较快。但是这个过程的确也经历过无数语法错误，以及学会适应在终端里编写、调试代码而没有成型的IDE可以依赖，最终是克服了困难，写

出了这十几行代码。

就 `$` 这个符号的使用就让我难受极了！最终一次的bug出在：已定义的变量之后再次被“赋值”的时候，仍然算再次定义而不是使用！也就是一个变量单独出现在赋值号的左边的时候是不需要使用 `$` 的。

另外要注意Shell编程的运算式对于空格的使用的严格要求。

程序截图



```
#!/bin/bash
sum=0
for i in $(seq 2 100); do
  for j in $(seq 2 $i); do
    if [ $(( $i % $j )) -eq 0 ]; then
      break
    fi;
  done
  #echo "i:$i j:$j"
  if [ $i -eq $j ]; then
    sum=$(( $sum + $i ))
    #echo "sum:$sum"
  fi;
done
echo "0-100之间的所有质数之和为$sum">1853931-hw1-q1.log
~
~
~
~
~
~
~/1853931-hw1-q1.sh 15L, 303C 1,1 All
```

程序分析

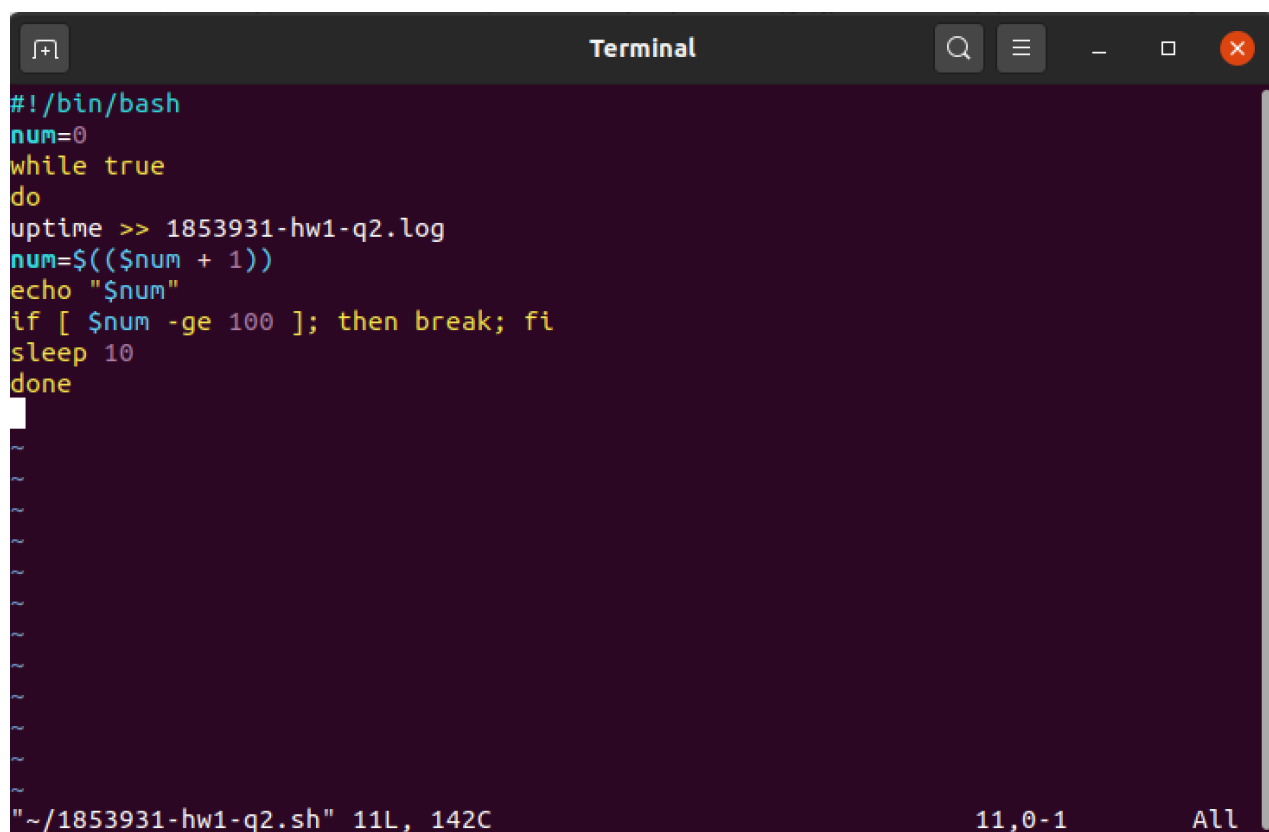
由于仅求100以内的质数和，所以算法比较朴素。两层嵌套的for函数，第一层遍历2-100间的所有整数，第二层对当前的i遍历小于i（大于1）的所有整数，如果有一个整数能整除i则跳出当前for循环；第二层循环外判断j是否等于i，如果相等则证明i是一个质数，将之累加至sum。最终输出sum并重定向至指定文件即可。

题目2 uptime

题目分析

第二题我们接触到了一个很有用的命令：uptime，也是这一道题让我意识到Unix并不能代替Linux来完成作业！我之前试图在Mac OS下的终端里完成作业，第一题非常顺利，因为Mac OS也属于类UNIX操作系统；但是发现第二题的输出中时间的格式只是XX:XX，并不是参考格式中的标准的XX:XX:XX，导致第三题想要转换为时间戳并不容易。所以我才觉悟了去找我的Ubuntu虚拟机！

程序截图

A terminal window titled "Terminal" with a dark background and light-colored text. The window contains a shell script being executed. The script starts with a shebang line, initializes a counter variable, enters a loop, and prints the counter value after each uptime command. The status bar at the bottom shows the file path, line count, column count, and cursor position.

```
#!/bin/bash
num=0
while true
do
uptime >> 1853931-hw1-q2.log
num=$((num + 1))
echo "$num"
if [ $num -ge 100 ]; then break; fi
sleep 10
done
~
~
~
~
~
~
~
~
~
~
~
~/1853931-hw1-q2.sh" 11L, 142C 11,0-1 All
```

程序分析

代码结构也很简单，用一个计数器记录当前输出的次数，然后在每次uptime之后使用sleep命令让进程休眠10秒。注意控制输出行数，整整100次比较整齐。

日志部分截图

```
Open 1853931-hw1-q2.log Save
25 21:37:51 up 21 min, 1 user, load average: 0.02, 0.21, 0.26
26 21:38:01 up 22 min, 1 user, load average: 0.02, 0.21, 0.25
27 21:38:11 up 22 min, 1 user, load average: 0.02, 0.20, 0.25
28 21:38:21 up 22 min, 1 user, load average: 0.01, 0.19, 0.25
29 21:38:31 up 22 min, 1 user, load average: 0.01, 0.18, 0.24
30 21:38:41 up 22 min, 1 user, load average: 0.01, 0.18, 0.24
31 21:38:51 up 22 min, 1 user, load average: 0.01, 0.17, 0.24
32 21:39:01 up 23 min, 1 user, load average: 0.00, 0.17, 0.23
33 21:39:11 up 23 min, 1 user, load average: 0.00, 0.16, 0.23
34 21:39:21 up 23 min, 1 user, load average: 0.00, 0.15, 0.23
35 21:39:31 up 23 min, 1 user, load average: 0.00, 0.15, 0.23
36 21:39:41 up 23 min, 1 user, load average: 0.00, 0.14, 0.22
37 21:39:51 up 23 min, 1 user, load average: 0.00, 0.14, 0.22
38 21:40:01 up 24 min, 1 user, load average: 0.00, 0.13, 0.22
39 21:40:11 up 24 min, 1 user, load average: 0.00, 0.13, 0.21
40 21:40:21 up 24 min, 1 user, load average: 0.08, 0.14, 0.22
41 21:40:31 up 24 min, 1 user, load average: 0.07, 0.14, 0.21
42 21:40:41 up 24 min, 1 user, load average: 0.06, 0.13, 0.21
43 21:40:51 up 24 min, 1 user, load average: 0.05, 0.13, 0.21
44 21:41:02 up 25 min, 1 user, load average: 0.04, 0.12, 0.21
45 21:41:12 up 25 min, 1 user, load average: 0.03, 0.12, 0.20
46 21:41:22 up 25 min, 1 user, load average: 0.03, 0.11, 0.20
47 21:41:32 up 25 min, 1 user, load average: 0.02, 0.11, 0.20
48 21:41:42 up 25 min, 1 user, load average: 0.02, 0.10, 0.19
49 21:41:52 up 25 min, 1 user, load average: 0.02, 0.10, 0.19
50 21:42:02 up 26 min, 1 user, load average: 0.01, 0.10, 0.19
51 21:42:12 up 26 min, 1 user, load average: 0.01, 0.09, 0.18
Plain Text Tab Width: 8 Ln 1, Col 1 INS
```

题目3 文件统计

题目分析

这道题属于最繁杂的一道。甚至将题目2的日志文件名字作为一个参数传给本题的脚本这个需求都是我最后快要交作业时才发现的需求！

首先定义一个变量，将 \$1 赋值与之，其含义是：运行脚本时，该变量会被赋上脚本名后紧跟的第一个参数的值。

前两问用到 **wc** (word count) 这个命令，参数 -l 用于输出指定文件的总行数，参数 -c 用于输出指定文件的总字符数。

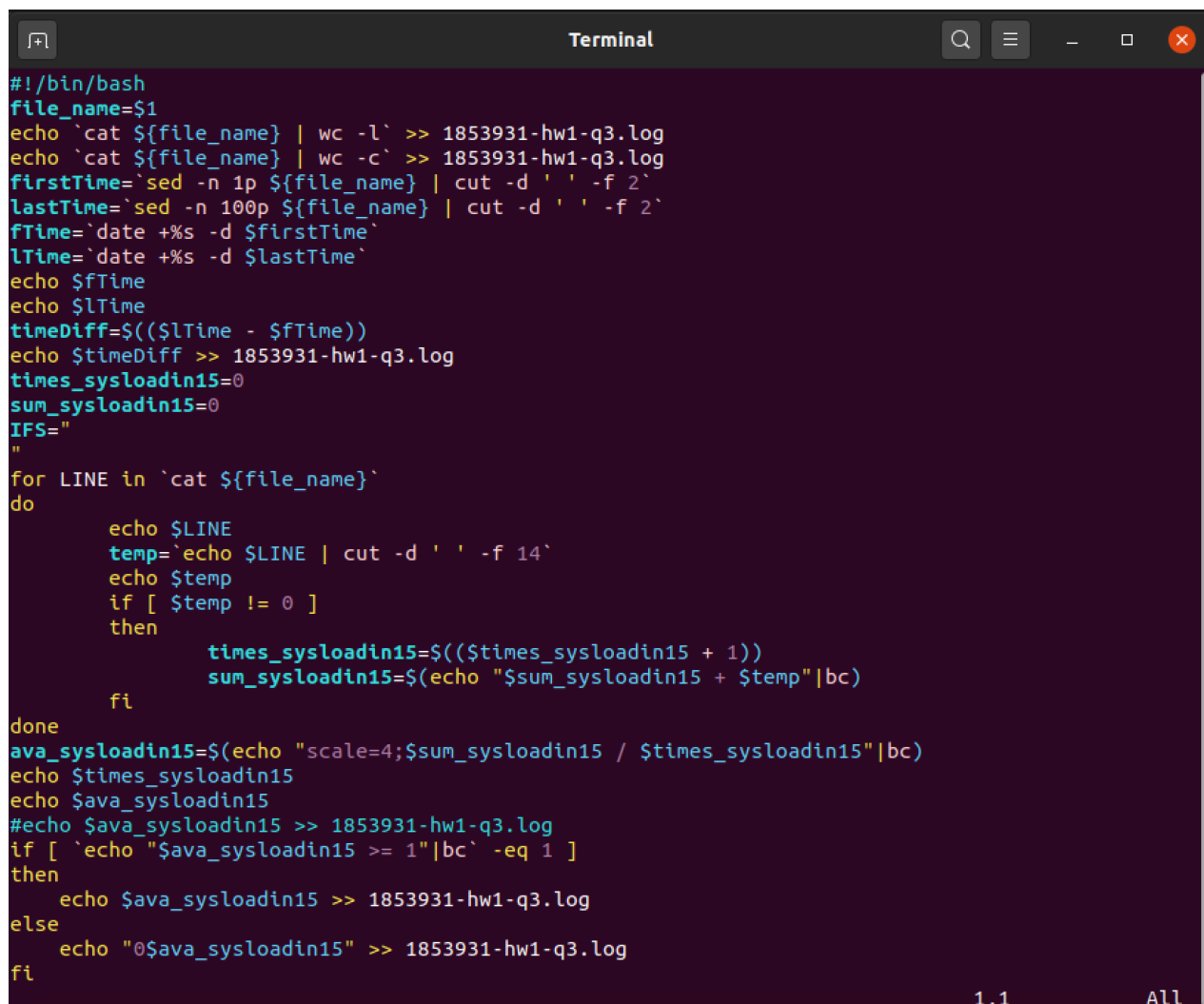
接下来第三问，对于取出某文件的指定行，我本来想用最直观的 head 和 tail 命令，但是没想到在我的环境下一直无法识别 head 这条命令，最后我只好作罢，选择了 **sed** 这条命令，取出第 1 行和第 100 行。另外用到了非常顺手的 **cut** 命令，将 uptime 的一行输出用 **空格** 进行切割是针对本问题非常简便的一种做法。然后使用 **date** 将系统时间格式化成时间戳形式再作差即可。

另外！本题的输出重定向要用到 >> 来在日志文件里追加输出，只用一个 > 的话会覆盖之前的文件！

对于第四问我将**IFS**定义为了回车，即在for循环中使用**LINE**得到的是用回车判断的行。然后在对小数做运算时要用到“**|bc**”（basic calculator/bench calculator）的写法，这是用管道方式简便的使用bc数学模块；否则会报“integer expected”的错。

最后我为了格式化输出，由于bc的除法结果如果是小于1的小数就会以“.XXXX”这样的格式输出，所以我通过一个判断为这样的情况前加了一个“0”（亦是為了后续方便）。

代码截图

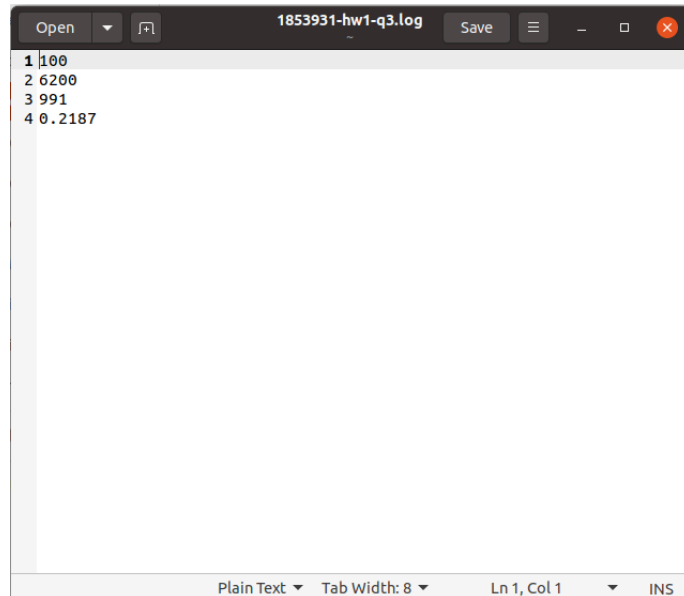


```
#!/bin/bash
file_name=$1
echo `cat ${file_name} | wc -l` >> 1853931-hw1-q3.log
echo `cat ${file_name} | wc -c` >> 1853931-hw1-q3.log
firstTime=`sed -n 1p ${file_name} | cut -d ' ' -f 2`
lastTime=`sed -n 100p ${file_name} | cut -d ' ' -f 2`
fTime=`date +%s -d $firstTime`
lTime=`date +%s -d $lastTime`
echo $fTime
echo $lTime
timeDiff=$((lTime - fTime))
echo $timeDiff >> 1853931-hw1-q3.log
times_sysloadin15=0
sum_sysloadin15=0
IFS="
"
for LINE in `cat ${file_name}`
do
    echo $LINE
    temp=`echo $LINE | cut -d ' ' -f 14`
    echo $temp
    if [ $temp != 0 ]
    then
        times_sysloadin15=$((times_sysloadin15 + 1))
        sum_sysloadin15=$(echo "$sum_sysloadin15 + $temp"|bc)
    fi
done
ava_sysloadin15=$(echo "scale=4;$sum_sysloadin15 / $times_sysloadin15"|bc)
echo $times_sysloadin15
echo $ava_sysloadin15
#echo $ava_sysloadin15 >> 1853931-hw1-q3.log
if [ `echo "$ava_sysloadin15 >= 1"|bc` -eq 1 ]
then
    echo $ava_sysloadin15 >> 1853931-hw1-q3.log
else
    echo "0$ava_sysloadin15" >> 1853931-hw1-q3.log
fi
```

程序分析

主要流程见上题目分析部分。其他一些细节：使用**cat**命令访问文件，管道方式使用**wc**命令并将输出重定位至指定文件。对于我们需要的uptime输出的最后一项“15分钟内系统负载”在用空格截取后的序号是14这个事我也是通过调试确定的，自己数永远数不对。最后要注意bc的正确使用。

日志截图



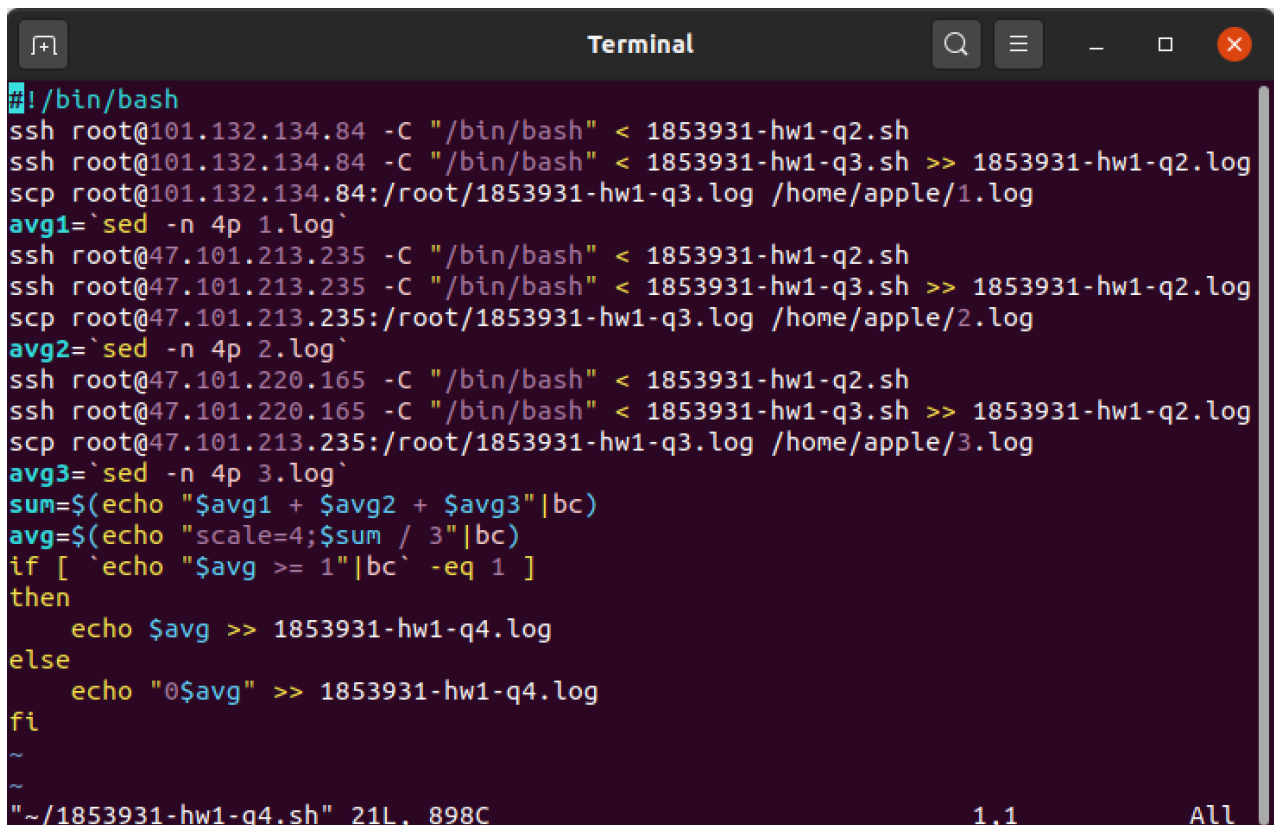
```
1 100
2 6200
3 991
4 0.2187
```

题目4 虚拟机通信

题目分析

这一题我的解决方案是使用阿里云ECS服务器。我加上我的两个舍友刚好三人，刚好满足三个远程虚拟机的条件，非常幸运。针对本题我的解决方案是使用ssh命令来访问远端服务器，在远端服务器上运行我的本地文件1853931-hw1-q2.sh和1853931-hw1-q3.sh，然后用scp命令将远端服务器上的1853931-hw1-q3.log文件安全地拷贝到本地——这样的目的是简化后续操作，不再需要访问远端服务器。这样得到了1.log、2.log和3.log三个远程虚拟机运行的日志文件，再分别取得第四行进行求平均的运算即可。

代码截图



```
#!/bin/bash
ssh root@101.132.134.84 -C "/bin/bash" < 1853931-hw1-q2.sh
ssh root@101.132.134.84 -C "/bin/bash" < 1853931-hw1-q3.sh >> 1853931-hw1-q2.log
scp root@101.132.134.84:/root/1853931-hw1-q3.log /home/apple/1.log
avg1=`sed -n 4p 1.log`
ssh root@47.101.213.235 -C "/bin/bash" < 1853931-hw1-q2.sh
ssh root@47.101.213.235 -C "/bin/bash" < 1853931-hw1-q3.sh >> 1853931-hw1-q2.log
scp root@47.101.213.235:/root/1853931-hw1-q3.log /home/apple/2.log
avg2=`sed -n 4p 2.log`
ssh root@47.101.220.165 -C "/bin/bash" < 1853931-hw1-q2.sh
ssh root@47.101.220.165 -C "/bin/bash" < 1853931-hw1-q3.sh >> 1853931-hw1-q2.log
scp root@47.101.213.235:/root/1853931-hw1-q3.log /home/apple/3.log
avg3=`sed -n 4p 3.log`
sum=$(echo "$avg1 + $avg2 + $avg3"|bc)
avg=$(echo "scale=4;$sum / 3"|bc)
if [ `echo "$avg >= 1"|bc` -eq 1 ]
then
    echo $avg >> 1853931-hw1-q4.log
else
    echo "0$avg" >> 1853931-hw1-q4.log
fi
~
~
~/1853931-hw1-q4.sh" 21L, 898C 1,1 All
```

程序分析

主要流程见上题目分析部分。注意远端虚拟机运行1853931-hw1-q3.sh时传参的格式为：

```
ssh root@101.132.134.84 -C "/bin/bash" < 1853931-hw1-q3.sh >> 1853931-hw1-q2.log
```

另外，101.132.134.84、47.101.213.235、47.101.220.165分别是我们三人的ECS服务器ip地址；测试时需要我们三个处于同一个局域网下。

● ECS服务器 已创建 | 已运行

ECS公网地址：101.132.134.84 

ECS登录名：root 


登录密码：lp3Le6No2D 


ECS实例ID：i-uf6ivitz6d5nq... 


IP白名单：101.84.198.139 

● ECS服务器 已创建 | 已运行

ECS公网地址：47.101.213.235 

ECS登录名：root 

登录密码：lb7Mi1Qz8L 

ECS实例ID：i-uf6291chj48g1... 

IP白名单：180.160.39.230 

● ECS服务器 已创建 | 已运行

ECS公网地址: 47.101.220.165

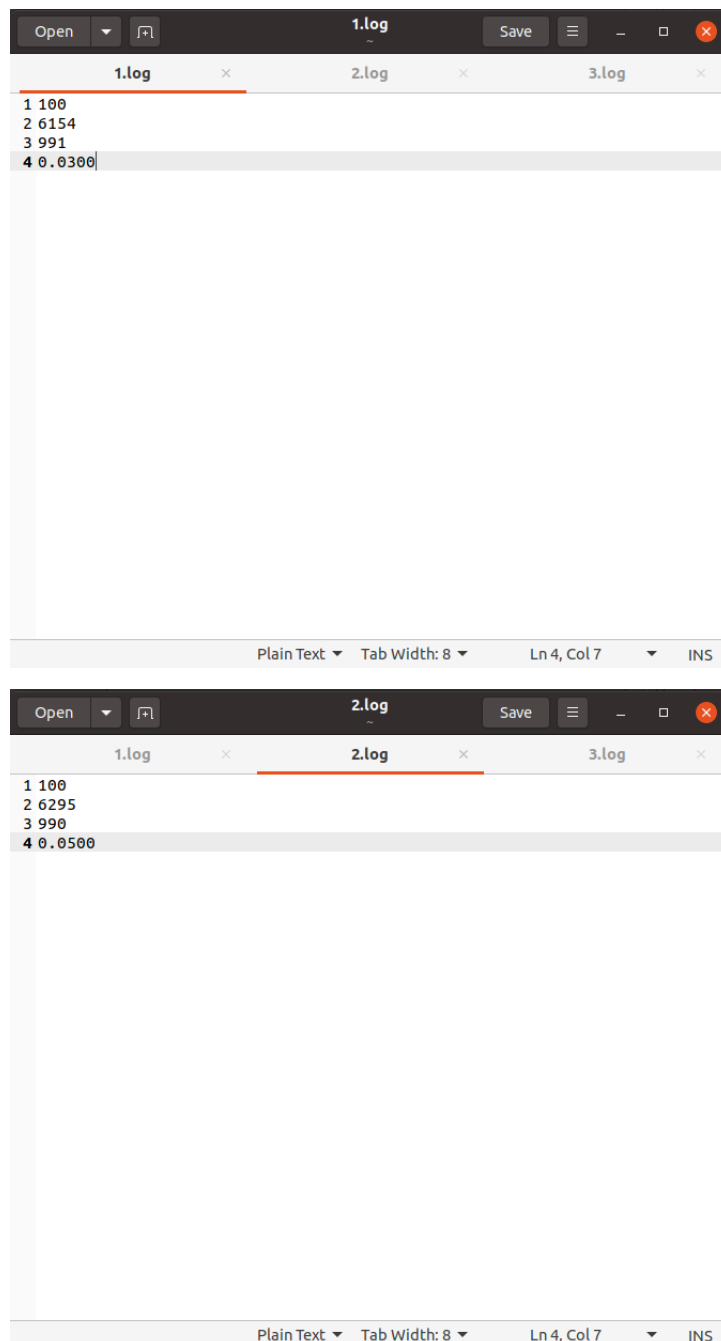
ECS登录名: root

登录密码: Ny0Jf4Ee6U

ECS实例ID: i-uf6i8o07kxftdg...

IP白名单: 58.41.204.78

日志截图



Open

3.log

Save

1.log

2.log

3.log

1 100

2 6300

3 990

4 0.0507

Plain Text

Tab Width: 8

Ln 4, Col 7

INS

Open

1853931-hw1-q4.log

Save

1 0.0435

Plain Text

Tab Width: 8

Ln 1, Col 7

INS