## 案例十一：人员分组

需求背景是这样的，我的培训是分班级的，班级里有几十人。第二阶段结束后会有一个作业安排，此时需要把班级里所有同学划分为若干小组。最早我的方法是按照同学们的QQ号尾号去划分小组，由于人数太少最终导致划分不均衡，所以就想到写一个shell脚本，把同学们随机分组。具体需求如下：

1）假设全班级一共有72人（提供一个人员名单，名字为汉语拼音）

2）要划分7个小组

3）要保证人员分布小组的随机性，即每次执行脚本的结果应该是不一样的

知识点一：产生随机数

在Linux系统中有个内置变量RANDOM，它其实就是一个随机数，用法如下：

# echo $RANDOM

32419

这个RANDOM变量取值范围为0-65535，如果想要获取一个更长随机数，可以通过多个RANDOM相叠加实现，例如：

# echo $RANDOM$RANDOM

297672691

第二个方法就是使用date命令，date有一个%N，可以打印当前时间的秒，单位是纳秒

# date +%N

708277533

第三个方法是通过读取系统的特殊设备/dev/urandom，如下：

# head -100 /dev/urandom|cksum

2639672374 25726

这个/dev/urandom是一个产生随机数的特殊设备，是二进制的，所以不能直接cat或者head，但是可以加管道把结果交给cksum。cksum是一个校验文件或者字符的工具，它会产生两个数字串。由于管道前面的字符串是随机的，所以这个数字串自然也是随机的。我们只需要把第一个数字串拿到即可，所以还需要用awk处理一下：

# head -100 /dev/urandom|cksum|awk '{print $1}'

3240257867

当然，cksum命令也可以计算一个字符串的校验值从而得到随机数，如下：

# echo "zhangsan"|cksum |awk '{print $1}'

3244387227

知识点二：打乱顺序

由于本案例要求随机性，所以给出的人员名单最好是给它打乱顺序，这样可以保证每次执行脚本都有一定的随机性，可以用shuf命令实现：

# seq 1 5 |shuf

5

3

4

1

2

# seq 1 5 |shuf

2

3

4

1

5

每次都会得到一个不一样的序列。也可以直接作用于文件，会把文件的行顺序打乱：

# shuf /etc/passwd |head -2

php-fpm:x:505:505::/home/php-fpm:/sbin/nologin

operator:x:11:0:operator:/root:/sbin/nologin

# shuf /etc/passwd |head -2

root:x:0:0:root:/root:/bin/bash

shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown

注意，shuf命令仅仅是把行顺序打乱，并不会改动行内字符的位置。

知识点三：shell中数字四舍五入

shell中的除法运算会直接把小数舍弃了，比如1.9直接为1，这样很明显不合理。下面的方法可以实现四舍五入。

#!/bin/bash

#以下函数针对除法运算，并且限定$1为除数，$2为被除数

div()

{

n=`echo "scale=1;$1/$2"|bc`

n1=`echo "scale=1;$n+0.5"|bc`

echo $n1|cut -d. -f1

}

div 10 3

本案例测试人员名单

# cat member.txt

xiaoguisheng

guoyuqing

xiongyongzheng

mengjintang

chaizuzhou

zhousheng

xufangming

zhaoliangyun

hanshiru

wangxianyi

zhangjipei

luxiuli

yangshugen

guoyongzhi

lijianguo

wuqiongchen

dinglin

yaoyashan

yinzijia

wangbencheng

liuxiuwen

chenzuqi

leyuguo

baozongyao

fenghao

sunxiaoquan

zhangyaxian

lijiuzhe

dulichun

lixi

shenpeiwen

zousilin

luoping

chaiyan

fandaozhang

huzixiang

jinzhen

zhujunfeng

liqianbiao

hangyanliang

luorenjian

loujianji

fujianzhou

gengyiwu

jinjigui

liuzhizhong

lisanyan

lisili

zhangyiyu

songguozhen

zhangxinghua

zhaozhiyong

huanghe

xiaojie

fanhongfei

wangguiwen

renshumin

songfuying

zhanghaibo

liguangqun

puaihua

yanzhihua

gaojixian

liulai

funing

chenruizhi

chendaxin

laishaoying

xujian

xiaozhekou

xuxiaping

jiangchunqing

本案例参考脚本

#!/bin/bash

#给人员划分小组

#作者：

#日期：2018-10-11

#版本：v1.0

#人员列表文件

f=member.txt

#小组数

group\_n=7

#人员总数

member\_n=`wc -l $f|awk '{print $1}'`

#根据姓名计算该用户所在小组的id

get\_n()

{

#根据姓名计算cksum值

l=`echo $1|cksum|awk '{print $1}'`

#获取一个随机数

n1=$RANDOM

#cksum值和随机数相加，然后除以小组数取余，这样可以确保每次获取到的余数都不一样

n2=$[$n1+$l]

g\_id=$[$n1%$group\_n]

#假如小组数为7，则余数范围0-6，如果余数为0，则小组为7

if [ $g\_id -eq 0 ]

then

g\_id=$group\_n

fi

echo $g\_id

}

for i in `seq 1 $group\_n`

do

#n\_$i.txt为临时文件，用来记录该小组内的成员

#脚本之前执行过，则该文件会存在，本次执行脚本前应该删除掉这个临时文件

[ -f n\_$i.txt ] && rm -f n\_$i.txt

done

shuf $f|while read name

do

#计算用户所在小组的id

g=`get\_n $name`

#将人员追加写入到他对应的小组里

echo $name >> n\_$g.txt

done

#定义计算文件行数的函数

nu(){

wc -l $1|awk '{print $1}'

}

#获取组员人数最多的小组

max(){

ma=0

for i in `seq 1 $group\_n|shuf`

do

n=`nu n\_$i.txt`

if [ $n -gt $ma ]

then

ma=$n

fi

done

echo $ma

}

#获取组员人数最少的小组

min(){

mi=$member\_n

for i in `seq 1 $group\_n|shuf`

do

n=`nu n\_$i.txt`

if [ $n -lt $mi ]

then

mi=$n

fi

done

echo $mi

}

#定义四舍五入函数

div()

{

n=`echo "scale=1;$1/$2"|bc`

n1=`echo "scale=1;$n+0.5"|bc`

echo $n1|cut -d. -f1

}

#小组组员平均值（非四舍五入）

ava\_n=$[$member\_n/$group\_n]

#小组组员平均值（四舍五入）

ava\_n1=`div $member\_n $group\_n`

#如果平均值第一位小数小于5，则说明最终分组结果中大多数组成员数等于$ava\_n

#所以此时判断条件里比较对象为最小组更合理

if [ $ava\_n -eq $ava\_n1 ]

then

#定义初始最小值

ini\_min=1

#以下while循环要做的事情，就是要把人数多的组里的人搞到人数少的组里去

#此while循环的条件是，当人数最少的组成员数小于组员平均值

while [ $ini\_min -lt $ava\_n1 ]

do

#找出人数最多的组

m1=`max`

#找出人数最少的组

m2=`min`

for i in `seq 1 $group\_n|shuf`

do

n=`nu n\_$i.txt`

#找到人数最多的组对应的文件f1（可能有多个，这里取出现的第一个即可）

if [ $n -eq $m1 ]

then

f1=n\_$i.txt

#找到人数最少的组对应的文件f2（可能有多个，这里取出现的第一个即可）

elif [ $n -eq $m2 ]

then

f2=n\_$i.txt

fi

done

#取f1中最后一个人名

name=`tail -n1 $f1`

#将这个人名追加写入f2中

echo $name >> $f2

#在f1中删除刚刚取走的人名

sed -i "/$name/d" $f1

#把此时的最少组人员数赋值给ini\_min

ini\_min=`min`

done

else

#如果平均值第一位小数大于5，则说明最终分组结果中大多数组成员数大于$ava\_n

#所以此时判断条件里比较对象为最大组更合理

#定义初始最大值

ini\_max=$member\_n

while [ $ini\_max -gt $ava\_n1 ]

do

#找出人数最多的组

m1=`max`

#找出人数最少的组

m2=`min`

for i in `seq 1 $group\_n|shuf`

do

n=`nu n\_$i.txt`

#找到人数最多的组对应的文件f1（可能有多个，这里取出现的第一个即可）

if [ $n -eq $m1 ]

then

f1=n\_$i.txt

#找到人数最少的组对应的文件f2（可能有多个，这里取出现的第一个即可）

elif [ $n -eq $m2 ]

then

f2=n\_$i.txt

fi

done

#取f1中最后一个人名

name=`tail -n1 $f1`

#将这个人名追加写入f2中

echo $name >> $f2

#在f1中删除刚刚取走的人名

sed -i "/$name/d" $f1

#把此时的最少组人员数赋值给ini\_min

ini\_max=`max`

done

fi

for i in `seq 1 $group\_n`

do

echo -e "\033[34m$i 组成员有：\033[0m"

cat n\_$i.txt

#把临时文件删除

rm -f n\_$i.txt

echo

done