Shell 脚本程序设计

第一题：生成 TCP 活动状况报告

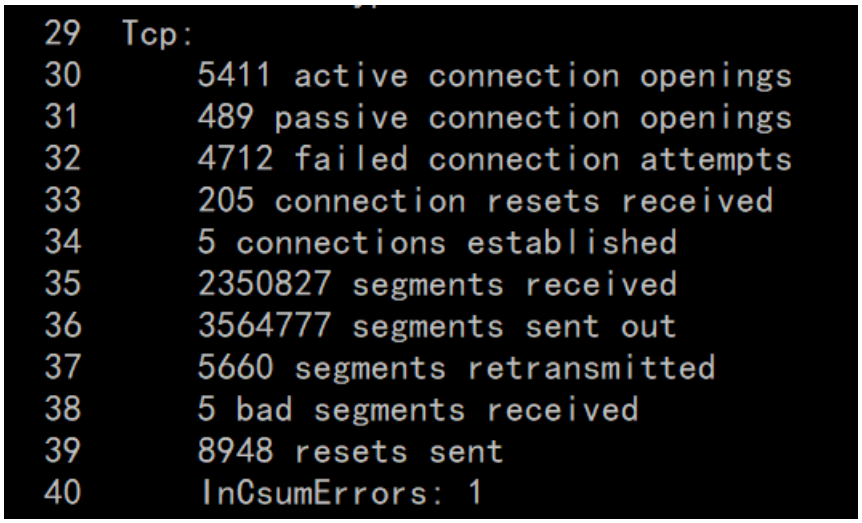
1.题目描述

netstat --statistics 命令可以列出 tcp 等协议的统计信息。编写 shell 脚 本程序，每隔 1 分钟生成 1 行信息：当前时间；这一分钟内 TCP 发送了多少报文； 接收了多少报文；收发报文总数；行尾给出符号+或-或空格（+表示这分钟收 发报文数比上分钟多 10 包以上，差别在 10 包或以内用空格，否则用符号-）。

2.基本思路

2.1 定位数据

首先通过这个 netstat –statistics|cat -n 可以知道我们关注的 tcp 数据在那一行



我们仅仅需要直达第 35 行收到的 segments 以及第 36 行发送的 segments 即可，然后通过 awk “NR==35”和 awk “NR==36”即可获取

2.2 提取数据

我先将两行提取出来分别存在两个变量中，然后在提取其中的数字分别代表 receive 和 send 的 segments 数

```
str1=`netstat -s|awk "NR==35"`
str2=`netstat -s|awk "NR==36"`
rec_before=`echo $str1|grep -o '[0-9]*'`
send_before=`echo $str2|grep -o '[0-9]*'`
```

当然这个也有简单写法：rec_before=`netstat -s|awk "NR==35"|grep -o '[0-9]*'` 对于 send_before 同理

另外我们目标是知道一分钟之类收发的 tcp 分段数，为此我们需要记录一分钟之后的以上数据，sleep 60 即可，记录如下

```
sleep 60
str1=`netstat -s|awk "NR==35"`
str2=`netstat -s|awk "NR==36"`
rec_after=`echo $str1|grep -o '[0-9]*'`
send_after=`echo $str2|grep -o '[0-9]*'`
```

2.3 计算收发总数

然后就是简单计算，得到收发 segments 的总数

```
rec=`expr $rec_after - $rec_before`
send=`expr $send_after - $send_before`
rec_send=`expr $rec + $send`
```

2.4 比较并标记

如果这一分钟收发数比上一分钟多 10，则将 label 记为‘+’，少 10 则将 label 记为‘-’，否则 label 记为‘ ’

```
if [ $flag -ne 0 ];then
    if [ $rec_send -ge $rec_send_last ];then
        d=`expr $rec_send - $rec_send_last`
        if [ $d -gt 10 ];then
            label="+"
        fi
    elif [ $rec_send -le $rec_send_last ];then
        d=`expr $rec_send_last - $rec_send`
        if [ $d -gt 10 ];then
            label="-"
        fi
    fi
fi
flag=1
rec_send_last=`expr $rec_send`   # 记下上一分钟的收发总数
```

2.5 结果输出

最后格式化输出结果

```
printf "%s %02d:%02d %8d %8d %8d %s\n" $date $hour $minute $rec $send $rec_send $label
```

源代码见附录

3.实际效果

```
u104@LiSongYang:~/10_ch_shell_control$ ./tcp1.sh
2020-05-06 00:01      819     1485     2304
2020-05-06 00:02      812     1498     2310
2020-05-06 00:03      762     1461     2223 -
2020-05-06 00:04      801     1449     2250 +
2020-05-06 00:05      826     1443     2269 +
2020-05-06 00:06      836     1486     2322 +
2020-05-06 00:07      827     1466     2293 -
2020-05-06 00:08      840     1476     2316 +
2020-05-06 00:09      864     1468     2332 +
2020-05-06 00:10      862     1501     2363 +
2020-05-06 00:11      834     1463     2297 -
2020-05-06 00:12      850     1471     2321 +
2020-05-06 00:13      857     1503     2360 +
2020-05-06 00:14      853     1491     2344 -
2020-05-06 00:15      835     1473     2308 -
2020-05-06 00:16      860     1482     2342 +
```

第二题：下载 bing 图库中图片

1.题目描述

访问 https://bing.ioliu.cn/?p=23 可以看到 bing 图库第 23 页的内容（见下一 PPT 中的图 片），这个 Web 页有多个图片小样，将鼠标放到某个小样上，如右上角，可见中文说明信息 “野花草甸上的一只欧亚雕鸮，德国莱茵兰-普法尔茨”和日期信息 2019-08-03，点击一下， 此图片就可以下载。 编写脚本程序 bing.sh，将这个图库中照片全部下载下来存放到本地 bing 目录中，上面 URL 中 p=23 可以换成 p=1 到 p=126 可访问 126 个页，每页有 12 个图，每个图的日期，中文说明信息和 下载地址及文件名 html 文件中可提取。要求下载后的文件命名为“日期 中文说明.jpg”例如： 2019-08-03 野花草甸上的一只欧亚雕鸮，德国莱茵兰-普法尔茨.jpg

2.要求（含选做）

■ 命令行参数

◆ ./bing.sh 后面可以跟两个参数，通过指定页号区间限定下载范围，没有参数时页号区间为 1-126

◆要允许多个程序并发，例如：一个终端上启动 ./bing.sh 1 63，另一个终端上启动./bing.sh 64 126，这样在两个终端上同时下载，以加快任务完成的速度。

■ 不重复下载已下载的图片

◆检查图片是否已下载，如果已下载，则不再下载，这样在一定程度上支持批量任务在被中断后可以从断点继续

■ 考虑下载文件出现故障的情况

◆如果一个图片有 5MB，接收 1.5MB 后网络断开，则残存一个不完整的图片文件。避免这种现象发生的一种方法是，wget 下载时使用一个临时文件名。判断 wget 是否成功，若成功则将文件改名为正式名称；若失败，删除临时文件。临时文件名的选取要考虑前述的并发问题，至少不可以固定一个名字导致两进程的争夺。

■ 获取更多图片

对已设计好的 bing.sh 进一步扩充，允许下面的命令行参数： ./bing.sh rand 500 实现的功能为：执行 500 次随机获取。每次成功的随机获取会得到一个图片，检查一下 这个图片是否本地已存在。如果已存在，丢弃，否则保存。 访问 https://bing.ioliu.cn/v1/rand?type=json 可得到一个 json 数据，每次 得到的内容是随机的，其中含有图片的日期、说明信息和获取它的 url 网址。新获得的 文件命名方式与以前的程序相同。要求：文件名不同但是内容完全相同的图片丢弃，例 如，下面两个文件虽然名字不同，但是内容是一样的，只保留其中一个文件。 2019-05-03 Ruff male displaying its plumage, Varanger Peninsula, Norway.jpg 2019-05-02 挪威瓦朗厄尔半岛上一只展示翎颌的雄性涉禽.jpg

3.基本思路

3.1 解析命令参数

首先解析命令行参数，如果参数个数为 0，就默认下载 page=1~126 的图片；如果参数个数是 2，两个参数如果是 rand num1，则进行随机下载图片，此过程进行相关的判重；两个参数如果是 num1 num2，则下载 page=num1~num2 的图片；其余情况都是输入错误，提示相关信息。

```
if [ $# = 0 ];then  # 没有参数就是默认情况
    echo "default args[1 126]..."
elif [ $# -eq 1 -o $# -gt 2  ];then # 一个参数或者多于两个参数
    echo "Usage: $0 : [num1 num2]/[rand num1]"
    exit 1
elif [ $# = 2 -a $1 = "rand" ];then # rand num1
md5sum small/*.jpg > jpg.md5
for i in `seq 1 $2`
do ...
done
    rm jpg.md5 tmp.txt
    exit 0
else    # num1 num2
    expr $1 + 6 &> /dev/null    # 判断$1是否为整数
    if [ $? -ne 0 ];then ...
    fi
    page_min=$1
    page_max=$2
fi
```

3.2 下载网页.html

下载 page=num1~num2 的图片，首先需要获取网页信息，我将他们存在 html/目录下面，每得到一个网页，我在网页中找到每个图片的日期 date、中文说明 name 以及图片地址 URL

```
wget -O html/${i}.html https://bing.ioliu.cn/?p=$i -nc -q
```

3.3 获取图片信息

3.3.1 分别获取图片信息

1）获取中文说明

对于 name，由于 name 中可能含有空格，将这个信息和日期、网址分开处理，我是先将每个网页中的 12 个名字提取出来存在一个数组中，

```
declare -a arr # 声明数组存放name
index=0
for row in `seq 1 12`
do
    arr[$index]=`cat html/$i.html| sed -e 's/<[^<>]*>/\n\0/g' -e 's/<h3>//g'| grep ©|sed -e 's/ (.*)//g'|awk "NR==$row"`
    let "index+=1"
done
index=0
```

其中某个网页提取出的 name 如下：

```
u104@LiSongYang: /down$ cat html/1.html| sed -e 's/<[^<>]*>//\n\0/g' -e 's/<h3>//g'| grep ©|sed -e 's/ (.*)//g'
库斯科附近萨克塞华曼的印加要塞，秘鲁
正在照看花草的孩子和祖父铜雕，西班牙科尔多瓦
乌尤尼盐沼，玻利维亚
琼斯海滩的雪鸮，纽约长岛
Kalaat M'Gouna的古堡遗址，摩洛哥
达恩附近普法尔茨森林中的Altdahn城堡，德国莱茵兰-普法尔茨(Dahn Rockland)，Palatinate Forest，Rhineland-Palatinate，Germany
极北朱顶雀的巢，芬兰拉普兰
普尔曼附近的帕卢斯一辆拖拉机在耕作时扬起尘土，华盛顿州
索尔兹伯里大教堂与放牧的羊群，英格兰
一只南美貘幼崽小跑着穿过草地
红宝石海滩的日落，华盛顿州奥林匹克国家公园
福克兰群岛上的南跳岩企鹅
```

2）获取日期和网址

对于 date 和 URL，和 name 类似，找到相关标志在网页提取中即可，为此需要 awk 文件提取

```
u104@LiSongYang:~/down$ vi file.awk
  1 /<em/ { date=$4;}
  2 { url_left="https://bing.ioliu.cn";}
  3 /download  href/ {
  4     url_right=$6;
  5     printf("%s|%s%s\n", date, url_left, url_right);
  6 }
```

比如某个网页的日期和 URL 按照下面的格式显示

```
u104@LiSongYang:~/down$ cat html/1.html -n| sed -e 's/<[^<>]*>/\n\0/g' -e 's/>/ /g' -e 's/"
 -f file.awk
2020-05-06|https://bing.ioliu.cn/photo/SiegeofCusco_ZH-CN9108219313?force=download
2020-05-05|https://bing.ioliu.cn/photo/CordovanCourts_ZH-CN8989880218?force=download
2020-05-04|https://bing.ioliu.cn/photo/LastJedi_ZH-CN8789881870?force=download
2020-05-03|https://bing.ioliu.cn/photo/LaughingOwl_ZH-CN8548558025?force=download
2020-05-02|https://bing.ioliu.cn/photo/KasbahRoses_ZH-CN8429310380?force=download
2020-05-01|https://bing.ioliu.cn/photo/BurgAltdahn_ZH-CN8281669977?force=download
2020-04-30|https://bing.ioliu.cn/photo/ArcticRedpoll_ZH-CN7968973967?force=download
2020-04-29|https://bing.ioliu.cn/photo/PalouseSpring_ZH-CN6803103328?force=download
2020-04-28|https://bing.ioliu.cn/photo/SalisburyCathedral_ZH-CN6366350896?force=download
2020-04-27|https://bing.ioliu.cn/photo/SouthAmericanTapir_ZH-CN6151058361?force=download
2020-04-26|https://bing.ioliu.cn/photo/RubySunset_ZH-CN5545965519?force=download
2020-04-25|https://bing.ioliu.cn/photo/FalklandRockhoppers_ZH-CN5370686595?force=download
```

提取这些信息之后存在变量 str 中，然后进行分割字符串的操作（**这个略显复杂，但是知识点颇丰，下面我再介绍一种不用分割字符串的操作，很简练**）其中分割字符串主要就是用 array1=(${string//|/ })操作替换之后变成数组，然后一个一个存储，这里有两次分割，一次是对于整体 str 的分割成每张图片的日期和网址信息，然后是对于该图片信息的分割提取日期和网址分别存在 date 和 url 中，主要就是下面两个 for 循环

```
for var1 in ${str[@]}
do
    array1=(${var1//|/ })
    date=""
    name=""
    url=""
    for var2 in ${array1[@]}
    do ...
    done
```

2.3.2 整体获取图片全部信息

首先需要将某个网页中的每个图片的日期、中文说明和网址提取出来（使用 sed | grep | awk 各种操作），总的命令如下：（将提取出的信息重定向到 up.txt 文件而不是像上面那样存在 str 变量中）

```
cat html/$i.html -n| sed -e 's/<[^<>]*>/\n\0/g' -e 's/>/ /g' -e 's/"/ /g' -e 's/([^()]*)//g'| grep -E 'download    href|<h3|[^0][0-9]{4}-[0-9]{2}-[0-9]{2}'|awk -f up.awk |sed -e 's/^ *//g' > up.txt
```

为此需要的 awk 文件如下

```
u104@LiSongYang:~/down$ vi up.awk
 1 /<h3/ { $1=""; name=$0; }
 2 /<em/ { date=$4;}
 3 { url_left="https://bing.ioliu.cn";}
 4 /download  href/ {
 5     url_right=$6;
 6     print date;
 7     print name;
 8     printf("%s%s\n",url_left,url_right);
 9 }
```

将所有信息按照如下格式（日期、中文说明、网址每个占一行）输出

```
2020-05-06
库斯科附近萨克塞华曼的印加要塞，秘鲁
https://bing.ioliu.cn/photo/SiegeofCusco_ZH-CN9108219313?force=download
2020-05-05
正在照看花草的孩子和祖父铜雕，西班牙科尔多瓦
https://bing.ioliu.cn/photo/CordovanCourts_ZH-CN8989880218?force=download
2020-05-04
乌尤尼盐沼，玻利维亚
https://bing.ioliu.cn/photo/LastJedi_ZH-CN8789881870?force=download
2020-05-03
琼斯海滩的雪鸮，纽约长岛
https://bing.ioliu.cn/photo/LaughingOwl_ZH-CN8548558025?force=download
2020-05-02
Kalaat M'Gouna的古堡遗址，摩洛哥
https://bing.ioliu.cn/photo/KasbahRoses_ZH-CN8429310380?force=download
2020-05-01
达恩附近普法尔茨森林中的Altdahn城堡，德国莱茵兰-普法尔茨，Palatinate Forest, Rhineland-Palatinate, Germany
https://bing.ioliu.cn/photo/BurgAltdahn_ZH-CN8281669977?force=download
```

然后使用 while read line 机制，读取上面的 up.txt 文件

```
while read date
do
    read name
    read url
    wget -O small/"${date} ${name}.jpg.tmp" $url -q -nc
    ret=`echo $?`
    if [ $ret -eq 0 ];then
        mv small/"${date} ${name}.jpg.tmp" small/"${date} ${name}.jpg"
        echo "$date ${name}.jpg"
    else
        echo "$date ${name}.jpg...failed, $ret"
        rm small/"${date} ${name}.jpg.tmp"
    fi
done < up.txt
```

这里有一个很大的坑，就是如果不用中间文件 up.txt 而直接重定向的话，比如：`cat test1 | while read line;do echo $line;done` 这个会开启一个子 shell（具体的坑请看总结部分）
源代码见附录

3.4 下载图片

然后提取之后就下载图片，这里首先是下载一个临时文件（.tmp），**由于每个页面的图片的名字不同，所以可以根据这个判重，使用 wget 的-nc 选项就不会下相同的图片，但是对于选做部分需要使用 md5sum 来比较新下载的图片和已有图片的区别来判重，**最后如果 wget 下载成功（$?=0）将图片重新命名为.jpg 格式，否则删除中间文件

```
wget -O small/"${date} ${name}.jpg.tmp" $url -q -nc
ret=`echo $?`
if [ $ret -eq 0 ];then
    mv small/"${date} ${name}.jpg.tmp" small/"${date} ${name}.jpg"
    echo "$date ${name}.jpg"
else
    echo "$date ${name}.jpg...failed, $ret"
    rm small/"${date} ${name}.jpg.tmp"
fi
```

3.5 并发处理

Linux shell 中多进程并发通过&命令进行后台挂起即可，对应的我在每个页面中对 12 张图片的下载可以并发进行，即在 for 循环下载 12 张图片的时候进行后台挂起

```
for var1 in ${str[@]}
do
{
    array1=(${var1//|/ })
    date=""
    name=""
    url=""
    for var2 in ${array1[@]}
    do
        if [ "$date" = "" ];then
            date=$var2
        elif [ "$url" = "" ];then
            url=$var2
        fi
    done
    name=${arr[$index]}
    echo "$name"
    wget -O small/"${date} ${name}.jpg.tmp" $url -q -nc
    ret=`echo $?`
    if [ $ret -eq 0 ];then
        mv small/"${date} ${name}.jpg.tmp" small/"${date} ${name}.jpg"
        echo "$date ${name}.jpg"
    else
        echo "$date ${name}.jpg...failed, $ret"
        rm small/"${date} ${name}.jpg.tmp"
    fi
} &
let "index+=1" # 注意这句话的位置
done
wait
```

这样 12 张图片一起在后台下载（但是感觉会有网速的分摊，速度并不会提升多大），wait（等待后台进程运行结束）之后进入下一个页面，**其中要注意的就是&之后的 let “index+=1”条件**
其实如果对于命令参数不交叉的可以不进行并发处理，直接在 shell 中使用**./bing.sh 1 63 &**就可以实现后台并发，并且上面的那个并发由于有网速的分摊下载速度并不会提升很大
源代码见附录

3.6 随机下载图片（选做部分）

3.6.1 随机下载次数

首先这个处理一下命令行参数得到随机下载次数$2，然后在 for 循环的控制下进入下面的操作

```
for i in `seq 1 $2`
do ...
done
    rm jpg.md5 tmp.txt
    exit 0
```

3.6.2 获取网页信息

这个即是获取给定网址 https://bing.ioliu.cn/v1/rand?type=json 中的相关信息（每次都是获取都是随机的），提取 date，URL 以及 name，重定向到一个临时文件 tmp.txt
简单的命令如下：
cat html/rand_$i.html | sed -e 's/"url":/\n\0 /' -e 's/"copyright":/\n\0 /' -e 's/(.*)/\n\0/g' -e 's/（.*）/\n\0/' -e 's/"/ /g' |awk -f rand.awk |sed -e 's/^ *//g' > tmp.txt

需要的 awk 文件：

```
u104@LiSongYang:~/down$ vi rand.awk
 1 /date/ { date=$6; }
 2 /url/ { url=$3; }
 3 /copyright/ {
 4     $1="";
 5     $2="";
 6     print date;
 7     print url;
 8     print $0;
 9 }
```

最终的文件信息形式如下：

```
20181116
http://h1.ioliu.cn/bing/MandarinDucksUK_EN-GB10090169541_1920x1080.jpg?imageslim
Mandarin ducks perched on a branch
```

3.6.3 下载图片

然后使用类似于上面 while read line 的操作读取文件，获取图片，同样要注意那个坑

```
while read date
do
    read url
    read name
    name1="$name"
    date1="$date"
    wget -O small/"$date $name.jpg.tmp" "$url" -nc -q
done < tmp.txt
```

3.6.4 判重—md5sum

对于判重需要计算 md5sum，在整个循环之前获取所有.jpg 文件的 md5sum 信息存在 jpg.md5 临时文件中，然后对于每个新下载的图片（.tmp），计算此图片的 md5sum 值，与 jpg.md5 文件中 md5sum 信息比较，如果没有相同的就将这个 md5.sum 添加在 jpg.md5 文件末尾，并且改变文件名（.jpg），输出提示信息；否则就删除.tmp 文件，提示文件已经存在的信息

```
str=`md5sum small/"$date1 $name1.jpg.tmp"`
md5_new=(${str:0:32})    # 获取临时文件的md5sum

while read line
do
    md5_old=(${line:0:32})  # 比较每个文件的md5sum
    if [ $md5_old == $md5_new ];then
        flag=1
        break
    fi
done < jpg.md5

if [ $flag = 1 ];then    # 图片已经存在
    echo "relative photo already exists."
    rm small/"$date1 $name1.jpg.tmp"
else
    echo "$date1 $name1.jpg"
    mv small/"$date1 $name1.jpg.tmp" small/"$date1 $name1.jpg"
    md5sum small/"$date1 $name1.jpg" >> jpg.md5
fi
```

源代码见附录

4 运行效果

4.1 给定范围下载

```
2020-03-20 一只苍鹭栖息在加利福尼亚索尔顿海的木桩上.jpg
2020-03-19 柬埔寨吴哥窟的日出.jpg
2020-03-18 藏红花.jpg
2020-03-17 纽格兰奇墓，爱尔兰博因河谷.jpg
2020-03-16 克卢恩国家公园里的凯瑟琳湖和沃辛顿山，育空.jpg
2020-03-15 缅因州达马里斯科塔地区的佩马基德灯塔.jpg
2020-03-14 哥本哈根的环形桥，丹麦.jpg
2020-03-13 泰河畔伊尔的Les Orgues，法国北加泰罗尼亚.jpg
2020-03-12 西耶斯塔海滩鸟瞰图，佛罗里达西耶斯塔岛.jpg
2020-03-11 瓦普斯克国家公园内向洞穴外张望的北极熊幼崽，加拿大马尼托巴.jpg
2020-03-10 加拿大恶地里石窟上方的银河，加拿大亚伯达省德拉姆黑勒.jpg
2020-03-09 克拉克湖国家公园中一只休憩的灰熊幼崽，阿拉斯加库克湾.jpg
2020-03-08 曼哈顿河滨公园中的圣女贞德纪念碑.jpg
2020-03-07 以东京塔为背景皇宫附近盛开的樱花，日本东京.jpg
2020-03-06 黄石国家公园里的一只雄性山地蓝知更鸟，怀俄明州.jpg
2020-03-05 科罗纳多国家森林里的巨柱仙人掌花，亚利桑那.jpg
2020-03-04 Dos Ojos自然公园里El Pit Cenote的潜水员，墨西哥金塔纳罗奥.jpg
2020-03-03 榕树上的幽灵眼镜猴，印度尼西亚Tangkoko Batuangus自然保护区.jpg
2020-03-02 “精灵烟囱”和窑洞，土耳其卡帕多西亚.jpg
2020-03-01 大西洋和特内里费山脉上空的流云，西班牙加那利群岛.jpg
2020-02-29 一只华莱士飞蛙掠过森林地面.jpg
2020-02-28 冬天的水獭溪，佛蒙特州布兰登.jpg
2020-02-27 北极国家野生动物保护区里的一只雌性北极熊和她的幼崽，阿拉斯加.jpg
2020-02-26 阿卡迪亚国家公园，缅因州.jpg
```

4.2 随机下载

```
20180210 「小樽雪あかりの路」北海道，小樽.jpg
20171109 Point Arena Light in California, USA.jpg
20180527 金门大桥，美国旧金山.jpg
relative photo already exists.
20190530 2018年曼哈顿悬日期间的克莱斯勒大厦与42号街，纽约市.jpg
20180809 瓦伊纳皮克丘山周围的云雾，秘鲁.jpg
20180815 在设得兰群岛休憩的欧亚水獭，苏格兰.jpg
20161220 一株被灯带装饰了的树形仙人掌，亚利桑那州.jpg
20180329 Blutbürzelarassari im Naturschutzgebiet Refugio Paz de Las Aves, Ecuador.jpg
20200331 连接巴林托伊附近两处悬崖的Carrick-a-Rede索桥，北爱尔兰安特里姆.jpg
20180602 「ソンム湾」フランス.jpg
20200208 【今日元宵节】祝大家平安健康，团团圆圆！.jpg
20181128 An Arctic fox in Dovrefjell, Norway.jpg
20170923 Nesting rabbit.jpg
relative photo already exists.
20180705 Fourth of July fireworks in Morton, Minnesota.jpg
20190116 The distinctive roof of the British Museum and the surrounding area.jpg
20190306 位于地中海的切法卢，意大利西西里岛.jpg
20171222 A grey squirrel peeking out of its den.jpg
20171219 Star trails above the Christmas tree at Wakehurst, West Sussex.jpg
relative photo already exists.
relative photo already exists.
20160920 拉达克，印度.jpg
20180122 「ヒガシアメリカオオコノハズク」アメリカ，ジョージア州.jpg
20180215 Ponte Te Rewa Rewa perto de New Plymouth, Nova Zelândia.jpg
20180426 瓜达洛普山国家公园的红葡萄酒杯仙人掌，美国德克萨斯州.jpg
20171109 「エンチャントメンツの紅葉」米国ワシントン州.jpg
20181101 「アカギツネの子ども」ドイツ，ラウジッツ.jpg
20190213 彩色灯光下的冰柱，日本秩父市.jpg
```

总结

1. 两个题目都是熟悉 linux shell 脚本的使用，以及对 linux 常用命令如：筛选工具 awk、流编辑 sed、grep 等等
2. 第一个题目主要就是练练手，其中有点简单的条件分支以及 test/[]命令的使用
3. 第二个题目觉得有点意义，做一个下载图片的工具。对一个网页中的所有照片信息提取出来之后，刚开始由于想着使用变量来存储这些信息，所以有点麻烦，但是学习了 linux 的字符串的常用操作以及各种坑，知识点颇丰；之后发现直接使用 while read line 来读取文件就不需要先将信息用变量存下来然后再一个一个使用，但是这里有一个大坑，就是刚开始我是使用 cat file | while read line ; do echo···;done 这样的形式，这样的管道是在一个子 shell 中执行（因为管道会开启子 **shell**，使得 **while** 中的命令都在子 **shell** 中执行，而且，**cat test1** 会一次性将 **test1** 文件所有数据装入内存，如果 **test1** 文件足够大，会直接占用巨量内存，并且其中的变量在退出 **while** 之后不可以使用），所以其中的变量在 while 循环结束之后无法使用，为此自己专门一遍博客总结 while read line 方法 https://blog.csdn.net/Miracle_ps/article/details/105996799

附录

./tcp.sh

```
date=`date +%Y-%m-%d`
hour=0
minute=0
rec_send_last=0
label=" "
flag=0
while true
do
    label=" "
    minute=`expr $minute + 1`
    if [ $minute -eq 60 ]
    then
        hour=`expr $hour + 1`
        minute=0
    fi
    str1=`netstat -s|awk "NR==35"`
```

```
    str2=`netstat -s|awk "NR==36"`
    rec_before=`echo $str1|grep -o '[0-9]*'`
    send_before=`echo $str2|grep -o '[0-9]*'`
    # rec_before=`netstat -s|awk "NR==35"|grep -o '[0-9]*'`
    sleep 60
    str1=`netstat -s|awk "NR==35"`
    str2=`netstat -s|awk "NR==36"`
    rec_after=`echo $str1|grep -o '[0-9]*'`
    send_after=`echo $str2|grep -o '[0-9]*'`
    rec=`expr $rec_after - $rec_before`
    send=`expr $send_after - $send_before`
    rec_send=`expr $rec + $send`
    if [ $flag -ne 0 ];then
        if [ $rec_send -ge $rec_send_last ];then
            d=`expr $rec_send - $rec_send_last`
            if [ $d -gt 10 ];then
                label="+"
            fi
        elif [ $rec_send -le $rec_send_last ];then
            d=`expr $rec_send_last - $rec_send`
            if [ $d -gt 10 ];then
                label="-"
            fi
        fi
    fi
    flag=1
    rec_send_last=`expr $rec_send`# 记下上一分钟的收发总数

    printf "%s %02d:%02d %8d %8d %8d %s\n" $date $hour $minute $rec $send
$rec_send $label
done
```

./bing.sh**（分别获取图片信息）**

./bing.sh（含选做部分）

```
page_min=1
page_max=126
if [ $# = 0 ];then
    echo "default args[1 126]..."
elif [ $# -eq 1 -o $# -gt 2    ];then
    echo "Usage: $0 : [num1 num2]/[rand num1]"
```

```
    exit 1
elif [ $# = 2 -a $1 = "rand" ];then
if [ ! -e small/*.jpg ];then       #  如果没有图片就自己添加文件内容
    echo "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx" > jpg.md5
else
    md5sum small/*.jpg > jpg.md5 #  首先计算已有文件的 md5
fi
for i in `seq 1 $2`
do
    flag=0
    date1="123"
    name1="ps"
    wget -O html/rand_$i.html "https://bing.ioliu.cn/v1/rand?type=json" -q
    cat html/rand_$i.html | sed -e 's/"url":/\n\0 /' -e 's/"copyright":/\n\0 /' -e
's/(.*)/\n\0/g' -e 's/"/ /g' |awk -f rand.awk |sed -e 's/^ *//g' > tmp.txt
    while read date
    do
        read url
        read name
        name1="$name"
        date1="$date"
        wget -O small/"$date $name.jpg.tmp" "$url" -nc -q
    done < tmp.txt

    str=`md5sum small/"$date1 $name1.jpg.tmp"`
    md5_new=(${str:0:32})      #  获取临时文件的 md5sum

    while read line
    do
        md5_old=(${line:0:32})      #  比较每个文件的 md5sum
        if [ $md5_old == $md5_new ];then
            flag=1
            break
        fi
    done < jpg.md5

    if [ $flag = 1 ];then
        echo "relative photo already exists."
        rm small/"$date1 $name1.jpg.tmp"
    else
        echo "$date1 $name1.jpg"
        mv small/"$date1 $name1.jpg.tmp" small/"$date1 $name1.jpg"
        md5sum small/"$date1 $name1.jpg" >> jpg.md5
    fi
```

```
done
    rm jpg.md5 tmp.txt
    exit 0
else
    expr $1 + 6 &> /dev/null  #  判断$1 是否为整数
    if [ $? -ne 0 ];then
        echo "Usage: $0 : [num1 num2]"
        exit 1
    fi
    page_min=$1
    page_max=$2
fi

for i in `seq $page_min $page_max`
do
    wget -O html/${i}.html https://bing.ioliu.cn/?p=$i -q

    str=`cat html/$i.html -n| sed -e 's/<[^<>]*>/\n\0/g' -e 's/>/ /g' -e 's/"/
/g'| grep -E 'download    href|[^0][0-9]{4}-[0-9]{2}-[0-9]{2}'|awk -f file.awk `

    declare -a arr #  声明数组存放 name
    index=0
    for row in `seq 1 12`
    do
        arr[$index]=`cat html/$i.html| sed -e 's/<[^<>]*>/\n\0/g' -e
's/<h3>//g'| grep ©|sed -e 's/ (.*)//g'|awk "NR==$row"`
        let "index+=1"
    done
    index=0

    for var1 in ${str[@]}
    do
    {
        array1=(${var1//|/ })
        date=""
        name=""
        url=""
        for var2 in ${array1[@]}
        do
            if [ "$date" = "" ];then
                date=$var2
            elif [ "$url" = "" ];then
                url=$var2
            fi
```

```
            done
            name=${arr[$index]}
            let "index+=1"
            wget -O small/"${date} ${name}.jpg.tmp" $url -q -nc
            ret=`echo $?`
            if [ $ret -eq 0 ];then
                mv small/"${date} ${name}.jpg.tmp" small/"${date} ${name}.jpg"

                echo "$date ${name}.jpg"
            else
                echo "$date ${name}.jpg...failed, $ret"
                rm small/"${date} ${name}.jpg.tmp"
            fi
        }
        done
done
```

./bing.sh（含选做和并发处理）

```
page_min=1
page_max=126
if [ $# = 0 ];then
        echo "default args[1 126]..."
elif [ $# -eq 1 -o $# -gt 2    ];then
        echo "Usage: $0 : [num1 num2]/[rand num1]"
        exit 1
elif [ $# = 2 -a $1 = "rand" ];then
if [ ! -e small/*.jpg ];then        #  如果没有图片就自己添加文件内容
        echo "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx" > jpg.md5
else
        md5sum small/*.jpg > jpg.md5 #  首先计算已有文件的 md5
fi
for i in `seq 1 $2`
do
        flag=0
        date1="123"
        name1="ps"
        wget -O html/rand_$i.html "https://bing.ioliu.cn/v1/rand?type=json" -q
        cat html/rand_$i.html | sed -e 's/"url":/\n\0 /' -e 's/"copyright":/\n\0 /' -e
's/(.*)/\n\0/g' -e 's/"/ /g' |awk -f rand.awk |sed -e 's/^ *//g' > tmp.txt
        while read date
        do
                read url
                read name
```

```
        name1="$name"
        date1="$date"
        wget -O small/"$date $name.jpg.tmp" "$url" -nc -q
    done < tmp.txt

    str=`md5sum small/"$date1 $name1.jpg.tmp"`
    md5_new=(${str:0:32})      # 获取临时文件的 md5sum

    while read line
    do
        md5_old=(${line:0:32})      # 比较每个文件的 md5sum
        if [ $md5_old == $md5_new ];then
            flag=1
            break
        fi
    done < jpg.md5

    if [ $flag = 1 ];then
        echo "relative photo already exists."
        rm small/"$date1 $name1.jpg.tmp"
    else
        echo "$date1 $name1.jpg"
        mv small/"$date1 $name1.jpg.tmp" small/"$date1 $name1.jpg"
        md5sum small/"$date1 $name1.jpg" >> jpg.md5
    fi
done
    rm jpg.md5 tmp.txt
    exit 0
else
    expr $1 + 6 &> /dev/null  # 判断$1 是否为整数
    if [ $? -ne 0 ];then
        echo "Usage: $0 : [num1 num2]"
        exit 1
    fi
    page_min=$1
    page_max=$2
fi

for i in `seq $page_min $page_max`
do
    wget -O html/${i}.html https://bing.ioliu.cn/?p=$i -q

    str=`cat html/$i.html -n| sed -e 's/<[^<>]*>/\n\0/g' -e 's/>/ /g' -e 's/"/
/g'| grep -E 'download    href|[^0][0-9]{4}-[0-9]{2}-[0-9]{2}'|awk -f file.awk `
```

```
    declare -a arr #  声明数组存放 name
    index=0
    for row in `seq 1 12`
    do
        arr[$index]=`cat html/$i.html| sed -e 's/<[^<>]*>/\n\0/g' -e
's/<h3>//g'| grep ©|sed -e 's/ (.*)//g'|awk "NR==$row"`
        let "index+=1"
    done
    index=0

    for var1 in ${str[@]}
    do
    {
        array1=(${var1//|/ })
        date=""
        name=""
        url=""
        for var2 in ${array1[@]}
        do
            if [ "$date" = "" ];then
                date=$var2
            elif [ "$url" = "" ];then
                url=$var2
            fi
        done
        name=${arr[$index]}
        wget -O small/"${date} ${name}.jpg.tmp" $url -q -nc
        ret=`echo $?`
        if [ $ret -eq 0 ];then
            mv small/"${date} ${name}.jpg.tmp" small/"${date} ${name}.jpg"

            echo "$date ${name}.jpg"
        else
            echo "$date ${name}.jpg...failed, $ret"
            rm small/"${date} ${name}.jpg.tmp"
        fi
    } &
    let "index+=1" #  注意这句话的位置
    done
    wait
done
```

./bing.sh（简洁，整体获取图片信息）

```
page_min=1
page_max=126
if [ $# = 0 ];then
	echo "default args[1 126]..."
elif [ $# -eq 1 -o $# -gt 2    ];then
	echo "Usage: $0 : [num1 num2]/[rand1 num1]"
	exit 1
elif [ $# = 2 -a $1 = "rand" ];then
if [ ! -e small/*.jpg ];then    #  如果没有图片就自己添加文件内容
	echo "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx" > jpg.md5
else
	md5sum small/*.jpg > jpg.md5 #  首先计算已有文件的 md5
fi
for i in `seq 1 $2`
do
	flag=0
	date1="123"
	name1="ps"
	wget -O html/rand_$i.html "https://bing.ioliu.cn/v1/rand?type=json" -q
	if test $? != 0;then
		echo "wget html failed."
		exit 1
	fi
	cat html/rand_$i.html | sed -e 's/"url":/\n\0 /' -e 's/"copyright":/\n\0 /' -e
's/(.*)/\n\0/g' -e 's/（.*）/\n\0/' -e 's/"/ /g' |awk -f rand.awk |sed -e 's/^
*//g' > tmp.txt
	while read date
	do
		read url
		read name
		name1="$name"
		date1="$date"
		wget -O small/"$date $name.jpg.tmp" "$url" -nc -q
	done < tmp.txt

	str=`md5sum small/"$date1 $name1.jpg.tmp"`
	md5_new=(${str:0:32})       #  获取临时文件的 md5sum

	while read line
	do
		md5_old=(${line:0:32})       #  比较每个文件的 md5sum
```

```
            if [ $md5_old == $md5_new ];then
                flag=1
                break
            fi
        done < jpg.md5

        if [ $flag = 1 ];then      #  图片已经存在
            echo "relative photo already exists."
            rm small/"$date1 $name1.jpg.tmp"
        else #  图片不存在
            echo "$date1 $name1.jpg"
            mv small/"$date1 $name1.jpg.tmp" small/"$date1 $name1.jpg"
            md5sum small/"$date1 $name1.jpg" >> jpg.md5       # append 新图
片的 md5sum
        fi
done
        rm jpg.md5 tmp.txt
        exit 0
else
        expr $1 + 6 &> /dev/null              #  判断$1 是否为整数
        if [ $? -ne 0 ];then
            echo "Usage: $0 : [num1 num2]"
            exit 1
        fi
        page_min=$1
        page_max=$2
fi

for i in `seq $page_min $page_max`
do
        wget -O html/${i}.html https://bing.ioliu.cn/?p=$i -q

        cat html/$i.html -n| sed -e 's/<[^<>]*>/\n\0/g' -e 's/>/ /g' -e 's/"/ /g' -e
's/([^()]*)//g'| grep -E 'download    href|<h3|[^0][0-9]{4}-[0-9]{2}-[0-9]{2}'|awk
-f up.awk |sed -e 's/^ *//g' > up.txt

        while true
        do
        {
            read date
            if [ $? != 0 ];then
                break
            fi
            read name
```

```
            read url
            wget -O small/"${date} ${name}.jpg.tmp" $url -q -nc
            ret=`echo $?`
            if [ $ret -eq 0 ];then
                mv small/"${date} ${name}.jpg.tmp" small/"${date} ${name}.jpg"

                echo "$date ${name}.jpg"
            else
                echo "$date ${name}.jpg...failed, $ret"
                rm small/"${date} ${name}.jpg.tmp"
            fi
        }
        done < up.txt
done
rm up.txt
```