浙江大学实验报告

课程名称：Linux 应用技术基础 实验类型：综合型

实验项目名称：实验三 程序设计

学生姓名：王子腾 专业：软件工程 学号：3180102173

电子邮件地址：3180102173@zju.edu.cn

实验日期：2020 年 6 月 21 日

一、 实验环境

计算机配置：

处理器 Intel Core i7-8550 CPU @ 1.80GHz 1.99Ghz

内存 8GB DDR4 2400MHz

硬盘 128GB PCle 固态硬盘+1TB 硬盘

显卡 NVIDIA GeForce 150MX

主操作系统环境：

Windows 10 家庭中文版

Linux 版本：

Ubantu 18.04

二、 实验内容和结果及分析

1. （15 分）编写一个 shell 脚本程序，它带一个命令行参数，这个参数是一个文件名。如果这个文件是一个普通文件，则打印文件所有者的名字和最后的修改日期。如果程序带有多个参数，则输出出错信息。

```
1. FileName: 1.sh
2. #Author: Ziteng Wang
3. #Create Date: 2020-06-19
4. #Last Modify: 2020-06-20
5. #Description:
6. #  if 1 ordinary file name argument: print owner name and modify
   date
7. #  if more than 1 argument: print error info
```

```
#  if the file is not an ordinary file: print error info

#! /bin/bash
if test $# -ne 1    # Not 1 Argument
    then
        echo "Exactly 1 argument is expected"   # Error info
        exit 1      # Error
fi
if test -f "$1"     # Ordinary File
    then
        filename="$1"       # Ordinary File
        set -- $(ls -l $filename)   # List result
        user_name="$3"      # Owner
        modify_date="$6$7 $8"   # Date
        echo "Name  Date"   # Head line
        echo "$user_name    $modify_date"   # Print
        exit 0
fi
echo "$1: argument should be an ordinary file"  # Error
exit 1
```

程序首先检测参数个数，若非单参数则报错并退出，之后检测是否为普通文件，若是则获取 ls 中的指定字段信息，并列出。

```
ziteng@Ziteng-VBox:~/lab3$ ls
1.sh  2.sh  3.sh  4.sh  5.sh  dirsync.sh  test  test5
ziteng@Ziteng-VBox:~/lab3$ ./1.sh
Exactly 1 argument is expected
ziteng@Ziteng-VBox:~/lab3$ ./1.sh test5
test5: argument should be an ordinary file
ziteng@Ziteng-VBox:~/lab3$ ./1.sh 3.sh
Name    Date
ziteng  6月20 13:44
```

2. （15 分）编写 shell 程序，统计指定目录下的普通文件、子目录及可执行文件的数目，统计该目录下所有普通文件字节数总和，目录的路径名字由参数传入。

```
#FileName: 2.sh
#Author: Ziteng Wang
#Create Date: 2020-06-19
#Last Modify: 2020-06-20
#Description:
#  if 1 direction file name argument:
#    calculate numbers of ordinary files/ subdirection files/ executable files
#    claculate bytes of ordinary files
#  if more than 1 argument: print error info
#  if not an direction file: print error info
```

```
11.
12. #! /bin/bash
13. if test $# -ne 1 # Not 1 Argument
14.  then
15.   echo "Exactly 1 argument is expected" # Error info
16.   exit 1  # Error
17. fi
18. f_cnt=0   # ordinary file counter
19. d_cnt=0   # dir file counter
20. x_cnt=0   # excutable file counter
21. b_cnt=0   # bytes counter
22. array=$(find $1 -maxdepth 1) # find all files within dir $1
23. for i in $array  # traversal $array
24. do
25.  if [ -f $i ] # ordinary file
26.   then
27.    ((f_cnt++))
28.    set `wc -c $i` # get bytes
29.    ((b_cnt+=$1))
30.  fi
31.  if [ -d $i ] # dir file
32.   then
33.    ((d_cnt++))
34.  fi
35.  if [ -x $i ] # excutable file
36.   then
37.    ((x_cnt++))
38.  fi
39. done
40. echo "Calculating Result:"  # head line
41. echo "-f -d -x bytes"
42. echo "$f_cnt $d_cnt $x_cnt $b_cnt" # result
43. exit 0
```

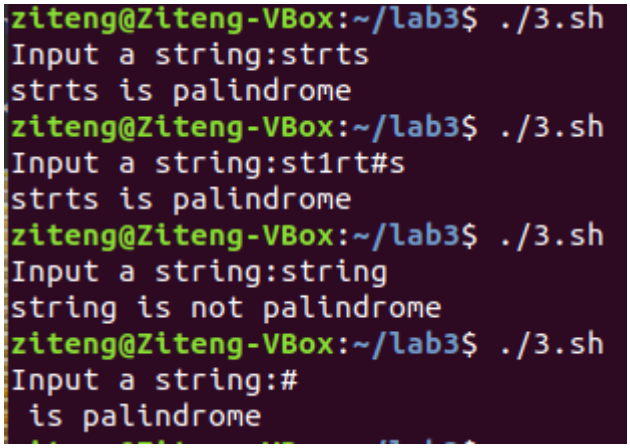
程序首先检测参数个数，若非单参数则报错并退出，之后设置各类型文件计数器，并开始循环遍历路径下所有文件，根据文件类型进行分别计数，并遍历结束后输出结果。

```
ziteng@Ziteng-VBox:~/lab3$ ./2.sh
Exactly 1 argument is expected
ziteng@Ziteng-VBox:~/lab3$ ./2.sh .
Calculating Result:
-f      -d      -x      bytes
6       3       9       5112
ziteng@Ziteng-VBox:~/lab3$ ./2.sh ..
Calculating Result:
-f      -d      -x      bytes
27      22      24      430187
ziteng@Ziteng-VBox:~/lab3$ ls
1.sh  2.sh  3.sh  4.sh  5.sh  dirsync.sh  test  test5
```

3. （15 分）编写一个 shell 脚本，输入一个字符串，忽略（删除）非字母后，检测该字符串是否为回文(palindrome)。对于一个字符串，如果从前向后读和从后向前读都是同一个字符串，则称之为回文串。例如，单词“mom”，“dad”和“noon”都是回文串。

```
1.  #FileName: 3.sh
2.  #Author: Ziteng Wang
3.  #Create Date: 2020-06-19
4.  #Last Modify: 2020-06-20
5.  #Description:
6.  #  input 1 string argument:
7.  #    ignore non-alphabetic character, check if palindrome
8.
9.  #! /bin/bash
10. echo -n "Input a string:"
11. read line   # read in a var in $line
12. line=`echo $line | tr -cd [:alpha:]` # delete non-alpha character
    s
13. rline=`echo $line | rev`      # reverse the string
14. if [[ $line == $rline ]]      # if $line equals to reverse $line
15.  then
16.   echo "$line is palindrome"  # palindrome
17.  else
18.   echo "$line is not palindrome" # not palindrome
19. fi
20. exit 0
```

首先读入一个字符串并存储至$line 变量，之后对字符串去除非字母字符，并将其反转的结果存在$rline 中，将两者比较，若相等则为回文，否则非回文。



4. （15 分）编写一个 shell 脚本，把当前目录下文件大小大于 100K 的文件全部移动到~/tmp/ 目录下。

```
#FileName: 4.sh
#Author: Ziteng Wang
#Create Date: 2020-06-19
#Last Modify: 2020-06-20
#Description:
#  move files larger than 100k to ~/tmp/

#! /bin/bash
array=$(find -size +100k) # size > 100k
for i in $array    # traverse
do
 mv $i ~/tmp/      # transport
done
echo "success"     # echo info
exit 0
```

首先使用 find 指令搜索大于 100k 的文件，并存储在$array 集合内，之后遍历集合内各文件并移入~/tmp/目录下，之后返回 success 回显并结束程序

```
ziteng@Ziteng-VBox:~/lab3$ ls -l
总用量 384
-rwxr-xr-x 1 ziteng ziteng    719 6月  21 18:52 1.sh
-rwxr-xr-x 1 ziteng ziteng   1029 6月  21 20:12 2.sh
-rwxr-xr-x 1 ziteng ziteng    410 6月  20 13:44 3.sh
-rwxr-xr-x 1 ziteng ziteng    243 6月  20 14:23 4.sh
-rwxr-xr-x 1 ziteng ziteng   2444 6月  21 11:53 5.sh
-rwxr-xr-x 1 ziteng ziteng    267 6月  21 12:02 dirsync.sh
-rw-r--r-- 1 ziteng ziteng 356570 6月  21 21:15 largeFile
drwxr-xr-x 2 ziteng ziteng   4096 6月  21 18:52 test
drwxr-xr-x 4 ziteng ziteng   4096 6月  21 21:14 test5
ziteng@Ziteng-VBox:~/lab3$ ./4.sh
success
ziteng@Ziteng-VBox:~/lab3$ ls -l ~/tmp/
总用量 352
-rw-r--r-- 1 ziteng ziteng 356570 6月  21 18:52 largeFile
ziteng@Ziteng-VBox:~/lab3$ ls -l .
总用量 32
-rwxr-xr-x 1 ziteng ziteng  719 6月  21 18:52 1.sh
-rwxr-xr-x 1 ziteng ziteng 1029 6月  21 20:12 2.sh
-rwxr-xr-x 1 ziteng ziteng  410 6月  20 13:44 3.sh
-rwxr-xr-x 1 ziteng ziteng  243 6月  20 14:23 4.sh
-rwxr-xr-x 1 ziteng ziteng 2444 6月  21 11:53 5.sh
-rwxr-xr-x 1 ziteng ziteng  267 6月  21 12:02 dirsync.sh
drwxr-xr-x 2 ziteng ziteng 4096 6月  21 21:16 test
drwxr-xr-x 4 ziteng ziteng 4096 6月  21 21:16 test5
```

5. （30 分）编写一个实现文件备份和同步的 shell 脚本程序 dirsync。程序的参数是两个需要备份同步的目录，如:

dirsync ~\dir1 ~\dir2　# ~\dir1 为源目录，~\dir2 为目标目录

dirsync 程序实现两个目录内的所有文件和子目录（递归所有的子目录）内容保持一致。程序基本功能如下。

1) 备份功能：目标目录将使用来自源目录的最新文件，新文件和新子目录进行升级，源目录将保持不变。dirsync 程序能够实现增量备份。
2) 同步功能：两个方向上的旧文件都将被最新文件替换，新文件都将被双向复制。源目录被删除的文件和子目录，目标目录也要对应删除。
3) 其它功能自行添加设计

```
1. #FileName: dirsync
2. #Author: Ziteng Wang
3. #Create Date: 2020-06-19
4. #Last Modify: 2020-06-21
5. #Description:
6. #   1: backup -- update new file and keep increment copy
7. #   2: synchronize -- update two dirs in bothway, keep both dir m
   ost updated
8.
```

```
#! /bin/bash
# Function Sync()
# synchronize
Sync(){
local cur1 cur2 status1 list1 list2 i j  # declare all local vars
cur1=$1  # src
cur2=$2  # dest
status1=0 # flag: find or not (0: not find)
if [ ! -d $cur2 ] # if $2 is not direction
then
 mkdir $cur2  # mkdir
fi
list1=$(ls $cur1) # content of 1
list2=$(ls $cur2) # content of 2
for i in $list1    # Outer loop: files in src
do
 for j in $list2   # Inner loop: files in dest
 do
  if [[ $i != $j ]] # Not same name
  then
   continue;
  elif [ -d ${cur1}/${i} ] && [ -d ${cur2}/${j} ] # same name && same -d
  then
   status1=1  # find
   # echo -n "digui1:"
   # echo "${cur1}/${i} ${cur2}/${j}"
   Sync ${cur1}/${i} ${cur2}/${j}
  elif [ -f ${cur1}/${i} ] && [ -f ${cur2}/${j} ] # same name && same -f
  then
   status1=1  # find
   # echo "here1"
   if [[ `find ${cur1}/${i} -newer ${cur2}/${j}` == ${cur1}/${i} ]] # $i is newer
   then
    \cp -rf ${cur1}/${i} ${cur2}/${j} # $i -> $j
   fi
  fi
 done

 if [[ $status1 == 0 ]] # not find $i in list2
 then
```

```
  if [ ${cur1}/${i} == ${cur2} ]  # check if (dest) is in (src) --> avoid recursive copy
  then
   echo "file exist" 1>/dev/null
  elif [ -d ${cur1}/${i} ] # dir file
  then
   # echo "copy: ${cur1}/${i} -> $cur2"
   cp -a ${cur1}/${i} $cur2 # copy the folder to (dest) dir
  else
   cp -a ${cur1}/${i} $cur2 # copy the file to (dest) dir
  fi
 fi

 status1=0  # re-initialise
done

status1=0   # initialise for next loop

for i in $list2  # Outer loop: files in dest
do
 for j in $list1 # inner loop: files in src
 do
  if [[ ${i} != ${j} ]] # Not same name
  then
   continue;
  elif [ -f ${cur2}/${i} ] && [ -f ${cur1}/${j} ] # same name && same -f
  then
   status1=1  # find
   # echo "here2"
   if [[ `find ${cur2}/${i} -newer ${cur1}/${j}` == ${cur2}/${i} ]] # $i is newer
   then
    \cp -rf ${cur2}/${i} ${cur1}/${j} # $i -> $j
   fi
  fi
 done

 if [[ $status1 == 0 ]] # not find $i in list1
 then
  if [ ${cur1} == ${cur2}/${i} ] # check if (dest) is in (src) --> avoid recursive copy
  then
   echo "file exist" 1>/dev/null
```

```
  elif [ -d ${cur2}/${i} ] # dir file
  then
   rm -r ${cur2}/${i} # remove dir that is not in list1
  else
   cp ${cur2}/${i} ${cur1}  # reverse update
  fi
 fi
 status1=0;     # re-init
done
# echo success
}

# Entrance for the shell
src=$1       # source dir
dest=$2       # destination dir
echo "Choose a mode:"   # tips
echo " 1:Backup"
echo " 2:synchronize"
echo -n "Your choice: "
read line      # read in "option"
case $line in
 1) rsync -ru ${src}/ ${dest} # function call
 ;;
 2) Sync $src $dest   # Sync function call
 ;;
esac
echo finish      # echo back
exit 0

```

程序首先输出提示，并读取用户输入，通过 case 语句进行功能匹配，进而调用 rsync 指令或 Sync 函数进行备份或同步。

在备份功能中，通过更新和增量模式将 src 地址中的文件以递归的方式进行更新和覆盖。

在同步功能中，Sync 函数首先判断当前目标地址是否存在，若不存在则新建地址；之后通过嵌套循环，分别将 src 目录中的文件与 dest 目录的文件依次匹配：若未匹配上，则直接 continue；若匹配上且为子目录，则递归进入该子目录并记录找到；若匹配上且为普通文件，则记录找到并判断 src 中文件修改日期是否更近，日期更近则将该文件在 dest 中更新，若没找到，则直接将文件复制到 dest

中；之后反向更新，在 dest 中以同样的方式反向将修改日期更近的文件反向更新，最终完成任务并结束函数。

备份功能测试：

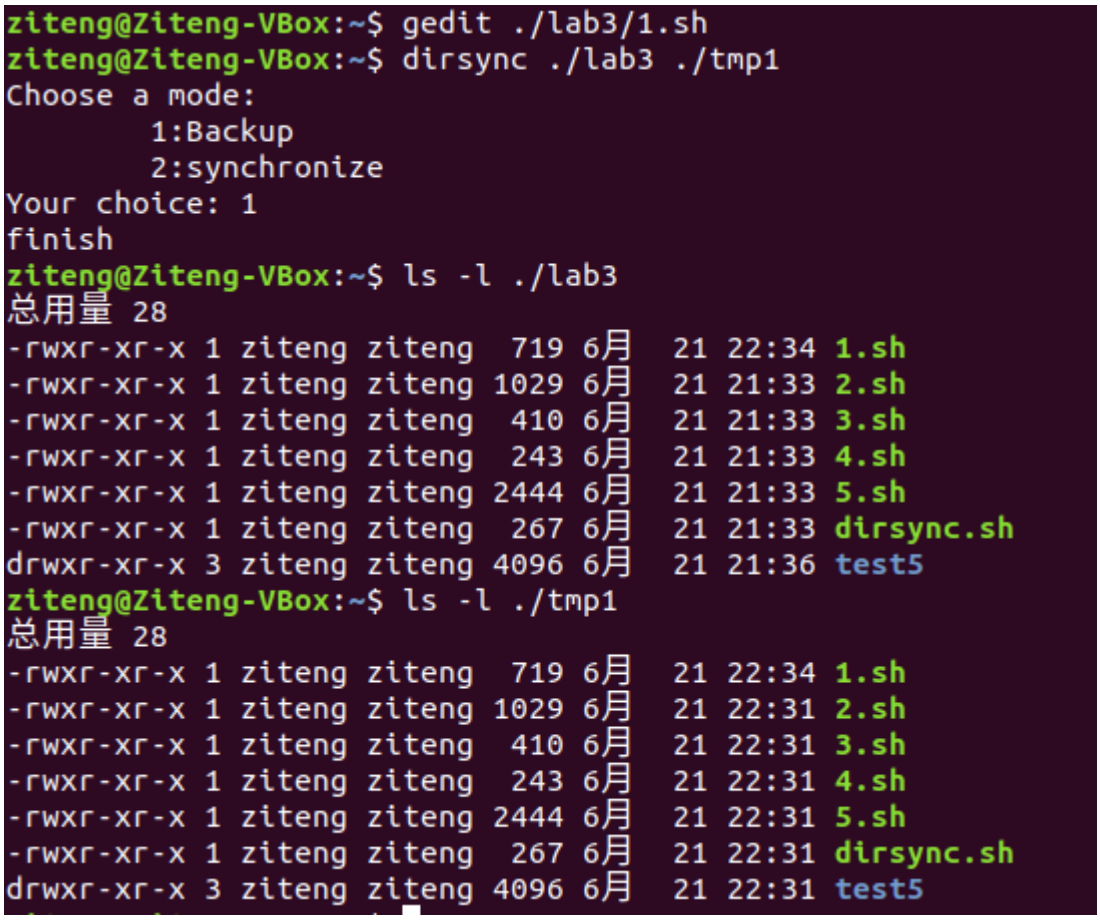
①创建~/tmp1 目录，并使其为空，之后通过备份功能将~/lab3 中的内容备份至~/tmp1 中；

测试结果：运行正确

```
ziteng@Ziteng-VBox:~$ mkdir tmp1
ziteng@Ziteng-VBox:~$ dirsync ./lab3 ./tmp1
Choose a mode:
        1:Backup
        2:synchronize
Your choice: 1
finish
ziteng@Ziteng-VBox:~$ ls -l ./lab3
总用量 28
-rwxr-xr-x 1 ziteng ziteng  719 6月  21 21:33 1.sh
-rwxr-xr-x 1 ziteng ziteng 1029 6月  21 21:33 2.sh
-rwxr-xr-x 1 ziteng ziteng  410 6月  21 21:33 3.sh
-rwxr-xr-x 1 ziteng ziteng  243 6月  21 21:33 4.sh
-rwxr-xr-x 1 ziteng ziteng 2444 6月  21 21:33 5.sh
-rwxr-xr-x 1 ziteng ziteng  267 6月  21 21:33 dirsync.sh
drwxr-xr-x 3 ziteng ziteng 4096 6月  21 21:36 test5
ziteng@Ziteng-VBox:~$ ls -l ./tmp1
总用量 28
-rwxr-xr-x 1 ziteng ziteng  719 6月  21 22:31 1.sh
-rwxr-xr-x 1 ziteng ziteng 1029 6月  21 22:31 2.sh
-rwxr-xr-x 1 ziteng ziteng  410 6月  21 22:31 3.sh
-rwxr-xr-x 1 ziteng ziteng  243 6月  21 22:31 4.sh
-rwxr-xr-x 1 ziteng ziteng 2444 6月  21 22:31 5.sh
-rwxr-xr-x 1 ziteng ziteng  267 6月  21 22:31 dirsync.sh
drwxr-xr-x 3 ziteng ziteng 4096 6月  21 22:31 test5
```

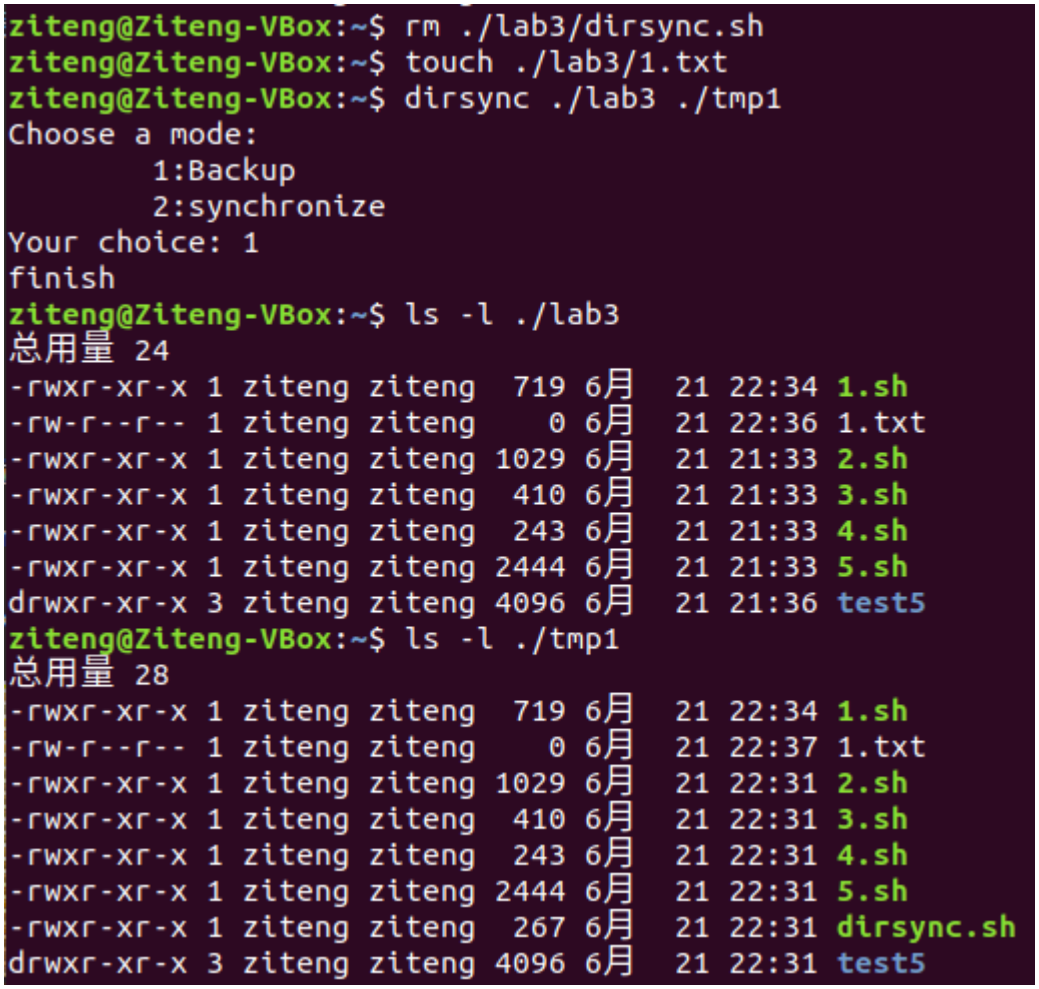
②修改~/lab3 中某一文件 1.sh，并再次备份至~/tmp1，检测更新旧文件；

测试结果：运行正确



③删除~/lab3 中某一文件 dirsync.sh，增加 1.txt，并再次备份至~/tmp1，检测增量备份；

测试结果：成功



同步功能测试：

①首先在文件管理中删除原有主目录下的 tmp 目录，并调用 dirsync 命令同步功能将~/lab3 与~/tmp 同步，检测程序在第二个参数目录不存在时能否如期创建新测试目录；

结果：运行正确

②根据题目描述，期望运行结果为两目录内容完全一致并且皆为最新文件；

测试结果：运行正确

```
ziteng@Ziteng-VBox:~$ dirsync ./lab3 ./tmp
Choose a mode:
        1:Backup
        2:synchronize
Your choice: 2
finish
ziteng@Ziteng-VBox:~$ ls ./lab3 -l
总用量 32
-rwxr-xr-x 1 ziteng ziteng  719 6月  21 21:33 1.sh
-rwxr-xr-x 1 ziteng ziteng 1029 6月  21 21:33 2.sh
-rwxr-xr-x 1 ziteng ziteng  410 6月  21 21:33 3.sh
-rwxr-xr-x 1 ziteng ziteng  243 6月  21 21:33 4.sh
-rwxr-xr-x 1 ziteng ziteng 2444 6月  21 21:33 5.sh
-rwxr-xr-x 1 ziteng ziteng  267 6月  21 21:33 dirsync.sh
drwxr-xr-x 2 ziteng ziteng 4096 6月  21 21:33 test
drwxr-xr-x 4 ziteng ziteng 4096 6月  21 21:33 test5
ziteng@Ziteng-VBox:~$ ls ./tmp -l
总用量 32
-rwxr-xr-x 1 ziteng ziteng  719 6月  21 21:33 1.sh
-rwxr-xr-x 1 ziteng ziteng 1029 6月  21 21:33 2.sh
-rwxr-xr-x 1 ziteng ziteng  410 6月  21 21:33 3.sh
-rwxr-xr-x 1 ziteng ziteng  243 6月  21 21:33 4.sh
-rwxr-xr-x 1 ziteng ziteng 2444 6月  21 21:33 5.sh
-rwxr-xr-x 1 ziteng ziteng  267 6月  21 21:33 dirsync.sh
drwxr-xr-x 2 ziteng ziteng 4096 6月  21 21:33 test
drwxr-xr-x 4 ziteng ziteng 4096 6月  21 21:33 test5
```

③为检验递归子目录功能，我们预先在~/lab3/test5 子目录中加入了两个普通文件和一个目录文件，经检测，子文件递归同步功能正常。

测试结果：运行正确

```
ziteng@Ziteng-VBox:~$ ls -l ./lab3/test5
总用量 12
-rw-r--r-- 1 ziteng ziteng 1954 6月  21 21:33 mediumFile
-rw-r--r-- 1 ziteng ziteng  963 6月  21 21:33 smallFile
drwxr-xr-x 2 ziteng ziteng 4096 6月  21 21:33 test1
ziteng@Ziteng-VBox:~$ ls -l ./tmp/test5
总用量 12
-rw-r--r-- 1 ziteng ziteng 1954 6月  21 21:33 mediumFile
-rw-r--r-- 1 ziteng ziteng  963 6月  21 21:33 smallFile
drwxr-xr-x 2 ziteng ziteng 4096 6月  21 21:33 test1
```

④分别修改~/tmp/1.sh 和~/lab3/2.sh 文件，检测旧文件被新文件替换以及双向更新功能；

测试结果：运行正确（程序用时跨越了 22:44 故两文件显示存在一分钟差异）

```
ziteng@Ziteng-VBox:~$ gedit ./tmp/1.sh
ziteng@Ziteng-VBox:~$ gedit ./lab3/2.sh
ziteng@Ziteng-VBox:~$ dirsync ./lab3 ./tmp
Choose a mode:
        1:Backup
        2:synchronize
Your choice: 2
finish
ziteng@Ziteng-VBox:~$ ls ./lab3 -l
总用量 24
-rwxr-xr-x 1 ziteng ziteng  719 6月  21 22:44 1.sh
-rwxr-xr-x 1 ziteng ziteng 1029 6月  21 22:44 2.sh
-rwxr-xr-x 1 ziteng ziteng  410 6月  21 21:33 3.sh
-rwxr-xr-x 1 ziteng ziteng  243 6月  21 21:33 4.sh
-rwxr-xr-x 1 ziteng ziteng 2444 6月  21 21:33 5.sh
drwxr-xr-x 3 ziteng ziteng 4096 6月  21 21:36 test5
ziteng@Ziteng-VBox:~$ ls ./tmp -l
总用量 24
-rwxr-xr-x 1 ziteng ziteng  719 6月  21 22:43 1.sh
-rwxr-xr-x 1 ziteng ziteng 1029 6月  21 22:44 2.sh
-rwxr-xr-x 1 ziteng ziteng  410 6月  21 21:33 3.sh
-rwxr-xr-x 1 ziteng ziteng  243 6月  21 21:33 4.sh
-rwxr-xr-x 1 ziteng ziteng 2444 6月  21 21:33 5.sh
drwxr-xr-x 3 ziteng ziteng 4096 6月  21 21:42 test5
```

⑤删除~/lab3 中的 test5 文件夹，测试目标文件的相应删除状况；

测试结果：运行正确

```
ziteng@Ziteng-VBox:~$ rm -r ./lab3/test5
ziteng@Ziteng-VBox:~$ dirsync ./lab3 ./tmp
Choose a mode:
        1:Backup
        2:synchronize
Your choice: 2
finish
ziteng@Ziteng-VBox:~$ ls ./lab3 -l
总用量 20
-rwxr-xr-x 1 ziteng ziteng  719 6月  21 22:46 1.sh
-rwxr-xr-x 1 ziteng ziteng 1029 6月  21 22:44 2.sh
-rwxr-xr-x 1 ziteng ziteng  410 6月  21 21:33 3.sh
-rwxr-xr-x 1 ziteng ziteng  243 6月  21 21:33 4.sh
-rwxr-xr-x 1 ziteng ziteng 2444 6月  21 21:33 5.sh
ziteng@Ziteng-VBox:~$ ls ./tmp -l
总用量 20
-rwxr-xr-x 1 ziteng ziteng  719 6月  21 22:46 1.sh
-rwxr-xr-x 1 ziteng ziteng 1029 6月  21 22:44 2.sh
-rwxr-xr-x 1 ziteng ziteng  410 6月  21 21:33 3.sh
-rwxr-xr-x 1 ziteng ziteng  243 6月  21 21:33 4.sh
-rwxr-xr-x 1 ziteng ziteng 2444 6月  21 21:33 5.sh
```

三、　讨论、心得（必填）（10 分）

本次实验让我熟悉了 shell 编程的逻辑与用法，以下为完成过程中遇到的问题的解决方案与相关心得：

①set 后的参数问题：在最开始使用 set 指令解析数据域时，我按照 ppt 里的格式写好的代码，总是会遇到系统提示 set 附近参数有误，通过上网查找相关资料才明白需要添加“--”指令来解析参数，改为 set -- $(ls -l $filename)后顺利解决；

②改进：经过后期的学习与巩固，我的作业 2.sh 中的循环体部分可以进一步精简为：

```
echo `find $1 -type f | wc -l`
echo `find $1 -type d | wc -l`
echo `find $1 -type f -executable | wc -l`
```

通过将查找结果用管道传递给 wc 指令进而直接将结果输出，就不需要再经过循环和判断语句计数了，这样可以提高代码运行效率；

③递归异常的问题：本次作业困扰我最大的点就在于作业 5 的递归部分，在最开始编写程序时，我参照了 c、c++ 等高级语言的递归思路，忽略了变量的作用域问题，使用普通变量用以保存当前目录与循环中间值，以为这些变量会作为临时变量保存在递归栈中，递归返回时自动恢复。但测试程序时发现循环异常退出，观察到留下的 echo 标记并没有被执行，程序的循环与递归部分发生了问题，几次修改不成后，通过上网搜索 shell 的变量定义属性，了解到函数内直接实用的变量会被自动定义为全局变量，并在触发递归后被传递到递归函数中，在函数内被修改并返回当前栈后，由于是全局变量，因此并不会恢复之前的值。了解问题原因后，我将函数体内的所有变量均声明为 local 作用域，递归异常的问题从而也就迎刃而解了。

④心得：在本次实验中，由于很多命令只能看懂但没有熟练运用，因此很多细节还是没有把握，通过做实验以及自己动手尝试，能够发现并理解很多的细节，比如空格的问题、[]与[[]] 的区别、变量的声明等。在这里做题的时候记录下来遇到的问题，可以方便日后进一步学习操作系统的指令，并加深对 shell 编程的熟练程度。