实验 3：命令行环境/Python 入门基础/Python 视觉应用

方欣 23020007021

https://github.com/nixgnaf/SDT_report.github.io

1 命令行环境

1.1 任务控制

1.1.1 (1) 在终端中执行 sleep 10000 这个任务。然后用 Ctrl-Z 将其切换到后台并使用 bg 来继续允许它。现在，使用 pgrep 来查找 pid 并使用 pkill 结束进程而不需要手动输入 pid。

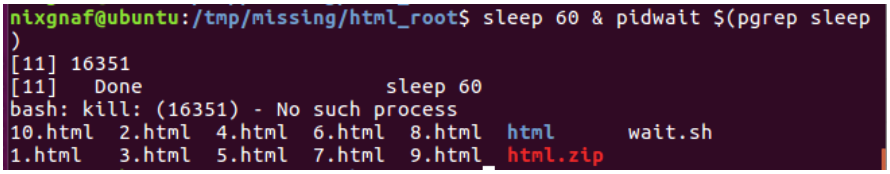
sleep 10000：将使终端挂起 10000 秒。
Ctrl-Z：发送 SIGTSTP 信号到当前前台进程，暂停它的执行，并将其放置在后台。
bg：将一个或多个暂停的进程恢复到后台继续运行。
pgrep：查找匹配特定模式的进程 ID。-f：匹配整个命令行（而不仅仅是进程名称）。
pkill：向所有命令行中包含 sleep 的进程发送信号 SIGTERM，请求终止进程。

```
nixgnaf@ubuntu:/tmp/missing/html_root$ sleep 10000
^Z
[11]+  Stopped                 sleep 10000
nixgnaf@ubuntu:/tmp/missing/html_root$ bg
[11]+ sleep 10000 &
nixgnaf@ubuntu:/tmp/missing/html_root$ pgrep sleep
15982
nixgnaf@ubuntu:/tmp/missing/html_root$ pkill -f sleep
[11]   Terminated              sleep 10000
```

1.1.2 (2) 请编写一个 bash 函数 pidwait ，它接受一个 pid 作为输入参数，然后一直等待直到该进程结束。您需要使用 sleep 来避免浪费 CPU 性能。

```
pidwait()
{
   while kill -0 $1
   do
   sleep 1
   done
   ls
}
```

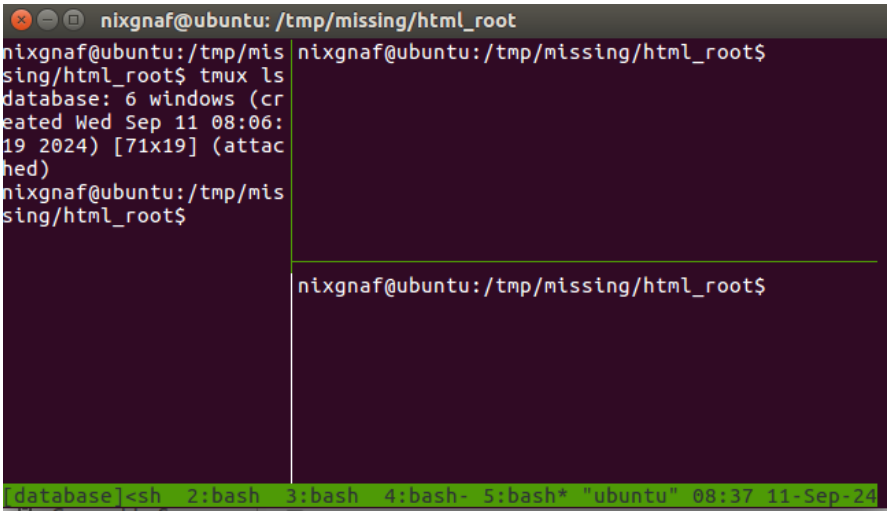
kill -0 $1 是一个用于检测进程是否存在的命令。kill -0 不会实际发送信号，但会返回进程是否存在的状态。如果进程存在，它返回 true，否则返回 false。$1 是传递给函数的第一个参数，表示待检测的进程 ID (PID)。



1.2 终端多路复用

1.2.1 (3) 请完成 tmux 教程, 参考这些步骤来学习如何自定义 tmux。

tmux new -s session_name: 创建一个带有指定名称的新会话
tmux ls: 列出所有会话
tmux attach -t session_name: 附加到现有会话
分离会话：Ctrl-b d
关闭会话：exit
创建新窗口：Ctrl-b c
切换窗口：Ctrl-b n(窗口编号)
水平分割面板：Ctrl-b ”
垂直分割面板：Ctrl-b %
切换面板：Ctrl-b <arrow key>
调整面板大小：Ctrl-b Ctrl-<arrow key>

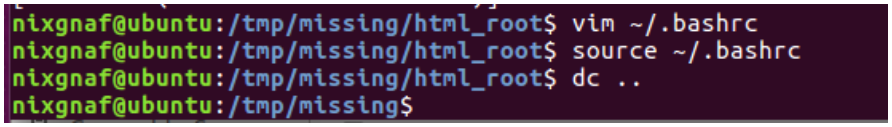


tmux set -g mouse on：开启鼠标控制
使用 Alt 键加方向键（箭头）在 tmux 面板之间快速切换的快捷键：

```
tmux bind -n M-Left select-pane -L
tmux bind -n M-Right select-pane -R
tmux bind -n M-Up select-pane -U
tmux bind -n M-Down select-pane -D
```

1.3 别名

1.3.1 (4) 创建一个 dc 别名，它的功能是当我们错误的将 cd 输入为 dc 时也能正确执行。



1.3.2 (5) 执行 history | awk '$1="";print substr($0,2)' | sort | uniq -c | sort -n | tail -n 10 来获取您最常用的十条命令，尝试为它们创建别名。

在 /.bashrc 文件中使用 alias 添加对应别名。

1.4 配置文件

1.4.1 (6) 为您的配置文件新建一个文件夹，并设置好版本控制，在其中添加至少一个配置文件，比如说您的 shell，在其中包含一些自定义设置（可以从设置 $PS1 开始）。

```
mkdir ~/gits/dotfiles
git init ~/gits/dotfiles
# 将本机的配置文件，如 .vimrc/.bashrc/.tmux.conf 等复制进该目录
ls -a ~/gits/dotfiles
```

```
nixgnaf@ubuntu:/tmp/missing$ mkdir ~/gits/dotfiles
nixgnaf@ubuntu:/tmp/missing$ git init ~/gits/dotfiles
Initialized empty Git repository in /home/nixgnaf/gits/dotfiles/.git/
```

1.4.2 (7) 建立一种在新设备进行快速安装配置的方法。最简单的方法是写一个 shell 脚本对每个文件使用 ln -s，也可以使用专用工具在新的虚拟机上测试该安装脚本。

```
#!/bin/bash
files=$(ls -a $1 | grep -E '.[^.]+' | grep -v .git)

for file in `echo $files`; do
    ln -s $1/$file ~/$file
done
```

脚本作用：自动创建软链接，使你在新设备上能快速使用相同的配置文件，而无需手动复制或修改。

files=$(ls -a $1 | grep -E '.[^.]+' | grep -v .git)：列出 $1 目录中的所有文件（排除.git 目录和.、..目录）。$1 是脚本的第一个参数，即配置文件所在的目录。

for file in \echo $files‘：遍历 $files‘ 中的每一个文件。

ln -s $1/$file /$file：为每个文件创建一个软链接，将其链接到主目录（ ）。

1.4.3 (8) 将您现有的所有配置文件移动到项目仓库里。将项目发布到 GitHub。

```
nixgnaf@ubuntu:~/gits/dotfiles/report_3$ cp ~/.bashrc .bashrc
nixgnaf@ubuntu:~/gits/dotfiles/report_3$ ls -a
.  ..  .bashrc
nixgnaf@ubuntu:~/gits/dotfiles/report_3$ git add .
nixgnaf@ubuntu:~/gits/dotfiles/report_3$ commit -m "-"
commit: command not found
nixgnaf@ubuntu:~/gits/dotfiles/report_3$ git commit -m "-"
[master 7e906bf] -
 1 file changed, 118 insertions(+)
 create mode 100644 report_3/.bashrc
nixgnaf@ubuntu:~/gits/dotfiles/report_3$ git push origin master
Counting objects: 3, done.
Delta compression using up to 2 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 287 bytes | 0 bytes/s, done.
Total 3 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To git@github.com:nixgnaf/SDT_report.github.io.git
   cc7364c..7e906bf  master -> master
```

1.5 远端设备

本次实验中在 VMware 上从一台虚拟机 SSH 登录到另一台虚拟机。

1.5.1 (9) 前往 /.ssh/ 并查看是否已经存在 SSH 密钥对。如果不存在，请使用 ssh-keygen -o -a 100 -t ed25519 来创建一个。

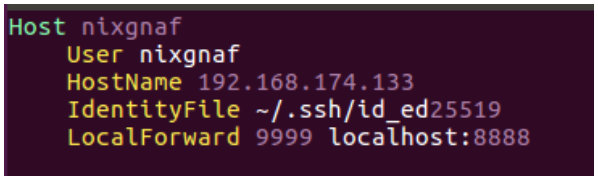
这里曾经配置过 ssh 密钥推送到 github，直接使用。

```
nixgnaf@ubuntu:~/.ssh$ ls
config  id_ed25519  id_ed25519.pub  known_hosts
```

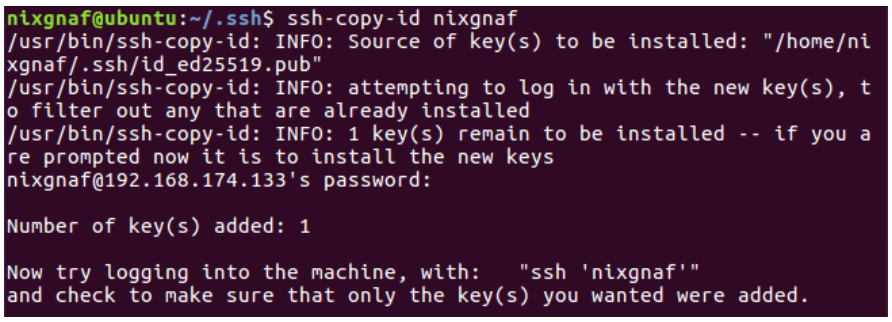
1.5.2 (10) 在.ssh/config 加入下面内容

它将公钥添加到远程主机的/.ssh/authorized_keys 文件中。这允许你通过 SSH 连接到远程主机而无需输入密码。这里将 usename 和 ip 改成目标虚拟机的用户名和 ip 地址。

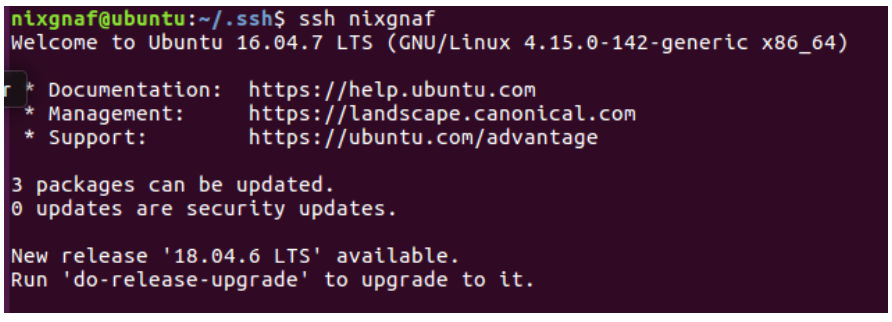
```
Host vm
   User username_goes_here
   HostName ip_goes_here
   IdentityFile ~/.ssh/id_ed25519
   LocalForward 9999 localhost:8888
```



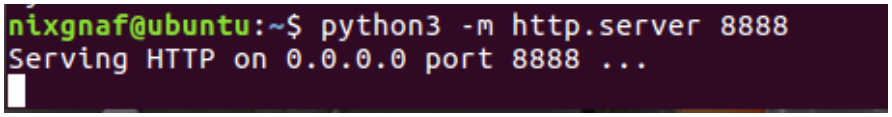
1.5.3 (11) 使用 ssh-copy-id vm 将您的 ssh 密钥拷贝到服务器。



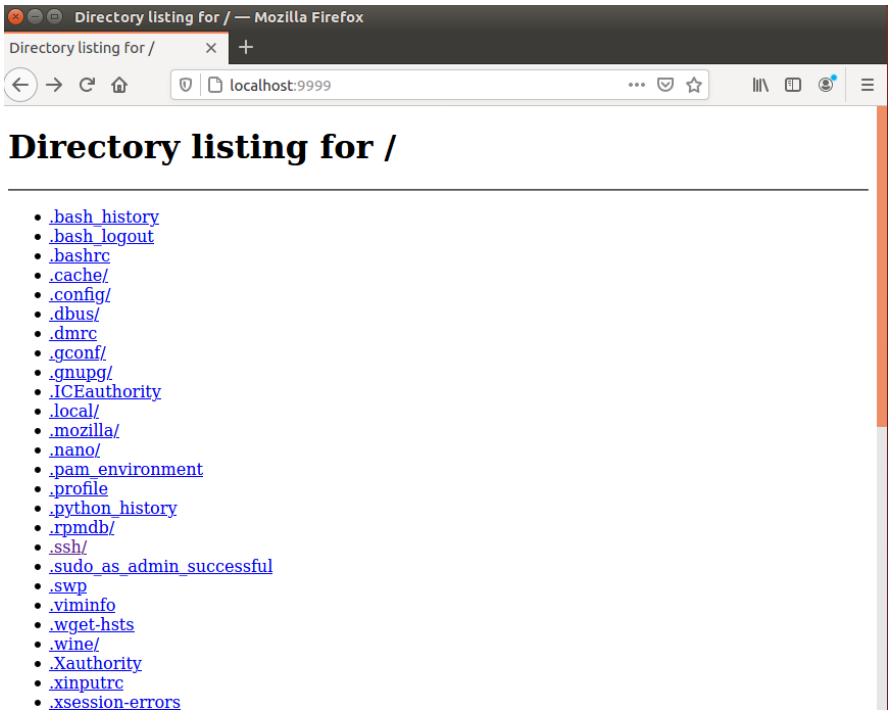
随后可以直接使用 ssh pi 进行免密登录



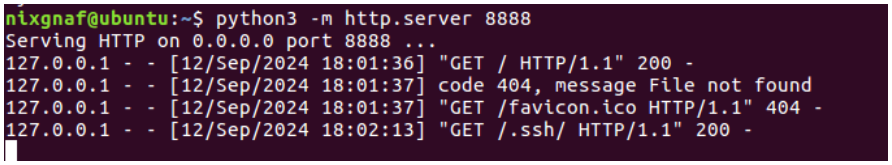
1.5.4 (12) 使用 python -m http.server 8888 在您的虚拟机中启动一个 Web 服务器并通过本机的 http://localhost:9999 访问虚拟机上的 Web 服务器



如图，在源虚拟机上打开浏览器并访问 http://localhost:9999 显示出目标虚拟机当前工作目录的文件列表。



目标虚拟机上记录了源虚拟机的请求。



2 Python 实例

2.1 (13) 逆序输出数组

```
#!/usr/bin/python3

if __name__ == '__main__':
    a = [9,6,5,4,1]
    N = len(a)
    print (a)
    for i in range(len(a) // 2):
        a[i],a[N - i - 1] = a[N - i - 1],a[i]
    print (a)
```

```
nixgnaf@ubuntu:~/Desktop$ ./reverse.py
[9, 6, 5, 4, 1]
[1, 4, 5, 6, 9]
```

2.2 (14) 回文数

```
#!/usr/bin/python3

a = int(input("please input a number:\n"))
x = str(a)
flag = True

for i in range(len(x)//2):
    if x[i] != x[-i - 1]:
        flag = False
        break
if flag:
    print ("%d is a palindrome!" % a)
else:
    print ("%d is not a palindrome!" % a)
```

```
nixgnaf@ubuntu:~/Desktop$ ./palindrome.py
please input a number:
13431
13431 is a palindrome!
```

2.3 (15) 分解质因数

```
#!/usr/bin/python3
```

```
def reduceNum(n):
    print ('{} = '.format(n), end=" ")
    if not isinstance(n, int) or n <= 0 :
        print ('please input a correct number !')
        exit(0)
    elif n in [1] :
        print ('{}'.format(n))
    while n not in [1] :
        for index in range(2, n + 1) :
            if n % index == 0:
                n //= index
                if n == 1:
                    print (index )
                else :
                    print ('{} *'.format(index), end=" ")
                break
reduceNum(90)
reduceNum(100)
```

```
nixgnaf@ubuntu:~/Desktop$ ./reduceNum.py
90 =  2 * 3 * 3 * 5
100 =  2 * 2 * 5 * 5
```

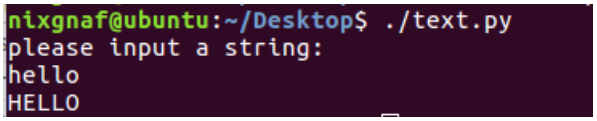
2.4 (16)lambda

```
#!/usr/bin/python

MAXIMUM = lambda x,y :  (x > y) * x + (x < y) * y
MINIMUM = lambda x,y :  (x > y) * y + (x < y) * x

if __name__ == '__main__':
    a = 10
    b = 20
    print ('The largar one is %d' % MAXIMUM(a,b))
    print ('The lower one is %d' % MINIMUM(a,b))
```

```
nixgnaf@ubuntu:~/Desktop$ ./lambda.py
The largar one is 20
The lower one is 10
```

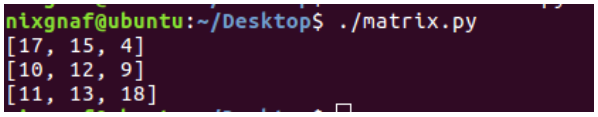
2.5 (17) 大写字符串并保存到文件

```
#!/usr/bin/python

if __name__ == '__main__':
    fp = open('test.txt','w')
    string = raw_input('please input a string:\n')
    string = string.upper()
    fp.write(string)
    fp = open('test.txt','r')
    print fp.read()
    fp.close()
```



2.6 (18) 矩阵相加

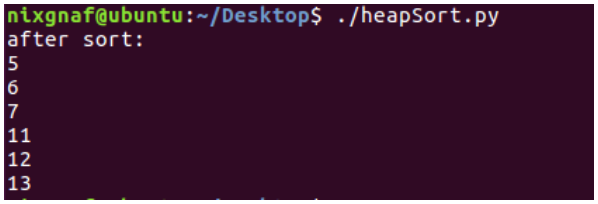
```
#!/usr/bin/python

X = [[12,7,3],
    [4 ,5,6],
    [7 ,8,9]]

Y = [[5,8,1],
    [6,7,3],
    [4,5,9]]

result = [[0,0,0],
          [0,0,0],
          [0,0,0]]

for i in range(len(X)):

   for j in range(len(X[0])):
       result[i][j] = X[i][j] + Y[i][j]

for r in result:
   print(r)
```



2.7 (19) 堆排序

```
#!/usr/bin/python
def heapify(arr, n, i):
    largest = i
    l = 2 * i + 1     # left = 2*i + 1
    r = 2 * i + 2     # right = 2*i + 2

    if l < n and arr[i] < arr[l]:
        largest = l

    if r < n and arr[largest] < arr[r]:
        largest = r

    if largest != i:
        arr[i],arr[largest] = arr[largest],arr[i]

        heapify(arr, n, largest)

def heapSort(arr):
    n = len(arr)

    for i in range(n, -1, -1):
        heapify(arr, n, i)

    for i in range(n-1, 0, -1):
        arr[i], arr[0] = arr[0], arr[i]
        heapify(arr, i, 0)

arr = [ 12, 11, 13, 5, 6, 7]
heapSort(arr)
n = len(arr)
print ("after sort:")
for i in range(n):
    print ("%d" %arr[i])
```

2.8 (20) 归并排序




```
#!/usr/bin/python

def merge(arr, l, m, r):
    n1 = m - l + 1
    n2 = r- m

    L = [0] * (n1)
    R = [0] * (n2)

    for i in range(0 , n1):
        L[i] = arr[l + i]

    for j in range(0 , n2):
        R[j] = arr[m + 1 + j]

    i = 0
    j = 0
    k = l

    while i < n1 and j < n2 :
        if L[i] <= R[j]:
            arr[k] = L[i]
            i += 1
        else:
            arr[k] = R[j]
            j += 1
        k += 1

    while i < n1:
        arr[k] = L[i]
        i += 1
        k += 1

    while j < n2:
        arr[k] = R[j]
        j += 1
```

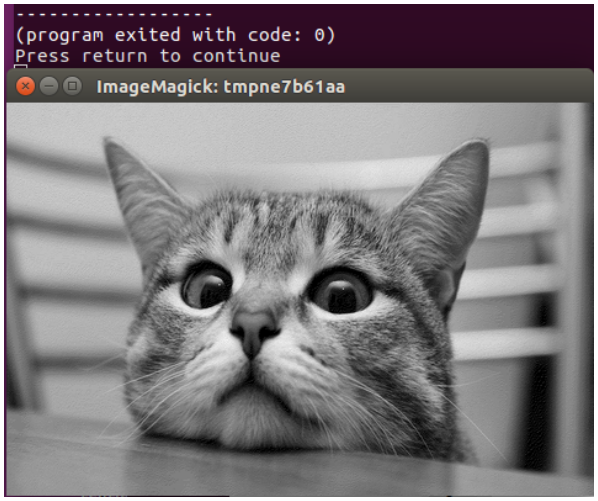
```
        k += 1

def mergeSort(arr,l,r):
    if l < r:


        m = int((l+(r-1))/2)


        mergeSort(arr, l, m)
        mergeSort(arr, m+1, r)
        merge(arr, l, m, r)


arr = [12, 11, 13, 5, 6, 7]
n = len(arr)
print ("array:")
for i in range(n):
    print ("%d" %arr[i]),

mergeSort(arr,0,n-1)
print ("\n\nafter sort:")
for i in range(n):
    print ("%d" %arr[i]),
```

```
nixgnaf@ubuntu:~/Desktop$ ./mergeSort.py
array:
12 11 13 5 6 7

after sort:
5 6 7 11 12 13
```

3 Python 视觉应用实例

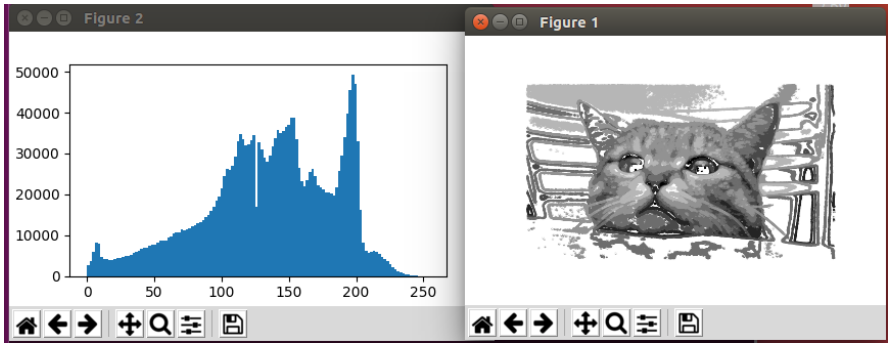
3.0.1 (21)PIL：转换灰度图像

```
from PIL import Image
import os

pil_im = Image.open('cat1.jpg').convert('1')
pil_im.show()
pil_im.save('binarycat.jpg')
```



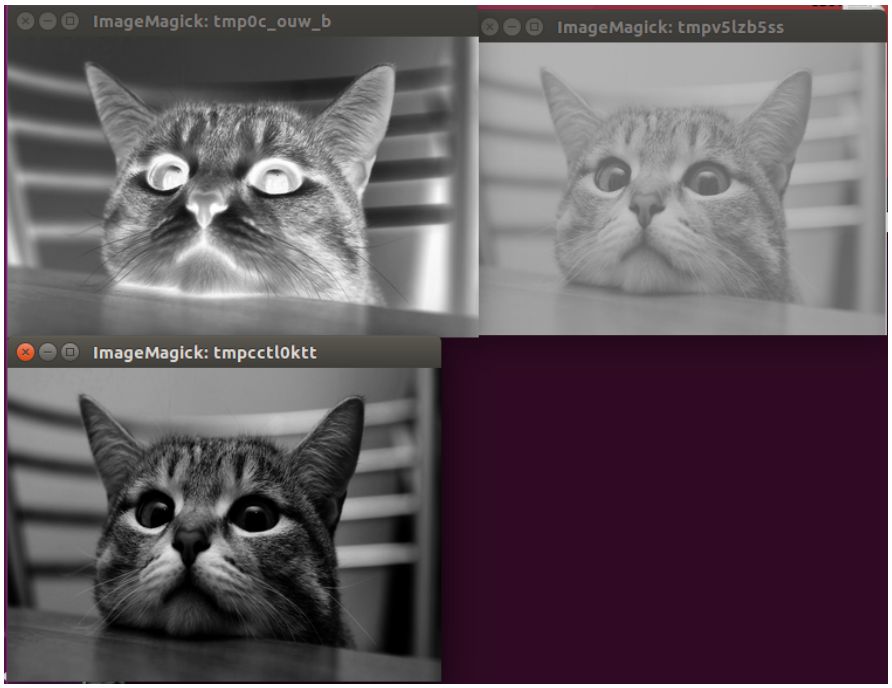
3.0.2 (22)Matplotlib 库：图像轮廓和直方图

```
from PIL import Image
from pylab import *

im = array(Image.open('cat1.jpg').convert('L'))

figure()

gray()

contour(im, origin='image')
axis('equal')
axis('off')

figure()
hist(im.flatten(),128)
show()
```



3.1 (23)Numpy：灰度变换

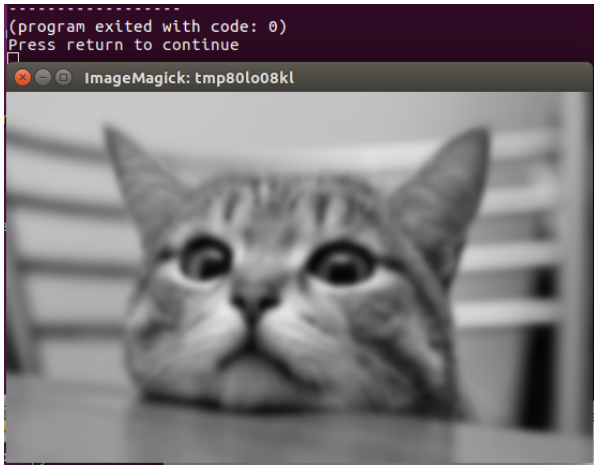
```
from PIL import Image
from numpy import *

im = array(Image.open('cat1.jpg').convert('L'))

im2 = 255 - im
pil_im = Image.fromarray(im2)
pil_im.show()
im3 = (100.0/255) * im + 100
pil_im = Image.fromarray(im3)
pil_im.show()
im4 = 255.0 * (im/255.0)**2
pil_im = Image.fromarray(im4)
pil_im.show()
```



3.2 (24)Scipy：高斯模糊

```
from PIL import Image
from numpy import *
from scipy.ndimage import filters

im = array(Image.open('cat1.jpg').convert('L'))
im2 = filters.gaussian_filter(im,10)
```

```
pil_im = Image.fromarray(im2)

pil_im.show()
```



4 结论和心得

通过本次实验，我学习到了改善 shell 及其他工具的工作流的方法，以及如何使用 SSH 操作远端机器。通过 Python 入门基础，我了解了 Python 语法知识，并在 Python 视觉应用中直观感受到图像处理的过程。