

# 实验报告

彭沛粟

September 16, 2025

## Contents

---

1	练习内容	1
1.1	命令行环境	1
1.2	<i>python</i> 入门基础	7
1.3	<i>python</i> 视觉应用	11
2	解题感悟	14
2.1	命令行环境	14
2.2	<i>python</i> 入门基础	14
2.3	<i>python</i> 视觉应用	14

---

## 1

### 练习内容

---

#### 1.1 命令行环境

1、实例 01：使用 **pgrep** 来查找 **pid** 并使用 **pkill** 结束进程而不需要手动输入 **pid**。

(1) 启动 sleep 任务并放入后台：执行命令 `sleep 10000` 启动一个长时间运行的 `sleep` 任务，然后按下 `Ctrl+z` 将任务暂停并切换到后台，按下 `bg` 让其在后台继续执行<sup>1</sup>

- (2) 用 `pgrep` 查找 `sleep` 进程的 PID
- (3) 用 `pkill` 结束进程<sup>2</sup>

```
peng-peisu@pengpeisu:~/tmp/missing$ sleep 10000
^Z
[1]+  Stopped                  sleep 10000
peng-peisu@pengpeisu:~/tmp/missing$ bg
[1]+ sleep 10000 &
```

Figure 1: 如图所示

```

peng-peisu@pengpeisu:~/tmp/missing$ pgrep -af "sleep 10000"
98713 sleep 10000
peng-peisu@pengpeisu:~/tmp/missing$ pkill -f "sleep 10000"
[1]+  Terminated                 sleep 10000

```

**Figure 2:** 如图所示

**2、实例 02：**如果您希望某个进程结束后再开始另外一个进程，应该如何实现呢？一种方法是使用 **wait** 命令。尝试启动这个休眠命令，然后待其结束后再执行 **ls** 命令。

- (1) 启动 **sleep 60** 并放入后台休息<sup>1</sup>
- (2) 执行命令：**pgrep sleep | wait**，阻塞当前终端，直到当前进程执行完毕
- (3) 当 **sleep 60** 结束后，执行 **ls** 列出当前目录文件<sup>3</sup>

<sup>1</sup> & 表示后台

```

peng-peisu@pengpeisu:~/tmp/missing$ sleep 60 &
[2] 99268
[1] Done                  sleep 60
peng-peisu@pengpeisu:~/tmp/missing$ pgrep sleep | wait
peng-peisu@pengpeisu:~/tmp/missing$ ls
data.txt  debug.sh  marco.sh  semester

```

**Figure 3:** 如图所示

**3、实例 03：**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) 输入命令 **alias dc=cd**
- (2) 使用 **dc** 进入一个目录，可见别名已经设置完成<sup>4</sup>

```

peng-peisu@pengpeisu:~$ alias dc=cd
peng-peisu@pengpeisu:~$ dc tmp/missing
peng-peisu@pengpeisu:~/tmp/missing$ 

```

**Figure 4:** 如图所示

**4、实例 04：**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。

- (1) 输入以上命令，获取到最常用的十条命令
- (2) 尝试使用 **alias** 命令来创建别名
- (3) 检查创建的别名是否能被正常使用<sup>5</sup>

**5、实例 05：**终端多路复用

- (1) 安装 **tmux**<sup>6</sup>
- (2) 使用命令 **tmux** 开始一个新的会话，按照教程尝试拆分窗格<sup>2</sup> (3) 输入命令 **exit** 或按下 **ctrl+z->x** 关闭拆分的窗格<sup>7</sup>
- (3) 执行命令 **unset TMUX**，强制允许嵌套
- (4) 按照教程尝试进行会话处理，创建一个名为 **database** 的新会话

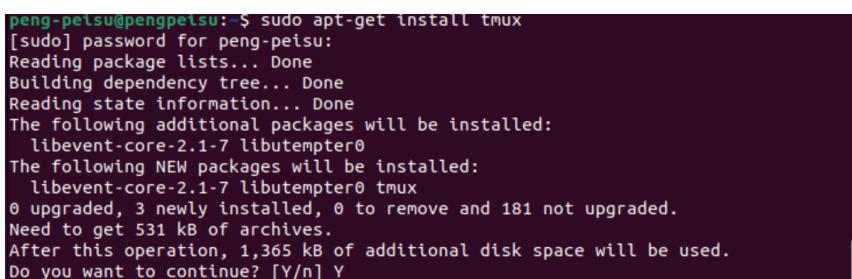
<sup>2</sup> ctrl+z->%



```

peng-peisu@pengpeisu:~$ history | awk '{\$1="";print substr(\$0,2)}' | sort | uniq
-c | sort -n | tail -n 10
3 ./debug.sh
3 ./semester
3 source marco.sh
3 sudo apt install git
3 vim marco.sh
3 vim ~/.vimrc
4 ls -l
5 ls
8 cd
8 cd tmp/missing
peng-peisu@pengpeisu:~$ alias sl=ls
peng-peisu@pengpeisu:~$ ls
Desktop           last-modified.txt      Pictures      tmp
Documents          marco_history.log    Public       Videos
download_models.sh MobileNetSSD_deploy.caffemodel snap        wget-log
Downloads          Music                  Templates
peng-peisu@pengpeisu:~$ sl
Desktop           last-modified.txt      Pictures      tmp
Documents          marco_history.log    Public       Videos
download_models.sh MobileNetSSD_deploy.caffemodel snap        wget-log
Downloads          Music                  Templates
peng-peisu@pengpeisu:~$
```

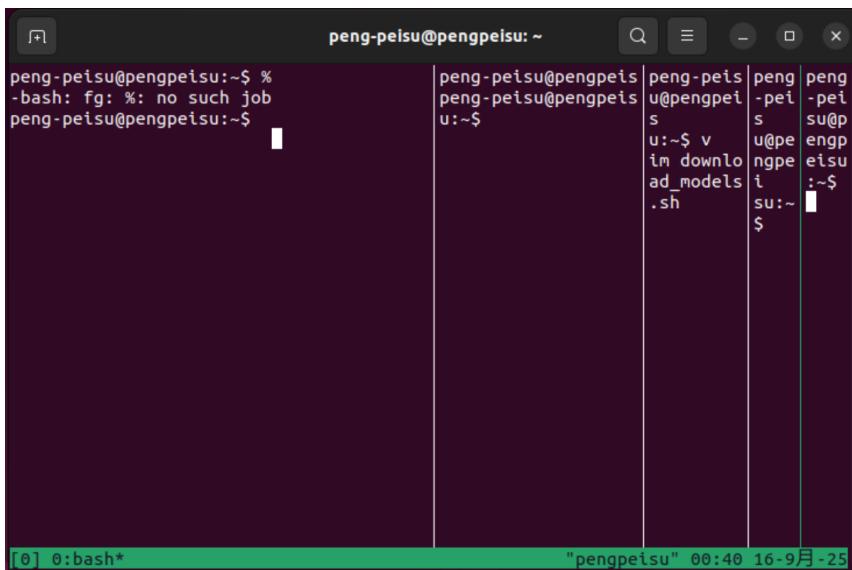
**Figure 5:** 如图所示，这里将 sl 设置为 ls 的别名，用于调取目录



```

peng-peisu@pengpeisu:~$ sudo apt-get install tmux
[sudo] password for peng-peisu:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libevent-core-2.1-7 libutempter0
The following NEW packages will be installed:
  libevent-core-2.1-7 libutempter0 tmux
0 upgraded, 3 newly installed, 0 to remove and 181 not upgraded.
Need to get 531 kB of archives.
After this operation, 1,365 kB of additional disk space will be used.
Do you want to continue? [Y/n] Y
```

**Figure 6:** 如图所示



```

peng-peisu@pengpeisu:~$ %
-bash: fg: %: no such job
peng-peisu@pengpeisu:~$
```

The terminal window has multiple vertical panes. The leftmost pane shows the command history and an error message. The rightmost pane shows a complex command history with many entries and some redacted parts.

**Figure 7:** 如图所示，是拆分过多次的窗格

(5) 重命名这个会话

(6) 执行命令 tmux ls 列出当前所有会话，可看见刚刚新建的会话 database<sup>8</sup>

```
peng-peisu@pengpeisu:~$ tmux rename-session -t 0 database
duplicate session: database
peng-peisu@pengpeisu:~$ tmux ls
0: 1 windows (created Tue Sep 16 01:00:14 2025) (attached)
database: 1 windows (created Tue Sep 16 01:02:04 2025) (attached)
peng-peisu@pengpeisu:~$
```

**Figure 8:** 如图所示，强制允许嵌套后新建会话，重命名后列出当前存在的所有会话

(7) 学习一些 tmux 快捷键：

<C-b> c 创建一个新的窗口

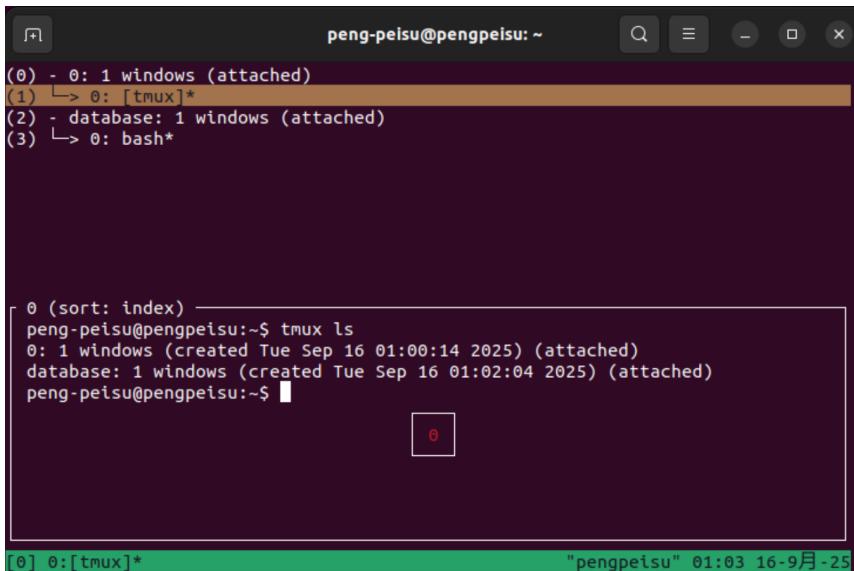
<C-d> 关闭

<C-b> N 跳转到第 N 个窗口

<C-b> p 切换到前一个窗口，<C-b> n 切换到下一个窗口

<C-b> , 重命名当前窗口

<C-b> w 列出当前所有窗口<sup>9</sup>



**Figure 9:** 如图所示，是使用 ctrl+b->w 后列出的当前所有窗口

## 6、实例 06：配置 ssh 远程连接

(1) 虚拟机设置为 NAT 模式<sup>3</sup>

<sup>3</sup> 虚拟机-> 设置-> 网络适配器-> 选择 NAT 模式

(2) 检查虚拟机是否配置 SSH

(3) 计算机报错，说明尚未配置，先配置 SSH<sup>10</sup>

(4) 重启 ssh 服务器，开启默认端口号，按教程修改好后再次重启<sup>11</sup><sup>12</sup>

(5) 执行命令 ifconfig，获取 ip 地址<sup>13</sup>

(6) 重新设置密码<sup>14</sup>

```

peng-peisu@pengpeisu: $ /etc/init.d/iptables start
bash: /etc/init.d/iptables: No such file or directory
peng-peisu@pengpeisu: $ sudo apt install openssh-server
[sudo] password for peng-peisu:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  ncurses-term openssh-sftp-server ssh-import-id
Suggested packages:
  molly-guard monkeysphere ssh-askpass
The following NEW packages will be installed:
  ncurses-term openssh-server openssh-sftp-server ssh-import-id
0 upgraded, 4 newly installed, 0 to remove and 181 not upgraded.
Need to get 751 kB of archives.
After this operation, 6,050 kB of additional disk space will be used.
Do you want to continue? [Y/n]

```

**Figure 10:** 如图所示，安装 SSH 服务

```

peng-peisu@pengpeisu: $ sudo service ssh restart
peng-peisu@pengpeisu: $ sudo vi /etc/ssh/sshd_config
peng-peisu@pengpeisu: $ sudo service ssh restart

```

**Figure 11:** 如图所示

```

Port 22
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::

#HostKey /etc/ssh/ssh_host_rsa_key
#HostKey /etc/ssh/ssh_host_ecdsa_key
#HostKey /etc/ssh/ssh_host_ed25519_key

# Ciphers and keying
#RekeyLimit default none

# Logging
#SyslogFacility AUTH
#LogLevel INFO

# Authentication:

#LoginGraceTime 2m
PermitRootLogin yes
StrictModes yes

```

**Figure 12:** 如图所示，进行修改

```

peng-peisu@pengpeisu: $ ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
        inet 192.168.117.129 netmask 255.255.255.0 broadcast 192.168.117.255
              inet6 fe80::df5f:f588:de39:9efc prefixlen 64 scopeid 0x20<link>
                ether 00:0c:29:fd:6f:df txqueuelen 1000 (Ethernet)
                  RX packets 143714 bytes 188683077 (188.6 MB)
                  RX errors 0 dropped 0 overruns 0 frame 0
                  TX packets 41302 bytes 2682089 (2.6 MB)
                  TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
        inet 127.0.0.1 netmask 255.0.0.0
              inet6 ::1 prefixlen 128 scopeid 0x10<host>
                loop txqueuelen 1000 (Local Loopback)
                  RX packets 1525 bytes 148175 (148.1 KB)
                  RX errors 0 dropped 0 overruns 0 frame 0
                  TX packets 1525 bytes 148175 (148.1 KB)
                  TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

**Figure 13:** 如图所示，安装 SSH 服务

```

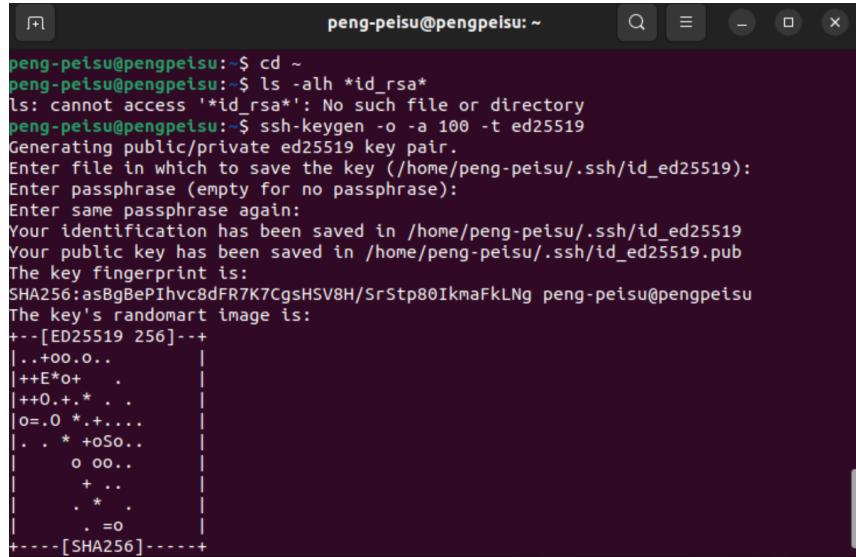
peng-peisu@pengpeisu: $ sudo passwd root
[sudo] password for peng-peisu:
New password:
Retype new password:
passwd: password updated successfully

```

**Figure 14:** 如图所示

7、实例 07：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。

- (1) 执行命令发现并不存在 SSH 密钥
- (2) 按照教程创建一个 SSH 密钥[15](#)



```

peng-peisu@pengpeisu:~$ cd ~
peng-peisu@pengpeisu:~$ ls -alh *id_rsa*
ls: cannot access '*id_rsa*': No such file or directory
peng-peisu@pengpeisu:~$ ssh-keygen -o -a 100 -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/peng-peisu/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/peng-peisu/.ssh/id_ed25519
Your public key has been saved in /home/peng-peisu/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:asBgBePIhvc8dFR7K7CgsHSV8H/SrStp80IkmaFkLNg peng-peisu@pengpeisu
The key's randomart image is:
+--[ED25519 256]--+
|...+oo.o...
|++E*o+ .
|++0+.+* . .
|o=.0 * .+.....
|. . * +oS0..
|       o oo...
|       + ...
|       . *
|       . =o
+---[SHA256]---
```

Figure 15: 如图所示

- (3) 执行命令 `ls /.ssh` 可见刚刚创建的密钥[16](#)

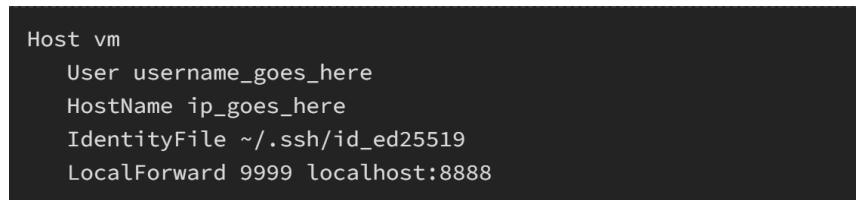


```

peng-peisu@pengpeisu:~$ ls ~/.ssh
id_ed25519  id_ed25519.pub
```

Figure 16: 如图所示

8、实例 08：在`.ssh/config` 加入下面内容[17](#):



```

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

Figure 17: 添加如图内容

- (1) 用 vim 打开该文件，向其中加入教程给出的内容，将里面的 `username` 和 `ip` 地址改成自己的[18](#)
- (2) 保存并退出

9、实例 09：使用 `ssh-copy-id vm` 将您的 `ssh` 密钥拷贝到服务器。

- (1) 执行命令，将密钥拷贝到服务器

```

1 Host vm
2   User peng-peisu
1   HostName 192.168.117.129
2   IdentityFile ~/.ssh/id_ed25519
3   LocalForward 9999 localhost:8888

```

**Figure 18:** 如图所示

(2) 因为第一次连接，SSH 会进行“首次连接安全认证”，输入 yes 即可<sup>19</sup>

```

peng-peisu@pengpeisu: ~ ssh-copy-id peng-peisu@192.168.117.129
The authenticity of host '192.168.117.129 (192.168.117.129)' can't be established.
ED25519 key fingerprint is SHA256:OMw2OKw2WeunITD8xkhfZhhxRu9t37pKIKsPb0wmaZ8.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter
out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompt
ed now it is to install the new keys
peng-peisu@192.168.117.129's password:

Number of key(s) added: 1

Now try logging into the machine, with: "ssh 'peng-peisu@192.168.117.129'"
and check to make sure that only the key(s) you wanted were added.

```

**Figure 19:** 如图所示

## 1.2 python 入门基础

### 1、实例 10：python 中文编码

(1)Python 中默认的编码格式是 ASCII 格式，在没修改编码格式时无法正确打印汉字，所以在读取中文时会报错。

(2)要解决这个问题需要在文件开头加入 `# -*- coding: UTF-8 -*-` 或者 `# coding=utf-8`<sup>20</sup>

```

# coding=utf-8
print("python入门基础学习")

```

**Figure 20:** 如图所示，注意一定要加 #，作为注释行，否则会被 python 认为是代码的一部分

### 2、实例 11：python 的行与缩进

(1)Python 的代码块不使用大括号来控制类，函数以及其他逻辑判断。

(2)缩进的空白数量是可变的，但是所有代码块语句必须包含相同的缩进空白数量，通常以四个空格为标准。

(3)若缩进错误，python 会报错<sup>21</sup>

### 3、实例 12：python 变量赋值

(1)python 可以同时为多个变量赋值

(2)python 不需要声明数据类型，可以直接赋值，且必须赋值，它通过赋值自动确定变量数据类型<sup>22</sup>

```

1 #!/usr/bin/python
2 # -*- coding: UTF-8 -*-
3
4 if True:
5     print ("Answer")
6     print ("True")
7 else:
8     print ("Answer")
9 print ("False")

```

The screenshot shows the PyCharm IDE interface. The project tree on the left shows a folder named 'python' containing a file 'print\_demo.py'. The code editor window displays the same Python script. Below the editor is the terminal window, which shows the command 'python print\_demo.py' being run. The output indicates a syntax error: 'IndentationError: unindent does not match any outer indentation level'. The error occurs at line 9, where the closing brace of the 'else' block is located.

**Figure 21:** 如图所示，`print ("False")` 没有空出四格，缩进错误导致了报错

```

1 #!/usr/bin/python
2 # -*- coding: UTF-8 -*-
3
4 a,b,c=1,2,"John"
5 print(a,b,c)
6
7 str="哈哈哈"
8 double=100.0
9 print(str,double)

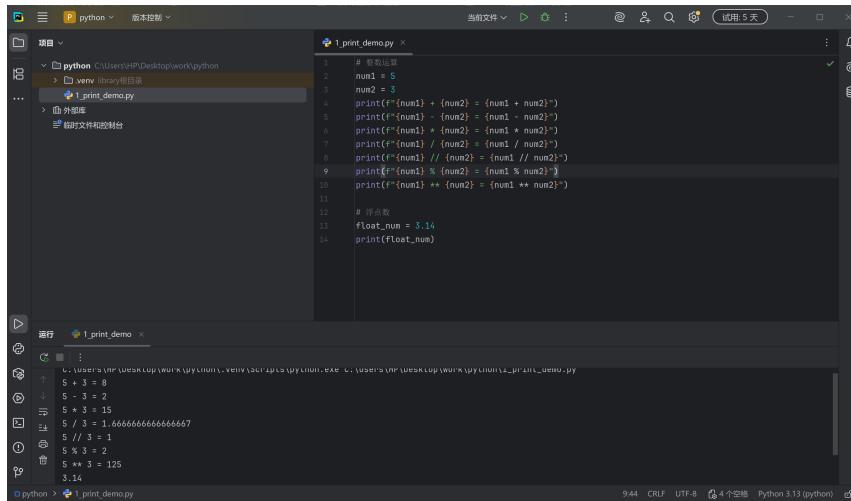
```

This screenshot shows the PyCharm IDE after the fix from Figure 21. The code now correctly prints the variables 'a', 'b', and 'c' followed by the string '哈哈哈' and the float '100.0'. The terminal output shows the expected results: '1 2 John' and '哈哈哈 100.0'.

**Figure 22:** 如图所示

#### 4、实例 13：数字类型相关练习

(1)python 中的部分运算符与 c 和 cpp 中不同，比如/就是除法，不会因为分子是正数就自动取整，//才是整除。再比如多了幂运算的运算符，是`**23`



```

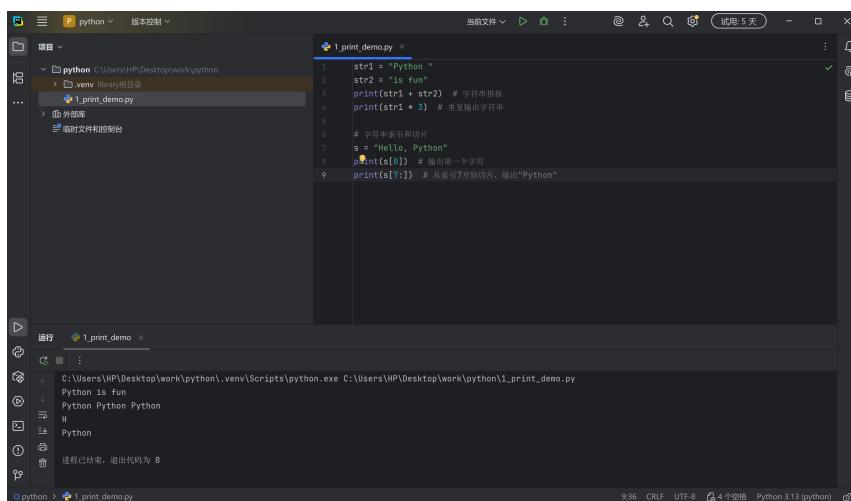
1. print_demo.py
1 # 整数运算
2 num1 = 5
3 num2 = 3
4 print(f"num1 + (num2) = {num1 + num2}")
5 print(f"num1 - (num2) = {num1 - num2}")
6 print(f"num1 * (num2) = {num1 * num2}")
7 print(f"num1 / (num2) = {num1 / num2}")
8 print(f"num1 // (num2) = {num1 // num2}")
9 print(f"num1 % (num2) = {num1 % num2}")
10 print(f"num1 ** (num2) = {num1 ** num2}")
11
12 # 浮点数
13 float_num = 3.14
14 print(float_num)

```

Figure 23: 如图所示

#### 5、实例 14：字符串相关练习

(1)python 中不需要引用什么头文件就可以实现字符串的拼接、重复、定位和切片`24`



```

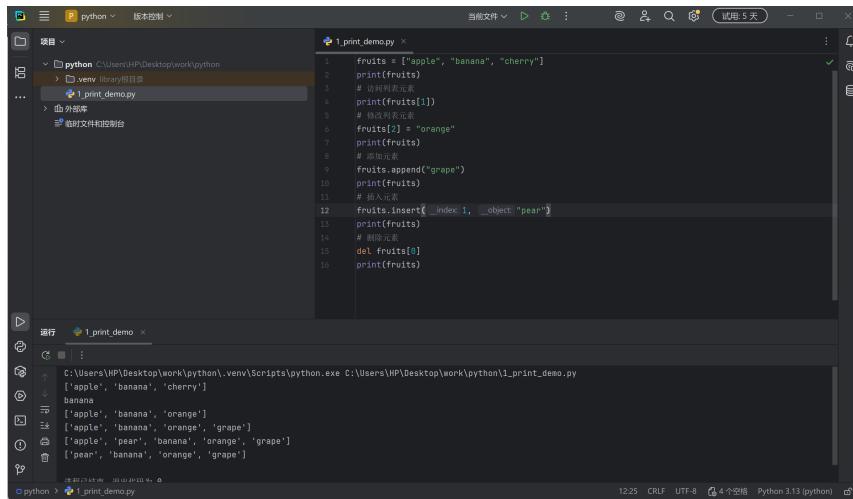
1. print_demo.py
1 str1 = "Python"
2 str2 = "is fun"
3 print(str1 + str2) # 字符串拼接
4 print(str1 * 3) # 重复输出字符串
5
6 # 字符串索引和切片
7 s = "Hello, Python"
8 print(s[0]) # 输出第一个字符
9 print(s[7:]) # 从索引7开始切片，输出"Python"

```

Figure 24: 如图所示

## 6、实例 15：列表相关练习

(1) 类似于 cpp 中 vector 类的部分函数，可以在指定位置插入元素、删除某一元素等等<sup>25</sup>



```

1 print_demo.py
1 fruits = ["apple", "banana", "cherry"]
2 print(fruits)
3 # 访问列表元素
4 print(fruits[1])
5 # 修改列表元素
6 fruits[2] = "orange"
7 print(fruits)
8 # 添加元素
9 fruits.append("grape")
10 print(fruits)
11 # 插入元素
12 fruits.insert(1, "pear")
13 print(fruits)
14 # 删除元素
15 del fruits[0]
16 print(fruits)

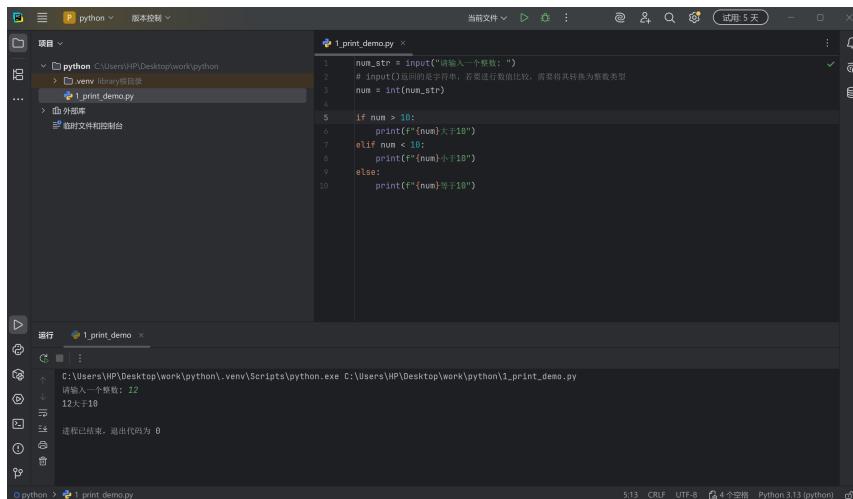
运行 1_print_demo.x
C:\Users\HP\Desktop\work\python\.venv\Scripts\python.exe C:\Users\HP\Desktop\work\python\1_print_demo.py
['apple', 'banana', 'cherry']
['apple', 'orange']
['apple', 'orange', 'grape']
['apple', 'pear', 'banana', 'orange', 'grape']
['pear', 'banana', 'orange', 'grape']

```

Figure 25: 如图所示

## 7、实例 16：条件语句相关练习

- (1) python 中的输入和 cpp、c 中的输入不太一样，使用 `input` 函数，但注意 `input` 函数返回的是字符串，需要转换成整型  
 (2) 条件语句和 c、cpp 的区别在于不需要大括号包裹，但是需要通过缩进区分每一部分，`else if` 写作 `elif`<sup>26</sup>



```

1 print_demo.py
1 num_str = input("请输入一个整数：")
2 # input()返回的是字符串，若要进行数值比较，需要将其转换为整数类型
3 num = int(num_str)
4
5 if num > 10:
6     print(f"{num}大于10")
7 elif num < 10:
8     print(f"{num}小于10")
9 else:
10     print(f"{num}等于10")

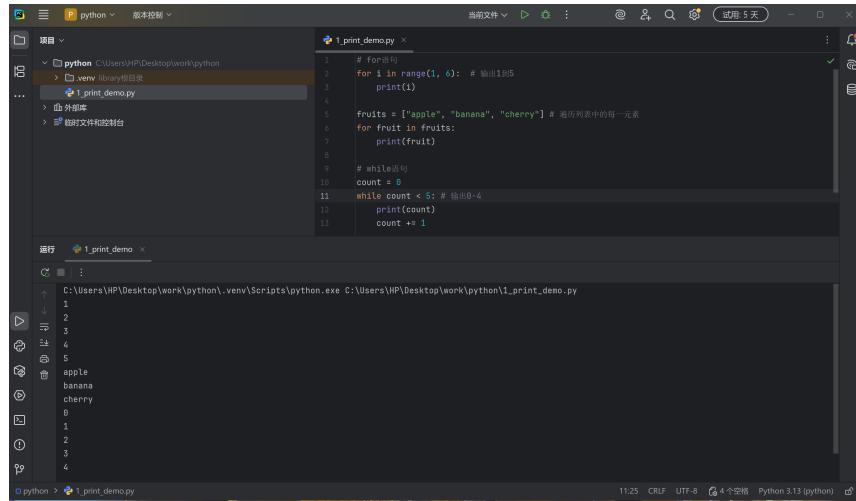
运行 1_print_demo.x
请输入一个整数： 12
12大于10
进程已结束，退出代码为 0

```

Figure 26: 如图所示

## 7、实例 17：循环语句相关练习

(1) python 中的循环关键词也是 for 和 while，但是注意没有 do...while 语句<sup>27</sup>



```

1 # for语句
2 for i in range(1, 6): # 输出1到5
3     print(i)
4
5 fruits = ["apple", "banana", "cherry"] # 遍历列表中的每一元素
6 for fruit in fruits:
7     print(fruit)
8
9 # while语句
10 count = 0
11 while count < 5: # 循环0~4
12     print(count)
13     count += 1

```

Figure 27: 如图所示

## 1.3 python 视觉应用

### 1、实例 18：使用 Pillow 来增加图像对比度

(1)<sup>28</sup>安装 Pillow 库<sup>4</sup>

<sup>4</sup> 执行命令: pip install pillow

(2) 新建一个 python 文件，将教程给出的代码写入，然后保存退出<sup>29</sup>

(3) 执行该脚本之后可以看见目标图片的对比度已经得到了加强<sup>30</sup>

```

peng-peltsu@pengpeltsu:~$ pip install pillow
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: pillow in /usr/lib/python3/dist-packages (9.0.1)
peng-peltsu@pengpeltsu:~$ vim image_process.py
peng-peltsu@pengpeltsu:~$ python3 image_process.py

```

Figure 28: 如图所示，新建并打开文件，执行文件

```

1 from PIL import Image,ImageEnhance
2 img_original = Image.open("dark.jpg")
3 img_original.show("Original Image")
4 img = ImageEnhance.Contrast(img_original)
5 img.enhance(3.8).show("Image With More Contrast")

```

Figure 29: 如图所示，为写入的代码，dark.jpg 是目标图片，写入该文件的位置路径

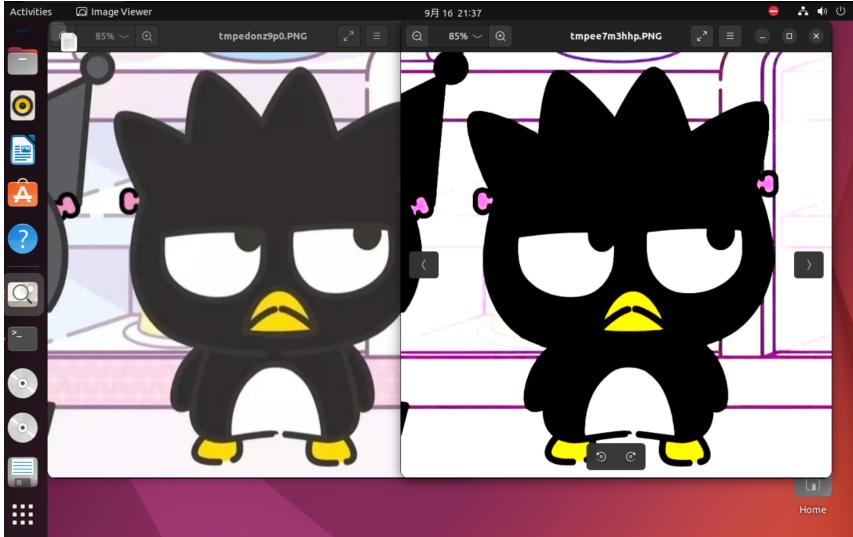
### 2、实例 19：使用 NumPy 库操纵图像的 RGB 值

(1)<sup>31</sup>安装 Numpy 库<sup>5</sup>

<sup>5</sup> 执行命令: pip install numpy

(2) 新建一个 python 文件，将教程给出的代码写入，然后保存退出<sup>32</sup>

(3) 执行该脚本之后可以看见目标图片的 RGB 值已经被调整了<sup>33</sup>



**Figure 30:** 如图所示，为加强对比度后的图片

```

peng-petsu@pengpetsu:~$ pip install numpy
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: numpy in ./local/lib/python3.10/site-packages (2
.2.6)
peng-petsu@pengpetsu:~$ vim image_2.py
peng-petsu@pengpetsu:~$ python3 image_2.py
/home/peng-petsu/image_2.py:3: DeprecationWarning: __array__ implementation does
n't accept a copy keyword, so passing copy=False failed. __array__ must implement
'dtype' and 'copy' keyword arguments. To learn more, see the migration guide h
ttps://numpy.org/devdocs/numpy_2_0_migration_guide.html#adapting-to-changes-in-t
he-copy-keyword
    img = np.array(Image.open('dark.jpg'))

```

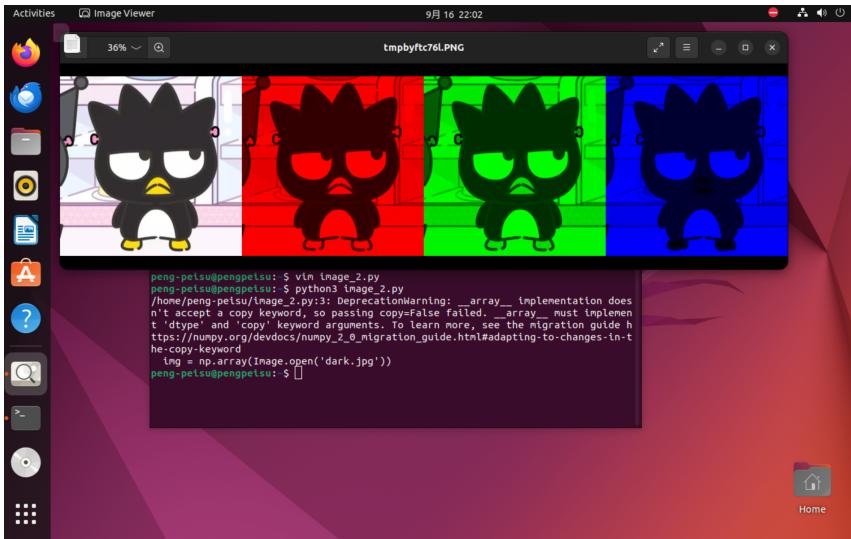
**Figure 31:** 如图所示，安装 Numpy 库，新建并打开文件，执行文件

```

1 from PIL import Image
2 import numpy as np
3 img = np.array(Image.open('dark.jpg'))
4 img_red = img.copy()
5 img_red[:, :, (1, 2)] = 0
6 img_green = img.copy()
7 img_green[:, :, (0, 2)] = 0
8 img_blue = img.copy()
9 img_blue[:, :, (0, 1)] = 0
10 img_ORGB = np.concatenate((img, img_red, img_green, img_blue), axis=1)
11 img_converted = Image.fromarray(img_ORGB)
12 img_converted.show() ## Combine Image Contains all four images

```

**Figure 32:** 如图所示，为写入的代码，dark.jpg 是目标图片，写入该文件的位置路径



**Figure 33:** 如图所示，为调整 RGB 值后的图片

### 3、实例 20：使用 Scipy 库对图像高斯模糊

- (1) 先安装 Scipy 库<sup>34</sup><sup>6</sup>
- (2) 新建文件，将教程代码复制进去，由于 scipy.misc 模块过时且功能失效，所以需要用 ascent = Image.open("dark.jpg") 来替换原来的 ascent = misc.ascent()，等等。<sup>35</sup>
- (3) 执行脚本，可见图片已被高斯模糊<sup>36</sup>

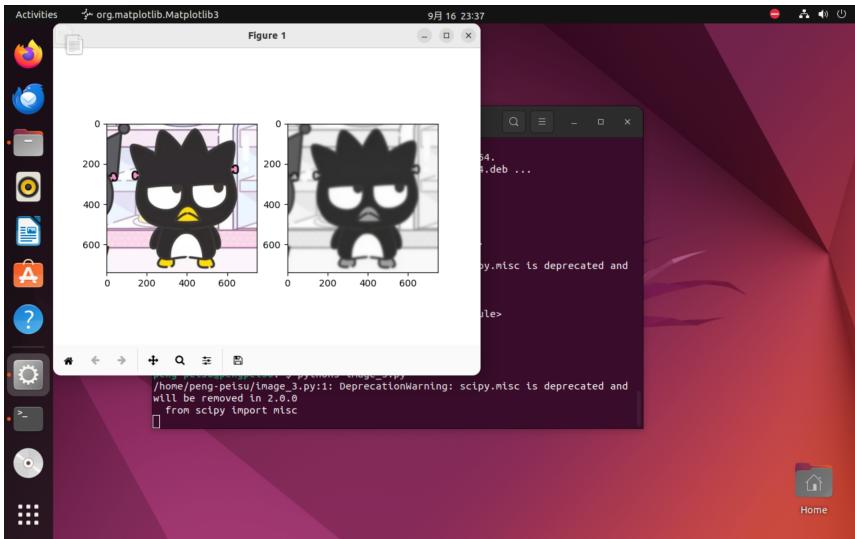
<sup>6</sup> 执行命令：pip install scipy

```
peng-petsu@pengpetsu:~$ pip install scipy
Defaulting to user installation because normal site-packages is not writeable
Collecting scipy
  Downloading scipy-1.15.3-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (37.7 MB)
    37.7/37.7 MB 24.7 kB/s eta 0:00:00
Requirement already satisfied: numpy<2.5,>=1.23.5 in ./local/lib/python3.10/site-packages (from scipy) (2.2.6)
Installing collected packages: scipy
Successfully installed scipy-1.15.3
peng-petsu@pengpetsu:~$ vim image_3.py
```

**Figure 34:** 如图所示，安装 Scipy 库

```
1 from scipy import misc
2 from PIL import Image
3 from scipy.ndimage import gaussian_filter
4 import matplotlib.pyplot as plt
5
6 fig = plt.figure()
7 plt.gray() # show the filtered result in grayscale
8 ax1 = fig.add_subplot(121) # left side
9 ax2 = fig.add_subplot(122) # right side
10 ascent = Image.open("dark.jpg")
11 result = gaussian_filter(ascent, sigma=5)
12 ax1.imshow(ascent)
13 ax2.imshow(result)
14 plt.show()
```

**Figure 35:** 如图所示，为写入的代码，dark.jpg 是目标图片，写入该文件的位置路径



**Figure 36:** 如图所示，为高斯模糊后的图片

## 2

### 解题感悟

---

#### 2.1 命令行环境

1、通过对命令行环境的学习，更加加深了对命令行操作高效的体会。比如使用 **pgrep** 和 **pkill** 来管理后台任务进程就能批量操作，并且避免了手动查找输入 **PID**，大大提高了工作效率。

2、还学习到了很多实用的命令行环境的功能，任务控制、终端通路复用、文件配置、远程设备连接等等。

#### 2.2 python 入门基础

1、配置好了 **python** 环境

2、初步学习到了 **python** 的基本语法，也与先前学习过的 **c**、**cpp** 语言进行了对比，可以很明显的感受出其中的异同，最明显的感受就是 **python** 中对缩进的应用。

3、感受到了很多 **python** 相较于其他编程语言的便利之处，比如声明变量时不需要声明其数据类型。

#### 2.3 python 视觉应用

1、通过许多不同的图像处理库，**python** 可以利用很简洁的代码对图像进行编辑处理，高效便捷。