

系统开发工具基础实验报告（三）

姓名：陈怡冰 学号：23020007007

September 11, 2024

目录

1 命令行环境	2
1.1 实验内容	2
1.1.1 远端设备	2
1.1.2 终端多路复用	2
1.1.3 任务控制	2
1.1.4 别名	3
1.1.5 配置文件	3
1.2 实验结果	3
1.2.1 远端设备	3
1.2.2 终端多路复用	7
1.2.3 任务控制	7
1.2.4 别名	7
1.2.5 配置文件	8
1.3 实验代码及关键步骤	9
1.3.1 链接到 Github	9
1.3.2 创建阿里云服务器并远程链接	9
1.3.3 终端多路复用	10
2 Python 入门基础	12
2.1 实验内容	12
2.2 实验结果	12
2.3 实验代码及关键步骤	14
3 Python 视觉应用	17
3.1 实验内容	17
3.2 实验结果	17
3.3 实验代码及关键步骤	20
3.4 实验中遇到的问题及解决方案	23
4 实验心得	24
5 我的 Github 仓库网址	25

实验部分 1 命令行环境

1.1 实验内容

1.1.1 远端设备

1. 连接到自己的 Github。
2. 创建阿里云服务器，并远程链接。
3. 使用 scp 将本地文件复制到阿里云服务器。
4. 使用 scp 将本地文件夹中的文件复制到阿里云服务器
5. 使用 rsync 将本地目录复制到阿里云服务器。
6. 使用 rsync 将本地文件与服务器文件同步。
7. 在.ssh/config 加入以下内容：

```
1      Host vm
2      User username_goes_here
3      HostName ip_goes_here
4      IdentityFile ~/.ssh/id_ed25519
5      LocalForward 9999 localhost:8888
```

- 使用 ssh-copy-id vm 将您的 ssh 密钥拷贝到服务器。
- 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
- 使用 sudo vim /etc/ssh/sshd_config 编辑 SSH 服务器配置，通过修改 PasswordAuthentication 的值来禁用密码验证。通过修改 PermitRootLogin 的值来禁用 root 登录。然后使用 sudo service sshd restart 重启 ssh 服务器，然后重新尝试。
- (附加题) 在虚拟机中安装 mosh 并启动连接。然后断开服务器/虚拟机的网络适配器。mosh 可以恢复连接吗？
- (附加题) 查看 ssh 的 -N 和 -f 选项的作用，找出在后台进行端口转发的命令是什么？

1.1.2 终端多路复用

1. 创建一个新的窗口，并在后台运行，即使终端关闭以不中断。
2. 在不同终端之间共享会话。

1.1.3 任务控制

1. 我们可以使用类似 ps aux | grep 这样的命令来获取任务的 pid，然后您可以基于 pid 来结束这些进程。但我们其实有更好的方法来做这件事。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。(提示：：使用 -af 标记)。

1.1.4 别名

1. 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。
2. 执行

```
1 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。

1.1.5 配置文件

1. 为您的配置文件新建一个文件夹，并设置好版本控制
2. 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）。
3. 建立一种在新设备进行快速安装配置的方法（无需手动操作）。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
4. 在新的虚拟机上测试该安装脚本。
5. 将您现有的所有配置文件移动到项目仓库里。
6. 将项目发布到 GitHub。

1.2 实验结果

1.2.1 远端设备

1. 实例 1
成功连接到自己的 Github。

```
daisy@daisy-virtual-machine:~$ ssh -T git@github.com  
Hi ChenFirstIce! You've successfully authenticated, but GitHub doe  
s not provide shell access.
```

图 1.1: 实例 1 结果图

2. 实例 2
成功创建阿里云服务器，并通过端口 22[ssh] 远程连接到服务器镜像。

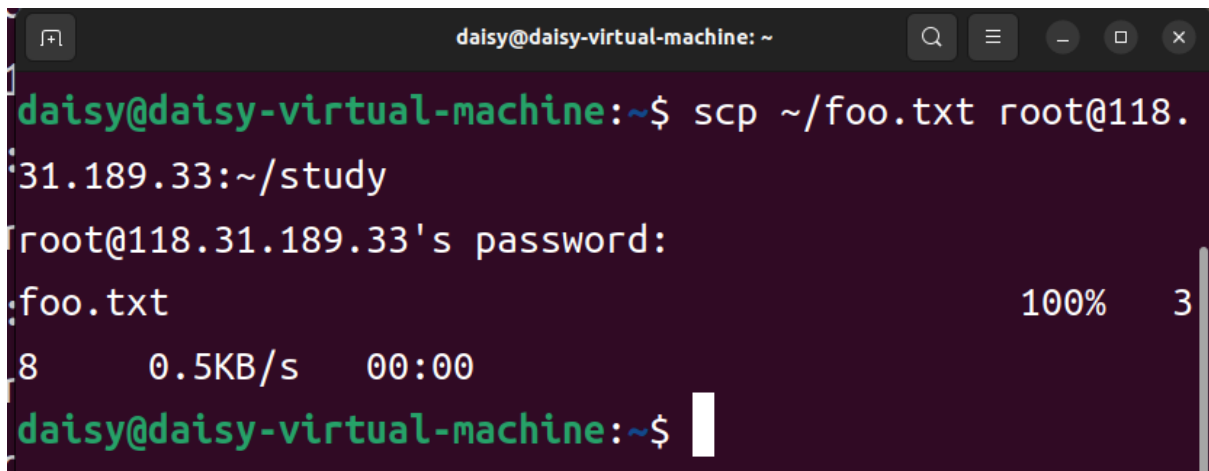
```
Welcome to Alibaba Cloud Elastic Compute Service !  
  
Last login: Wed Sep 11 11:03:15 2024 from 8.139.99.239  
root@iZbp1f9vj534drdxate2gqZ:~#
```

图 1.2: 实例 2 结果图

3. 实例 3
使用

```
1 scp ~/foo.txt root@118.31.189.33:~/study
```

将文件`foo.txt`复制到远端设备中的文件夹`study`



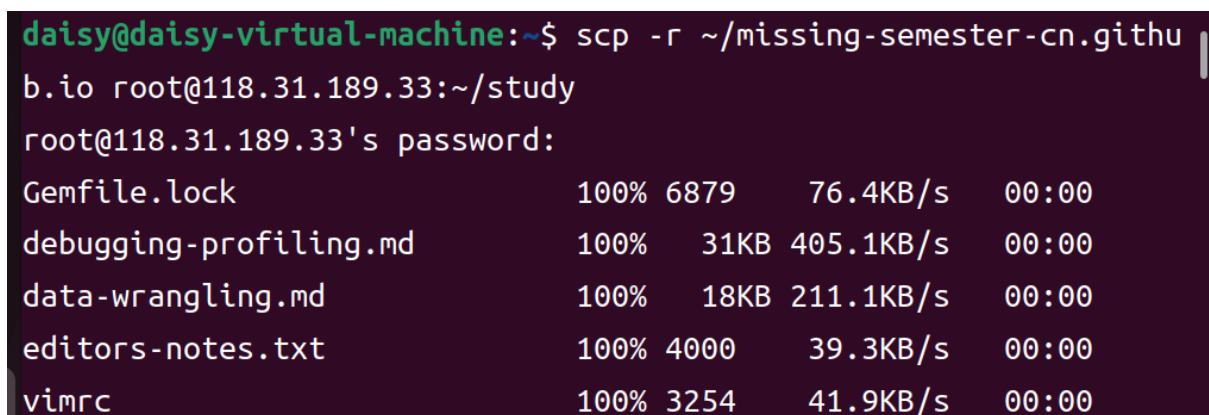
```
daisy@daisy-virtual-machine: ~  
daisy@daisy-virtual-machine:~$ scp ~/foo.txt root@118.31.189.33:~/study  
root@118.31.189.33's password:  
foo.txt 100% 3  
8 0.5KB/s 00:00  
daisy@daisy-virtual-machine:~$
```

图 1.3: 实例 3 结果图

4. 实例 4 使用

```
1 scp -r ~/missing-semester-cn.github.io root@118.31.189.33:~/study
```

将文件夹中的文件以递归复制到远端设备上。



```
daisy@daisy-virtual-machine:~$ scp -r ~/missing-semester-cn.github.io root@118.31.189.33:~/study  
root@118.31.189.33's password:  
Gemfile.lock 100% 6879 76.4KB/s 00:00  
debugging-profiling.md 100% 31KB 405.1KB/s 00:00  
data-wrangling.md 100% 18KB 211.1KB/s 00:00  
editors-notes.txt 100% 4000 39.3KB/s 00:00  
vimrc 100% 3254 41.9KB/s 00:00
```

图 1.4: 实例 4 结果图

5. 实例 5

```
1 rsync -avz ~/try root@118.31.189.33:~/study
```

将文件夹中的文件以归档模式复制到远端设备上。
使用`rsync`比`scp`的效率更高

```
daisy@daisy-virtual-machine:~$ rsync -avz ~/try root@118.31.189.33:~/study
root@118.31.189.33's password:
sending incremental file list
try/
try/main.py
try/.git/
try/.git/HEAD
```

图 1.5: 实例 5 结果图

6. 实例 6 使用

```
1 rsync -av --delete ~/try root@47.97.244.53:~/study
```

将文件夹中的文件以同步到远端设备上。
在同步之前删除本地文件夹 *try* 中的 *main.py* 文件。

```
daisy@daisy-virtual-machine:~$ rsync -av --delete ~/try root@47.97.244.53:~/study
root@47.97.244.53's password:
sending incremental file list
deleting try/main.py
try/

sent 854 bytes  received 46 bytes  120.00 bytes/sec
total size is 23,870  speedup is 26.52
```

图 1.6: 实例 6 结果图

7. 实例 7(1) 我的配置是阿里云服务器

```
1 Host ali
2 User root
3 HostName 47.97.244.53
4 IdentityFile ~/.ssh/id_ed25519
5 LocalForward 9999 localhost:8888
```

阿里云关闭再开启后公网 IP 改变，所以与上文 IP 不同
将 ssh 密钥拷贝到服务器以后，可以直接使用 `ssh ali` 免密登录。

```
daisy@daisy-virtual-machine:~/.ssh$ ssh ali
Welcome to Ubuntu 22.04.4 LTS (GNU/Linux 5.15.0-117-generic x86_64)

* Documentation:  https://help.ubuntu.com
* Management:    https://landscape.canonical.com
* Support:       https://ubuntu.com/pro
```

图 1.7: 实例 7(1) 结果图

8. 实例 7(2)
使用

```
1 python3 -m http.server 8888
```

在虚拟机中启动一个 Web 服务器并通过本机的<http://localhost:9999> 访问虚拟机上的 Web 服务器。

```
daisy@daisy-virtual-machine:~$ python3 -m http.server 8888
Serving HTTP on 0.0.0.0 port 8888 (http://0.0.0.0:8888/) ...
```

图 1.8: 实例 7(2) 结果图

9. 实例 7(3)

使用 `sudo vim /etc/ssh/sshd_config` 编辑 SSH 服务器配置，通过修改 `PasswordAuthentication` 的值来禁用密码验证。通过修改 `PermitRootLogin` 的值来禁用 root 登录。重新尝试登录不需要输入密码

```
daisy@daisy-virtual-machine:~$ ssh root@47.97.244.53
Welcome to Ubuntu 22.04.4 LTS (GNU/Linux 5.15.0-117-generic x86_64)

* Documentation:  https://help.ubuntu.com
* Management:    https://landscape.canonical.com
* Support:       https://ubuntu.com/pro
```

图 1.9: 实例 7(3) 结果图

10. 实例 7(4)

mosh (Mobile Shell) 可以在网络不稳定的情况下保持连接状态。但如果完全断开服务器或虚拟机的网络适配器，**mosh 的连接也将中断，并且不能自动恢复**，直到网络恢复并且客户端能够重新与服务器进行通信。**mosh 虽然可以处理临时的网络中断或 IP 地址改变，但它不能恢复一个完全断开的连接。**

11. 实例 7(5)

-N:-N 告诉 SSH 不要执行任何命令，即只进行端口转发而不打开一个远程 shell。

-f: 让 SSH 在所有需要的端口都转发并且认证完成之后放到后台运行。

后台进行端口转发的命令:`ssh -fN -L 9999:localhost:8888 ali`

1.2.2 终端多路复用

1. 实例 1

详情请看步骤。

2. 实例 2

在新建的窗口2中，使用

```
1 unset TMUX
2 tmux attach -t study
```

共享两个窗口。

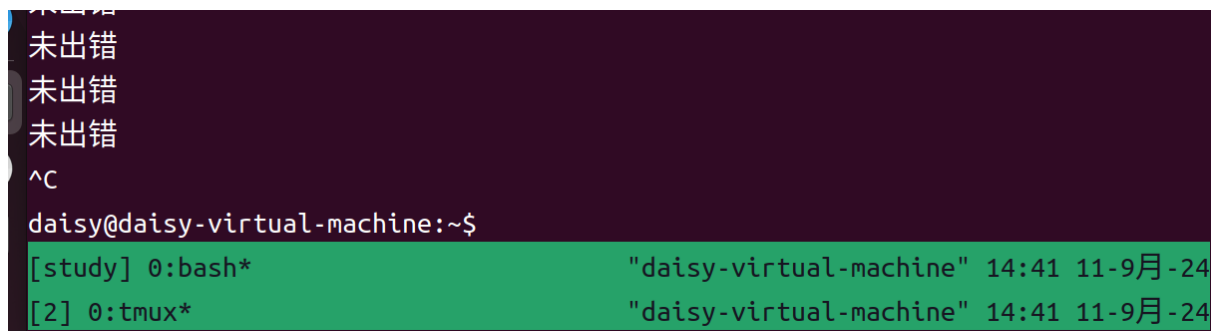


图 1.10: 实例 2 结果图

1.2.3 任务控制

1. 实例 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。

```
1 sleep 10000
2 Ctrl-Z
3 bg
```

使用 `pgrep` 来查找 `pid`，并使用 `kill` 结束进程而不需要手动输入 `pid`。

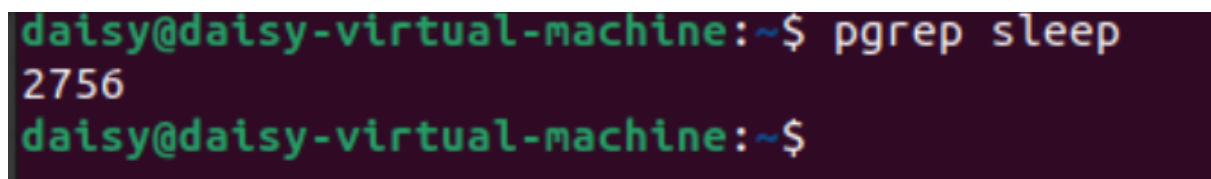


图 1.11: pgrep

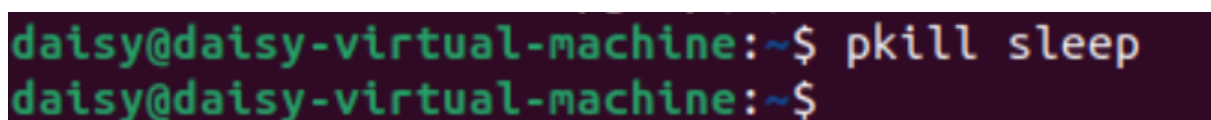


图 1.12: pkill

1.2.4 别名

1. 实例 1

使用

```
1 alias dc=cd
```



```
daisy@daisy-virtual-machine:~$ alias dc=cd
daisy@daisy-virtual-machine:~$ dc ..
daisy@daisy-virtual-machine:/home$
```

图 1.13: 实例 1 结果图

2. 实例 2 使用

```
1 history | awk '{ $1=""; print substr($0,2) }' | sort | uniq -c |
  sort -n | tail -n 10
```

获得使用频率前十的命令。我最常用的是 `git clone`，所以设置了一个别名。

```
daisy@daisy-virtual-machine:~$ history | awk '{ $1=""; print substr($0,2) }' | sort
t | uniq -c | sort -n | tail -n 10
    10 git clone https://github.com/missing-semester-cn/missing-semester-cn.git
    10 git graph
    10 vim cdf.py
    12 git stash
    14 git log
    14 vim hello.txt
    23 python3 gray.py
    29 vim gray.py
    45 cd ..
    55 ls
daisy@daisy-virtual-machine:~$ alias gitc="git clone"
daisy@daisy-virtual-machine:~$
```

图 1.14: 实例 2 结果图

1.2.5 配置文件

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）。建立一种在新设备进行快速安装配置的方法（无需手动操作）。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。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。

```
daisy@daisy-virtual-machine: /
daisy@daisy-virtual-machine:/$ mkdir ~/gits/dotfiles
daisy@daisy-virtual-machine:/$ git init ~/gits/dotfiles
hint: Using 'master' as the name for the initial branch. This default branch
hint: is subject to change. To configure the initial branch name to use in a
hint: of your new repositories, which will suppress this warning, call:
hint: git config --global init.defaultBranch <name>
hint: Names commonly chosen instead of 'master' are 'main', 'trunk' and
hint: 'development'. The just-created branch can be renamed via this command
hint: git branch -m <name>
Initialized empty Git repository in /home/daisy/gits/dotfiles/.git/
daisy@daisy-virtual-machine:/$ ls -a ~/gits/dotfiles
.  ..  .git
daisy@daisy-virtual-machine:/$
```

图 1.15: 结果图

1.3 实验代码及关键步骤

1.3.1 链接到 Github

1. 将创建的公钥添加到 Github 中，即可。

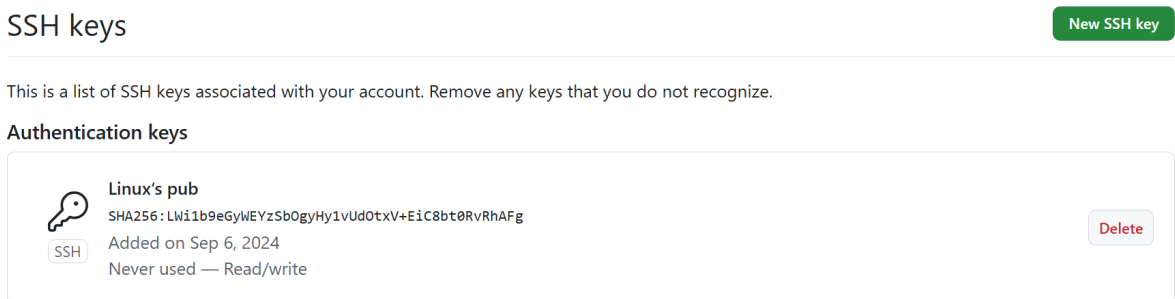


图 1.16: 将公钥添加到 Github

可以使用

```
1 ssh -T git@github.com
```

查看是否连接成功。

1.3.2 创建阿里云服务器并远程链接

1. 去“阿里云服务器”创建一个实例。

<input type="checkbox"/>	实例 ID / 名称	状态	标签	操作系统	监控	可用区	配置	操作
<input type="checkbox"/>	i-bp1f9vj534drdxate2gq iZbp1f9vj534drdxate2gqZ	运行中				华东1（杭州） H	2核(vCPU) 2 ecs.e-c1m1.lai	远程连接 资源变配 停止 ...

图 1.17: 创建一个服务器实例

2. 添加一条允许端口 22 (SSH) 的规则。
3. 提前在客机上远程链接服务器并下载 `ssh`

```
root@iZbp1f9vj534drdxate2gqZ:~# sudo systemctl status ssh
● ssh.service - OpenBSD Secure Shell server
   Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: enabled)
```

图 1.18: 下载 ssh

4. 在虚拟机上使用

```
1 ssh -i ~/.ssh/id_ed25519 root@118.31.189.33
```

链接到服务器。

1.3.3 终端多路复用

1. 创建一个新的窗口

```
1 tmux new -s study
```

```
daisy@daisy-virtual-machine:~$ tmux new -s study
[detached (from session study)]
```

图 1.19: 创建一个新的窗口

2. 在该窗口执行一个循环命令。

```
未出错
未出错
未出错
未出错
未出错
未出错
未出错
未出错
未出错
未出错
未出错
[study] 0:bash* "daisy-virtual-machine" 14:32 11-9月-24
```

图 1.20: 循环命令

3. 关闭终端再打开，使用

```
1 tmux attach -t study
```

程序仍然在运行。

```
daisy@daisy-virtual-machine:~$ tmux attach -t study
```

图 1.21: 打开原来的窗口

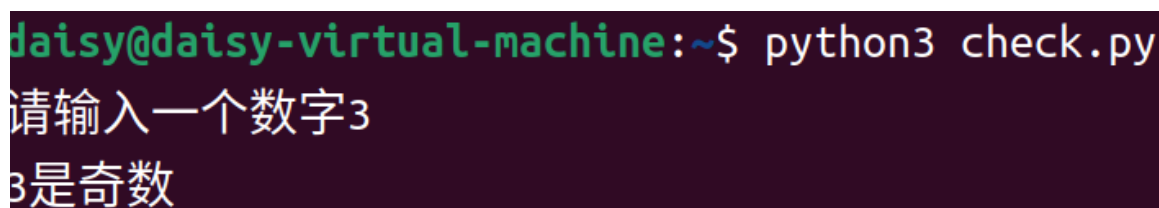
实验部分 2 Python 入门基础

2.1 实验内容

1. 判断一个数是不是奇数。
2. 做一个能够实现双目加减乘除运算的计算器。
3. 找出列表中的最大值。
4. 打印斐波那契数列。
5. 用 python 实现 Hanoi 塔。

2.2 实验结果

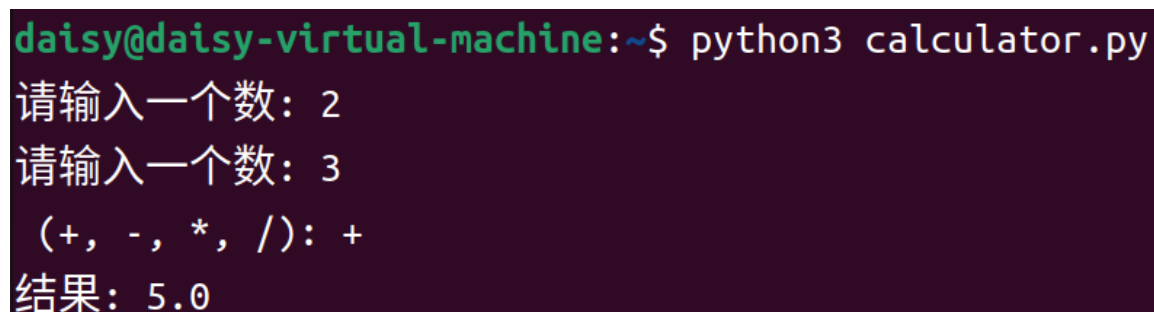
1. 实例 1
判断一个数是不是奇数。



```
daisy@daisy-virtual-machine:~$ python3 check.py
请输入一个数字3
3是奇数
```

图 2.1: 实例 2 结果图

2. 实例 2
做一个能够实现双目加减乘除运算的计算器。



```
daisy@daisy-virtual-machine:~$ python3 calculator.py
请输入一个数: 2
请输入一个数: 3
(+, -, *, /): +
结果: 5.0
```

图 2.2: 实例 2 结果图

3. 实例 3
找出列表中的最大值。

```
daisy@daisy-virtual-machine:~$ python3 max.py
10
5
3
8
20
7
最大的数是： 20
```

图 2.3: 实例 3 结果图

4. 实例 4

打印斐波那契数列。

```
daisy@daisy-virtual-machine:~$ python3 fibo.py
请输入一个数： 11
0 1 1 2 3 5 8 13 21 34 55
```

图 2.4: 实例 4 结果图

5. 实例 5

用 python 实现 Hanoi 塔。

```
daisy@daisy-virtual-machine:~$ python3 hanoi.py
A -> C
A -> B
C -> B
A -> C
B -> A
B -> C
A -> C
```

图 2.5: 实例 5 结果图

2.3 实验代码及关键步骤

1. 实例 1

```
import sys

def main():
    number = input("请输入一个数字")

    if int(number) % 2 == 0:
        print(f"{number}不是奇数")
    else:
        print(f"{number}是奇数")

if __name__ == "__main__":
    main()
```

图 2.6: check.py

2. 实例 2

```
1 import sys
2
3 def calculator():
4     num1 = float(input("请输入一个数: "))
5     num2 = float(input("请输入一个数: "))
6     operation = input(" (+, -, *, /): ")
7
8     if operation == '+':
9         print(f"结果: {num1 + num2}")
10    elif operation == '-':
11        print(f"结果: {num1 - num2}")
12    elif operation == '*':
13        print(f"结果: {num1 * num2}")
14    elif operation == '/':
15        if num2 != 0:
16            print(f"结果: {num1 / num2}")
17        else:
18            print("除0无意义")
19    else:
20        print("无效")
21
22 calculator()
```

3. 实例 3

```
import sys

numbers = [10, 5, 3, 8, 20, 7]

for i in numbers:
    print(i)

max_value = numbers[0]
for num in numbers:
    if num > max_value:
        max_value = num

print("\n最大的数是:", max_value)
```

图 2.7: max.py

4. 实例 4

```
import sys

def fibonacci(n):
    a, b = 0, 1
    for i in range(n):
        print(a, end=" ")
        a, b = b, a + b

num = input("请输入一个数: ")
fibonacci(int(num))
print("\n")
```

图 2.8: fibo.py

5. 实例 5


```
import sys

def hanoi(n, source, target, auxiliary):
    if n == 1:
        print(f"{source} -> {target}")
        return
    hanoi(n - 1, source, auxiliary, target)
    print(f"{source} -> {target}")
    hanoi(n - 1, auxiliary, target, source)

hanoi(3, 'A', 'C', 'B')
```

图 2.9: hanoi.py

实验部分 3 Python 视觉应用

3.1 实验内容

1. 对载入的图像实现灰度变换并将其反相处理，同时打印出原图的等高线图。
2. 将图片直方图均衡化。
3. 对图片进行图片模糊。
4. 对图片进行暗通道去雾。
5. 对图像进行边缘检测。

3.2 实验结果

1. 实例 1

从右至左依次为：等高线图，灰度图，灰度图的反相图

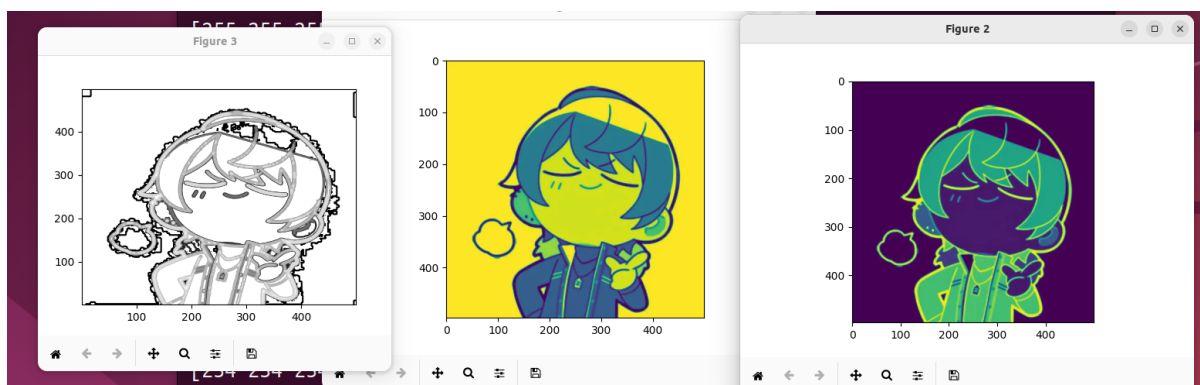


图 3.1: 实例 1 结果图

2. 实例 2

从右至左依次为：灰度图，均衡化后的灰度图。

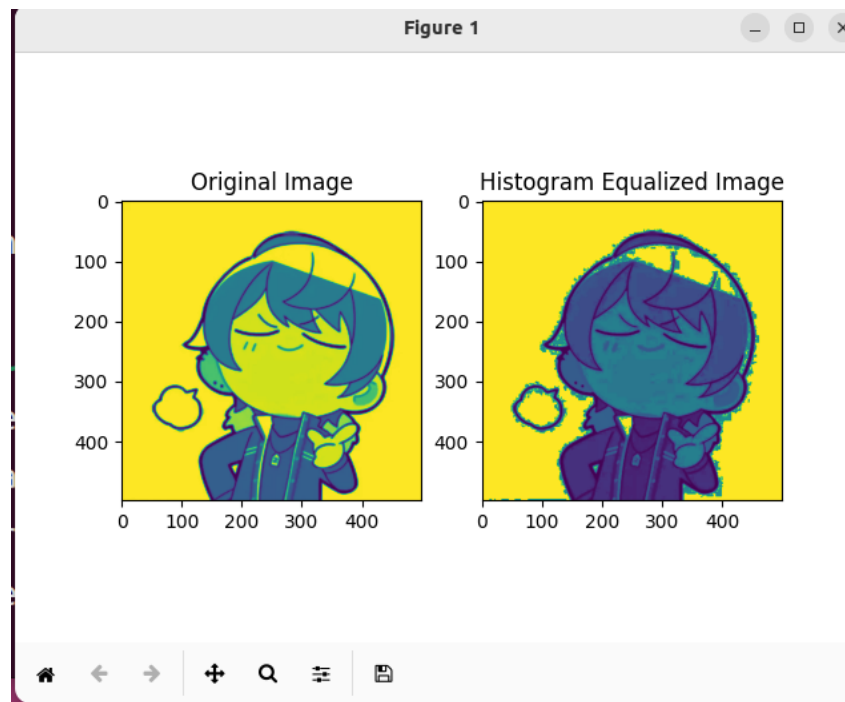


图 3.2: 实例 2 结果图

3. 实例 3

对（灰度）图像和一个高斯核进行卷积操作，随着参数的增大，图片也越来越模糊。

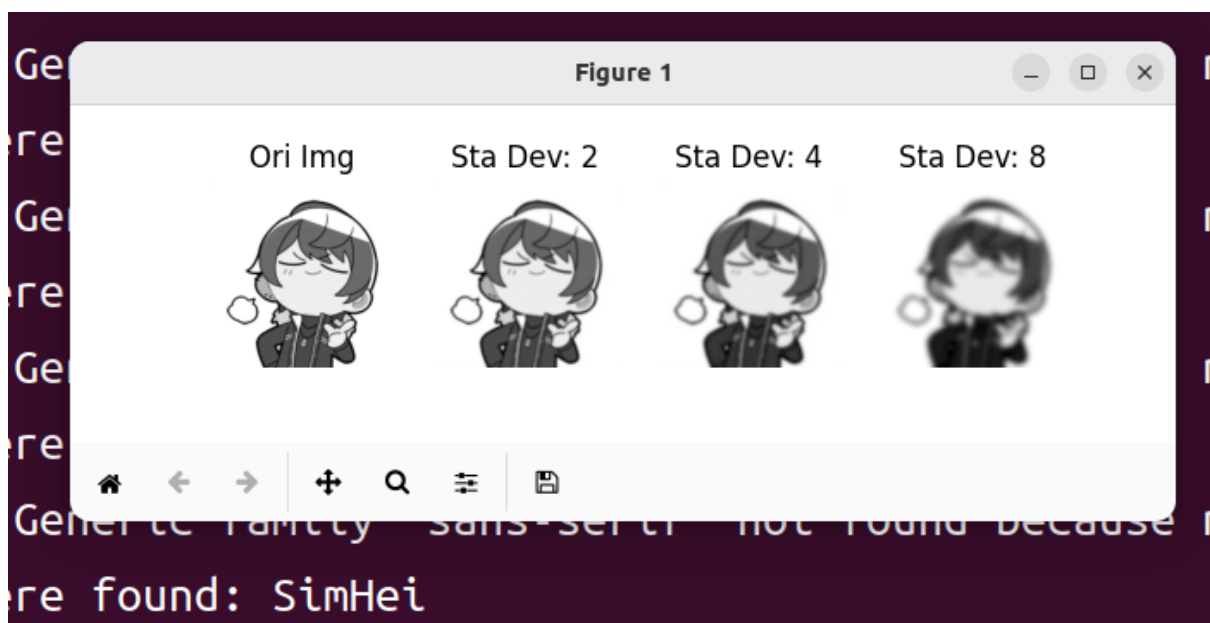


图 3.3: 实例 3 结果图

4. 实例 4

使用最小值滤波和引导滤波来计算大气遮罩图像，并基于这个遮罩来恢复图像的颜色和亮度，来达到去雾效果。使用自然风景图效果会更明显一点



图 3.4: 实例 3 结果图

5. 实例 5

对图进行边缘检测，从左至右，从上至下依次为：灰度图，膨胀处理后图像，腐蚀处理后图像，前两者的绝对差分二值化图像，二值化图像的反色图像。



图 3.5: 实例 4 结果图

3.3 实验代码及关键步骤

1. 实例 1

对载入的图像实现灰度变换并将其反相处理，同时打印出原图的等高线图。

```

1 import matplotlib.pyplot as plt
2 import numpy as np
3 from PIL import Image
4
5 im = np.array(Image.open('QQ.jpg').convert('L'))
6 im2 = 255 - im
7
8 plt.imshow(im)
9
10 print(im)
11
12 plt.figure()
13 plt.imshow(im2)
14
15
16 plt.figure()
17 plt.gray()
18 plt.contour(im2, origin='image')
19
20 plt.show()

```

2. 实例 2

将图片直方图均衡化。

```
1 import matplotlib.pyplot as plt
2 import numpy as np
3 from PIL import Image
4
5 def histeq(im, bins=256):
6     imhist, bins2 = np.histogram(im.flatten(), bins, density=True)
7     cdf = imhist.cumsum()
8     cdf = 255 * cdf / cdf[-1]
9
10    im2 = np.interp(im.flatten(), bins2[:-1], cdf)
11
12    return im2.reshape(im.shape), cdf
13
14 im = np.array(Image.open('QQ.jpg').convert('L'))
15 im2, cdf = histeq(im)
16
17 plt.subplot(1, 2, 1)
18 plt.imshow(im)
19 plt.title('Original Image')
20
21 plt.subplot(1, 2, 2)
22 plt.imshow(im2)
23 plt.title('Histogram Equalized Image')
24
25 plt.show()
```

3. 实例 3

对图片进行图片模糊。

```
1 import matplotlib.pyplot as plt
2 import numpy as np
3 from PIL import Image
4 from scipy.ndimage import gaussian_filter
5
6 plt.rcParams['font.sans-serif'] = ['SimHei']
7
8 im = np.array(Image.open('QQ.jpg').convert('L'))
9
10 plt.figure(figsize=(10, 5))
11 plt.gray()
12 plt.axis('off')
13
14 plt.subplot(1, 4, 1)
15 plt.axis('off')
16 plt.title('Ori Img')
17 plt.imshow(im, cmap='gray')
18
19 for bi, blur in enumerate([2, 4, 8]):
20     im2 = gaussian_filter(im, blur)
21     im2 = np.uint8(im2)
22
23     plt.subplot(1, 4, 2 + bi)
24     plt.axis('off')
25     plt.title(f'Sta Dev: {blur}')
26     plt.imshow(im2, cmap='gray')
27
```

```
28 plt.show()
```

4. 实例 4

对图片进行暗通道去雾。

```
1 import cv2
2 import numpy as np
3
4 def zmMinFilterGray(src, r=7):
5     return cv2.erode(src, np.ones((2*r+1, 2*r+1)))
6
7 def guidedfilter(I, p, r, eps):
8     height, width = I.shape
9     m_I = cv2.boxFilter(I, -1, (r, r))
10    m_p = cv2.boxFilter(p, -1, (r, r))
11    m_Ip = cv2.boxFilter(I*p, -1, (r, r))
12    cov_Ip = m_Ip - m_I * m_p
13    m_II = cv2.boxFilter(I*I, -1, (r, r))
14    var_I = m_II - m_I * m_I
15    a = cov_Ip / (var_I + eps)
16    b = m_p - a * m_I
17    m_a = cv2.boxFilter(a, -1, (r, r))
18    m_b = cv2.boxFilter(b, -1, (r, r))
19    return m_a * I + m_b
20
21 def Defog(m, r, eps, w, maxV1):
22     V1 = np.min(m, 2)
23     Dark_Channel = zmMinFilterGray(V1, 7)
24     cv2.imshow('Dark Channel', Dark_Channel)
25     cv2.waitKey(0)
26     cv2.destroyAllWindows()
27     V1 = guidedfilter(V1, Dark_Channel, r, eps)
28     bins = 2000
29     ht = np.histogram(V1, bins)
30     d = np.cumsum(ht[0]) / float(V1.size)
31     for lmax in range(bins-1, 0, -1):
32         if d[lmax] <= 0.999:
33             break
34     A = np.mean(m, 2)[V1 >= ht[1][lmax]].max()
35     V1 = np.minimum(V1*w, maxV1)
36     return V1, A
37
38 def deHaze(m, r=81, eps=0.001, w=0.95, maxV1=0.80, bGamma=False):
39     Y = np.zeros(m.shape)
40     Mask_img, A = Defog(m, r, eps, w, maxV1)
41     for k in range(3):
42         Y[:, :, k] = (m[:, :, k] - Mask_img) / (1 - Mask_img / A)
43     Y = np.clip(Y, 0, 1)
44     if bGamma:
45         Y = Y**(np.log(0.5) / np.log(Y.mean()))
46     return Y
47
48 if __name__ == '__main__':
49     m = deHaze(cv2.imread('QQ.jpg')/255.0)*255
50     cv2.imwrite('QQ2.jpg', m)
```

5. 实例 5

对图像进行边缘检测。

```

1 import cv2
2 import numpy as np
3
4 image = cv2.imread("QQ.jpg", cv2.IMREAD_GRAYSCALE)
5
6 kernel = cv2.getStructuringElement(cv2.MORPH_RECT, (3, 3))
7
8 dilate_img = cv2.dilate(image, kernel)
9
10 erode_img = cv2.erode(image, kernel)
11
12 absdiff_img = cv2.absdiff(dilate_img, erode_img)
13
14 retval, threshold_img = cv2.threshold(absdiff_img, 40, 255, cv2.
    THRESH_BINARY)
15
16 result = cv2.bitwise_not(threshold_img)
17
18 cv2.imshow("Original Image", image)
19 cv2.imshow("Dilated Image", dilate_img)
20 cv2.imshow("Eroded Image", erode_img)
21 cv2.imshow("Absolute Difference Image", absdiff_img)
22 cv2.imshow("Threshold Image", threshold_img)
23 cv2.imshow("Result Image", result)
24
25 cv2.waitKey(0)
26 cv2.destroyAllWindows()

```

3.4 实验中出现的問題及解決方案

- **問題**
在第一个实验中，从 `pylab` 导入模块时遇到了问题。具体来说，Python 没有找到名为 `pylab` 的模块。
- **解決方案**
查询后安装了 `Matplotlib`，并使用其中的 `pyplot`。

```

daisy@daisy-virtual-machine:~$ pip install matplotlib
Defaulting to user installation because normal site-packages is not writeable
Collecting matplotlib
  Downloading matplotlib-3.7.2-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x
86_64.whl (11.6 MB)
    11.6/11.6 MB 11.7 MB/s eta 0:00:00
Requirement already satisfied: pillow>=6.2.0 in /usr/lib/python3/dist-packages (
from matplotlib) (9.0.1)
Requirement already satisfied: pyparsing<3.1,>=2.3.1 in /usr/lib/python3/dist-pa
ckages (from matplotlib) (2.4.7)

```


实验部分 4 实验心得

在最近的实验中，我重点探索了 SSH 的使用、Python 入门基础代码的编写以及图像处理技术的应用。这些方面的学习和实践不仅加深了我对编程和系统管理的理解，也提高了我的实战能力。

- 在处理远程服务器时，SSH 是一个非常重要的工具。在《Missing Semester》的文档中，我学到了如何使用 SSH 进行安全的远程访问和管理。通过 SSH，我能够安全地连接到远程机器，并执行命令、编辑文件等操作。这不仅方便了我的开发工作，还增强了我对远程管理的理解。具体而言，我掌握了使用 `ssh-keygen` 命令生成 SSH 密钥对，并将公钥部署到远程服务器。这使得每次连接时可以免去输入密码，提升了效率；配置 SSH 客户端：通过配置 `/.ssh/config` 文件，简化了远程连接的过程。例如，可以为不同的服务器设置别名，减少了输入复杂命令的麻烦；使用 SSH 进行端口转发：我还学会了如何使用 SSH 隧道技术，将本地端口转发到远程服务器，进行安全的数据传输。这些技能为我后续的开发工作提供了重要的支持，使得我能够更加高效地管理和调试远程服务器上的应用。
- Python 的基础知识是编程的基石。在这段时间的学习中，我编写了一些基础代码示例，这些示例涵盖了常见的编程概念和技巧，例如：掌握了如何定义和使用不同的数据类型，如整数、浮点数和字符串；了解了如何使用条件语句（如 `if`、`else`）和循环结构（如 `for`、`while`）来控制程序的执行流；函数定义和调用：学习了如何定义函数、传递参数以及返回值，以实现代码的模块化和复用。这些基础知识为编写更复杂的程序打下了坚实的基础，并帮助我理解了代码的结构和逻辑。
- 在图像处理方面，我进行了多个实验，包括图像的灰度化、二值化、形态学操作以及图像分割。通过使用 OpenCV 和 numpy 库，我能够实现以下功能：1. 图像灰度化：将彩色图像转换为灰度图像，以简化后续的处理步骤；2. 二值化处理：通过设定阈值将灰度图像转换为二值图像，突出图像中的重要特征；3. 形态学操作：使用膨胀和腐蚀操作来增强图像的边缘和细节；4. 图像分割：基于像素值统计进行图像分割，从而提取图像中的特定区域。这些技术使我能够有效地处理和分析图像数据，并为进一步的计算机视觉任务奠定了基础。例如，在处理风景图像时，我观察到了形态学操作对细节的显著影响，进一步验证了这些技术的实用性。

附录 1

我的 Github 仓库网址

我的 Github 仓库 [\[点击即可\]](#)