

# 报告三：Python 和命令行环境

23020007011 崔涛

2024 年 9 月 10 日

## 1 实验目的

本次课程主要讲授了命令行环境以及 Python 编程，掌握基本的操作，方便以后的工作需要。

## 2 介绍

### 2.1 命令行环境和 Python 编程

命令行环境提供了一种轻量级、灵活且直接的方式来执行和管理 Python 脚本，而 Python 编程因其易用性、强大的库支持和跨平台能力而受到广泛欢迎。命令行环境资源占用小，直接性和灵活性强，适合自动化和批处理并且易于远程管理。Python 语法简单易懂，强大的标准库和第三方库，跨平台性强同时具有面向对象和多范式支持的特点。

## 3 练习内容

### 3.1 命令行环境学习例子 10 个

1. `jobs` 命令会列出当前终端会话中尚未完成的全部任务。可以使用 `pid` 引用这些任务（也可以用 `pgrep` 找出 `pid`）。也可以使用百分号 + 任务编号（`jobs` 会打印任务编号）来选取该任务。如果要选择最近的一个任务，可以使用 `$!` 这一特殊参数。

2. 执行 `history | awk '{ $1="" ; print substr($0,2) }' | sort | uniq -c | sort -n | tail -n 10` 来获取最常用的十条命令

3. 终端多路复用 `tmux` 会话操作。

- `tmux` 开始一个新的会话
- `tmux new -s NAME` 以指定名称开始一个新的会话

```
cuitao@ubuntu:~$ history | awk '{ $1="" ; print substr($0,2) }' | sort | uniq -c | sort -n | tail -n 10
  8 cd Desktop
 10 cd ~
 11 gcc -g -Wall '/home/cuitao/Desktop/Untitled Folder 2/2.c' -o dianzhen
 11 ./hellow
 12 cd
 12 su - root
 15 cd ./Desktop
 15 pwd
 17 ./dianzhen
 30 ls
```

图 1: 常用的十条指令

```

cuitao@ubuntu:~$ alias ll="ls -lh"
cuitao@ubuntu:~$ ll
总用量 144K
-rwxrwxr-x 1 cuitao cuitao 9.5K May  9 16:25 a.out
-rw-rw-r-- 1 cuitao cuitao 212 Aug 30 10:08 buggy.sh
drwxrwxr-x 2 cuitao cuitao 4.0K Nov 24 2023 build-untitled-Desktop-Debu
drwxrwxr-x 2 cuitao cuitao 4.0K May  7 14:49 cui
drwxrwxr-x 2 cuitao cuitao 4.0K May  7 15:19 cuitao1
-rwxrwxr-x 1 cuitao cuitao 9.5K May  9 16:26 cuitao.asd
-rw-rw-r-- 1 cuitao cuitao 229 Aug 30 10:09 debug_for.sh
drwxr-xr-x 2 cuitao cuitao 4.0K Sep 10 22:19 Desktop
drwxr-xr-x 2 cuitao cuitao 4.0K Oct  8 2023 Documents
drwxr-xr-x 2 cuitao cuitao 4.0K May 21 19:05 Downloads
-rw-r--r-- 1 cuitao cuitao 8.8K Oct  8 2023 examples.desktop
-rw-rw-r-- 1 cuitao cuitao  7 May  7 14:46 hah.txt
-rw-rw-r-- 1 cuitao cuitao 54 Aug 30 10:03 narco.sh

```

图 2: Enter Caption

- `tmux ls` 列出当前所有会话
- 在 `tmux` 中输入 `<C-b> d`，将当前会话分离
- `tmux a` 重新连接最后一个会话。您也可以通过 `-t` 来指定具体的会话

#### 4. 终端多路复用 `tmux` 会话操作

- `<C-b> c` 创建一个新的窗口，使用 `<C-d>` 关闭
- `<C-b> N` 跳转到第  $N$  个窗口，注意每个窗口都是有编号的
- `<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

#### 5. 终端多路复用 `tmux` 面板操作

- `<C-b> "` 水平分割
- `<C-b> %` 垂直分割
- `<C-b> <方向>` 切换到指定方向的面板，`<方向>` 指的是键盘上的方向键
- `<C-b> z` 切换当前面板的缩放
- `<C-b> [` 开始往回滚动屏幕。您可以按下空格键来开始选择，回车键复制选中的部分
- `<C-b> <空格>` 在不同的面板排布间切换

6. 别名的设置  
别名设置的格式为：`alias alias_name="command_to_alias arg1 arg2"` 实际应用  
例如将 `ll` 设置为 `ls -lh`，这样可以简化指令，便于记忆。`alias ll="ls -lh"`

#### 7. 远程设备的连接

- 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

- 在 `.ssh/config` 加入以下内容  
`Host vm`  
`User username_goes_here`  
`HostName ip_goes_here`  
`IdentityFile ~/.ssh/id_ed25519`  
`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:` 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, 然后重新尝试即可连接到虚拟机。

## 8. tmux 常用指令

- `tmux ls` 会显示出所有正在运行的对话
- `tmux attach -t 0` 连接到特定的对话。
- `tmux rename-session -t 0 database` 重命名当前会话。

## 9. 常见目录说明

- `/bin`: 存放**二进制可执行文件** (`ls`, `cat`, `mkdir` 等), **常用命令一般都在这里**;
- `/sbin`: 存放二进制可执行文件, 只有 root 才能访问。这里存放的是系统管理员使用的系统级别的管理命令和程序。如 `ifconfig` 等;
- `/etc`: 存放**系统管理和配置文件**;
- `/root`: 超级用户 (系统管理员) 的主目录;
- `/home`: 存放所有用户文件的根目录, 是用户主目录的基点, 比如用户 `user` 的主目录就是 `/home/user`, 可以用 `~user` 表示;
- `/dev`: 用于存放**设备文件**;
- `/usr`: 用于存放**系统应用程序**;
- `/lib` 和 `/lib64`: 存放着**和系统运行相关的库文件**;

10. `top` 指令可以查看系统进程和资源占用情况。

```
top - 22:49:38 up 26 min, 2 users, load average: 0.00, 0.00, 0.00
tasks: 227 total, 1 running, 160 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.5 us, 0.7 sy, 0.0 ni, 98.8 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
KiB Mem : 4015908 total, 2758660 free, 656568 used, 600680 buff/cache
KiB Swap: 998396 total, 998396 free, 0 used, 3057196 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1821	cuictao	20	0	1285556	118680	81976	S	1.3	3.0	0:07.22	compliz
958	root	20	0	525688	92732	48912	S	0.3	2.3	0:04.14	Xorg
1853	cuictao	20	0	533016	35732	30396	S	0.3	0.9	0:01.81	vmtoolsd
2245	cuictao	20	0	609984	43636	35088	S	0.3	1.1	0:00.70	gnome-ter+
3134	cuictao	20	0	43640	4212	3524	R	0.3	0.1	0:00.37	top
1	root	20	0	119864	6036	4088	S	0.0	0.2	0:02.10	systemd
2	root	20	0	0	0	0	S	0.0	0.0	0:00.02	kthreadd
3	root	20	0	0	0	0	I	0.0	0.0	0:00.08	kworker/0+
4	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/0+
6	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	mm_percpu+
7	root	20	0	0	0	0	S	0.0	0.0	0:00.02	ksoftirqd+
8	root	20	0	0	0	0	I	0.0	0.0	0:00.21	rcu_sched
9	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_bh
10	root	rt	0	0	0	0	S	0.0	0.0	0:00.00	migration+
11	root	rt	0	0	0	0	S	0.0	0.0	0:00.00	watchdog/0
12	root	20	0	0	0	0	S	0.0	0.0	0:00.00	cpuhp/0

图 3: top 指令查看系统情况

## 3.2 Python 学习例子 10 个

1.python 实现字符串编码解码操作。编码是将字符串转换为二进制数据 bytes；而解码则是将 bytes 类型的数据转换成字符串类型

```
s = '系统开发工具基础'

# 编码
print(s.encode(encoding='GBK')) # 一个中文字符, 两个字节
print(s.encode(encoding='UTF-8')) # 一个中文字符, 三个字节

# 解码
byte = b'\xb4\xf3\xba\xd3\xcf\xf2\xb6\xab\xcl\xf7'

print(byte.decode(encoding='GBK'))

byte = b'\xe5\xa4\xa7\xe6\xb2\xb3\xe5\x90\x91\xe4\xb8\x9c\xe6\xb5\x81'
print(byte.decode(encoding='UTF-8'))
```

图 4: Python 编码解码

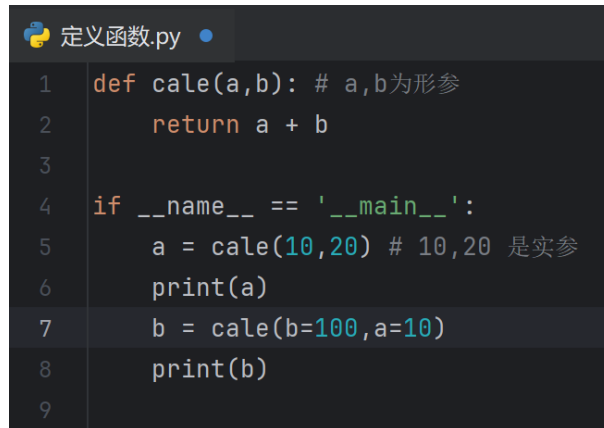
2.Python 实现文件读写

```
with open('example.txt', 'r') as file:
    content = file.read() # 读取整个文件内容
    print(content)

with open('example.txt', 'w') as file:
    file.write('Hello, World!') # 写入内容到文件
```

图 5: 文件读写

3.Python 实现自定义函数，从而实现有效的代码复用。



```

定义函数.py
1 def cale(a,b): # a,b为形参
2     return a + b
3
4 if __name__ == '__main__':
5     a = cale(10,20) # 10,20 是实参
6     print(a)
7     b = cale(b=100,a=10)
8     print(b)
9

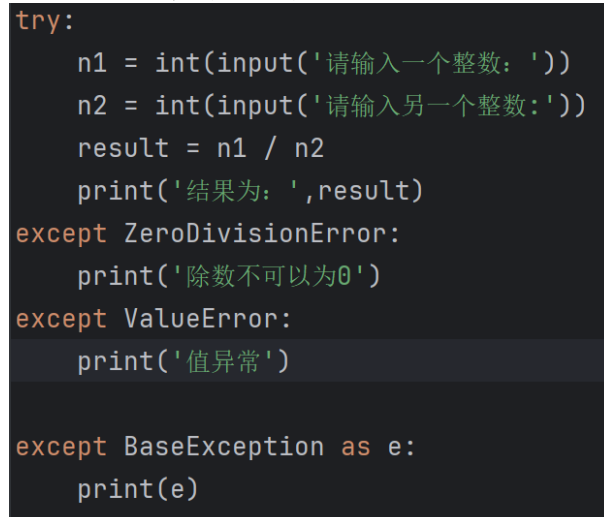
```

图 6: Python 函数的定义

#### 4. Python 常见抛出的异常类型

- ZeroDivisionError # 除或取模零的类型
- IndexError # 序列中没有此索引
- KeyError # 映射中没有这个键
- NameError # 未声明/初始化对象 (没有属性)
- SyntaxError # Python 语法错误
- ValueError # 传入无效的参数

使用 try 和 except 来抛出和捕获异常，保证程序正确运行。



```

try:
    n1 = int(input('请输入一个整数: '))
    n2 = int(input('请输入另一个整数: '))
    result = n1 / n2
    print('结果为: ', result)
except ZeroDivisionError:
    print('除数不可以为0')
except ValueError:
    print('值异常')

except BaseException as e:
    print(e)

```

图 7: Python 异常处理

#### 5. Python 类的声明定义

6. 通过 pip install 模块名，可以安装第三方模块，避免重复造轮子。

7. 切换颜色向输出台打印文字: `print('\033[0:35m\t\tPython 切换颜色打印\033[m')`

8. python 通过循环来实现冒泡排序

```
class Student:
    address = '青岛'
    def __init__(self,name,age):
        self.name = name
        self.age = age
    def info(self):
        print('name:',self.name,'age:',self.age)

stu = Student('崔涛','01')
print(stu.name)
```

图 8: 类的声明定义

```
def BubbleSort(lst):
    n=len(lst)
    if n<=1:
        return lst
    for i in range (0,n):
        for j in range(0,n-i-1):
            if lst[j]>lst[j+1]:
                (lst[j],lst[j+1])=(lst[j+1],lst[j])
        return lst
x=input("请输入待排序数列: \n")
y=x.split()
arr=[]
for i in y:
    arr.append(int(i))
arr=BubbleSort(arr)
#print(arr)
print("数列按序排列如下: ")
for i in arr:
    print(i,end=' ')
```

图 9: 冒泡排序

## 9. 字典数据结构

字典的一般形式为  $d = \{\text{key1} : \text{value1}, \text{key2} : \text{value2} \}$

值可以取任何数据类型，但键必须是不可变的，如字符串，数字或元组。

## 10. 求解斐波那契数列第 $n$ 项

```
def fib(n):
    if n <= 0:
        return 0
    elif n == 1:
        return 1
    else:
        return fib(n-1) + fib(n-2)
```

图 10: 求解斐波那契数列

## 4 收获感悟

通过学习命令行环境，我学会了编写简单的脚本，来实现一些基本的文件管理、数据处理等，极大地提高了日常效率。同时我还学会了如何远程连接虚拟机，这样方便远程协作。通过学习 tmux，我学会了多视窗协作，大大方便了以后的工作。学习 Python 最喜欢的就是它的库。我发现无论是爬虫还是机器学习，Python 都有相应的第三方库来简化，大大降低了难度，更加容易上手，但是我又发现 Python 程序的运行时间普遍是要长于之前学习过的 c 语言，我想这应该是 Python 的一个缺陷。通过 Python 和命令行环境这俩的综合学习，我明白了如何通过命令行运行 Python 脚本，如何利用 Python 来处理命令行中的文本和数据，大大提高了效率。

github 路径您可以在这里查看项目的源代码:

<https://github.com/cuitao223/homework3>