**day01笔记:**

自我介绍:

魏明择

email：　 weimz@tedu.cn

1. 毕业 (水利水电动力工程)

Java / C语言精通 / C++语言 / Python

Python 人工智能课程介绍:

1. 　Python 软件开发基础

Linux操作系统, Python的语法，字符串，各种语句,模块，文件操作,异常处理，面向对象

2048游戏，学生信息管理

2. Python高级编程

数据库，网络通信，进程，线程，错误调试.

网络通信软件

3. Python 全栈式web开发

前端:

Html, CSS, Javascript, JQuery ...

后端(服务器端):

Django框架开发，Tornado web框架

4. 爬虫, 大数据分析，人工智能

numpy Scripy框架,pandas

监督学习，非监督学习，语音语别，人脸识别

互联网 + ABC

A ---> AI(Artificial Intelligence)人工智能

B ---> Big Data 大数据

C ---> Cloud Computing 云计算

操作系统：

Windows

(win32,win95,win98, win me,win2000, win xp, win vista, win7, win8, win10)

UNIX

IOS, MAC OS, AIX(IBM), Solaris(SUN,Oracle)

Linux (Linux Torvas)

Android(安卓)

Ubuntu(教学环境)

Red hat

cent OS

vmware 使用：

1. 启动(双击图标)

2. 选择虚拟机运行

Ubuntu的密码是: tarena

快捷键：

ctrl + alt 释放鼠标到主机

ctrl + alt + enter 全屏/退出全屏切换

Linux 有使用：

通过终端 Terminal 操作计算机

启动终端:

Linux命令：

命令的格式：

命令名 [选项] [参数]

注：

1. 命令名必须存在

2. 命令名区分大小写

3. [] 内部的选项和参数可以没有

date 命令：

作用：

用于显示系统日期

格式：

date +%D 月/日/年 格式

date +%T 时:分:秒 格式

cal 命令

作用：

显示日历:

$ cal # 显示当月的日历

$ cal 2018 # 显示2018年的全部日历

$ cal 8 2020 # 显示2020年8月的日历

目录树：

根(root) /

路径:

路径是表示一个文件或文件夹的字符串

例:

/home/tarena

/home/tarena/2.txt

Linux下的路径：

分为两种：

绝对路径:

相对路径

绝对路径：

以'/' 字符开头的路径为绝对路径

相对路径:

不以'/' 开头的路径

开始符号:

文件/文件夹名称

. 当前文件夹(目录)

.. 上一级文件夹(目录)

~ 用户主目录

pwd 命令：

作用：

显示当前的工作路径在哪儿

例：

$ pwd

/home/tarena

ls 命令：

作用：

显示指定文件夹或文件信息

常用选项:

-l 列表显示详细信息

-a 显示全部的文件/文件夹

cd 命令：

作用：

　　进入到一个目录(切换工作路径)

格式:

cd 文件夹名(目录名)

示例：

　　cd # 回家(回到用户主目录)

cd / # 到根目录下

cd /etc/init.d # 到 /etc/init.d下

cd .. # 到上一级目录

cd ~ # 回家,等同于 cd

cd - 回到进到此目录之前的文件夹

mkdir 命令：

创建文件（新建文件夹)

格式：

mkdir 文件夹名

rmdir 命令：

　　删除文件夹(文件夹必须为空时才能删除)

格式:

rmdir 文件夹名(可以是文件夹路径)

touch 命令：

　　创建一个或多个空文件

格式:

touch 文件名1　文件名2 ...

rm 命令:

删除一个文件或多个文件

格式:

rm 文件名1 文件名2 ...

练习：

　　创建一个目录结构：

　　　　　　　　myfolder

|

+----+---+---+

| | |

src doc lib

(目录)(目录)　　　(目录)

|

+-------+---------+

| | |

games office README.txt

(目录) 　(目录) (文件)

tree 命令

**day02笔记:**

三个主流操作系统：

Windows（微内核)

UNIX (微内核)

Linux(单体内核)

终端的使用

　　打开终端：

　　　　ctrl + alt + t

Linux　命令：

　　date 命令

　　　　　显示日期时间

　　cal 命令：

　　　　 显示日历

文件系统相关的命令：

pwd 显示当前工作目录(路径)

路径：

　　绝对路径

　　　　/

C:\

D:\

　　相对路径

　　　　文件名

.

..

~

ls 命令:

查看文件或文件夹

-l 显示详细信息

-a 显示所有文件

cd 命令

　　切换工作目录

**day02 笔记**

mkdir 命令

　　　新建文件夹

常用选项:

-p 如果中间文件夹不存在，则逐级创建所有文件夹

示例：

　　mkdir -p a/bb/ccc/ddd

tree

.

└── a

└── bb

└── ccc

└── dddd

rmdir 命令

　　 删除文件夹

常用选项:

-p

示例：

　　rmdir -p a/bb/ccc/

touch 命令:

作用：

　　1. 如果文件不存在，则创建一个空文件

　　2. 如果文件存在，则用系统时间更新它的修改时间

格式：

　　　touch 文件名1　文件名2 ...

rm命令:

作用：

删除文件或文件夹

格式:

rm　[选项] 文件或文件夹

常用选项：

-r 递归删除文件夹内的文件和文件夹

-i 删除前给出提示(y代表yes,n代表no)

-f 强制删除， 不给任何提示

练习：

1. 在用户主目录下 创建目录: 我的计算机

2. 在"我的计算机"下创建三个目录: 软件，游戏，学习

3. 在"学习"目录中创建三个文件： python.txt, AI.txt ai.txt

4. 删除学习目录中的AI.txt文件

5. 删除"游戏"这个文件夹

--help帮助选项:

用于显示命令的帮助信息

格式：

命令名 --help

示例：

ls --help

man 命令：

作用：

查看命令所对应的帮助文档（手册)

格式：

man 命令名

示例：

man ls

操作键：

翻页键: j(下), k(上)

退出键: q

tab键

在输入命令时能自动补全命令名 或 路径

history 命令

作用：

把以前键入的命令都列出来

调用以前执行的命令：

!编号 执行对应的命令

注：

在终端提示符($或#)下，用上下键可以翻出历史命令，然后回车执行

通配符

\* 代表0个,1个或多个任意字符

? 代表1个任意字符

示例：

touch a ab ac abc aabb bc cd

ls a\*b # 列出所有以a开头，以b结尾的文件

ls c\*

ls \*.txt # 列出所有以.txt结尾的文件

练习：

1. 创建文件夹myfile

2. 在myfile下创建以下文件：

a.mp3 ab.txt ac.mp3 abc.txt aabb.mp3

ab.txt cd.mp4

3. 列出所有的.mp3结尾的文件

4. 删除.之前的文件名只有一个字符的文件(a.mp3)

5. 列出文件名中含有c这个字符的文件

笔记的上传网址：

code.tarena.com.cn

用户名: tarenacode

密码是：code\_2013

路径: AID1802/

文本编辑器

vi/vim

sublime text 3

vi/vim 文本编辑器

作用：

　　编辑文本文件信息，主要用来写程序的源代码

　　启动命令：

　　 vi

　　　　或

vi 文件名

　　退出:

:q<回车>

vim 两种常用的模式：

a/i/o/O

　　　命令模式 --------> 编辑模式

<--------

ESC键

vim 常用的命令：

　　:q 退出

:q! 不保存,强制退出

:w 保存

:w 文件名 保存到指定的文件(另存...)

:wq 保存并退出

vim 的复制命令：

　　yy 复制一行或多行

p 粘帖

y$ 复制到一行行尾

y^ 复制当前光标处到行首

dd 删除当标所在的行

撤销(Undo) 和重做(Redo) 操作：

　　u 　　　　　　取消最近一次的操作,并恢复操作结果

ctrl + r 对使用u命令撤销的操作进行恢复

练习:

在用户主目录下创建一个文件 mypasswd.txt

将/etc/passwd文件的前10行抄写到 mypasswd.txt 中

(要求: 用vi 编辑器)

sublime text 3　文本编辑器

启动命令:

subl<回车>

subl 文件名<回车>

sublime text 的快捷操作:

1. 打开和隐藏side bar　的快捷键

ctrl + k + b

注：只有打开文件夹时才能打开侧边栏

菜单操作：

　　　view-> side bar --> Hide/Show Side Bar

sublime 创建多个光标:

ctrl + 鼠标点击 　　创建多个光标

按ESC键或单击鼠标左键可以取消多个光标

sublime 多选相同的块

ctrl + d

其它快捷键:

ctrl + c 复制光标所在的行(未选中任何内容时)

ctrl + v 在光标的上一行粘贴复制的行

ctrl + + 字体放大

ctrl + - 字体缩小

ctrl + s 保存文件(重要)

文本文件查看相关的命令:

cat 命令：

作用：

　　将文件*内容作*为标准输出显示

格式：

　　cat 文件名1 文件名2 ...

示例：

　　cat /etc/passwd

more 命令：

　　作用：

　　　　分页显示文本文件的内容

示例：

　　　more /etc/passwd

基本操作：

　　q 退出

空格 下翻一页

less 命令：

作用：

　　 分页显示文本文件内容（可以上下回滚,翻页)

格式：

　　less 文件名

less 基础操作:

q 退出

j 下翻

k 上翻

空格 下翻一页

head 命令:

查看文件头几行内容

tail 命令:

查看文件的末尾几行内容

格式:

head -n 文件名

tail -n 文件名

注：　n 为整数数字(默认为10)

cp 命令

作用：

　　复制文件或文件夹

格式:

cp [选项] 源文件名　目标文件或文件夹名

cp -a 源文件夹名 目标文件夹

示例:

cp mypasswd.txt 桌面

　　　　cp mypasswd.txt 桌面/a.txt

　　　　cp -a game 桌面

mv 命令

　　作用：

　　　　文件搬移或更名

格式:

mv 文件1 目标文件2

mv 源文件夹 目标文件夹

练习：

　　１. 创建一个文件夹: 我的电脑

　　2. 在"我的电脑"内创建两个文件夹:c , d

　　3. 在"我的电脑/d"里创建如下几个文件夹:

　　 音乐，电影, 笔记, python程序

　　４. 在笔记文件夹内用文本编辑器创建两个文件:

　　 目录.txt,和　日记.txt

　　5. 在目录.txt中写入:

　　 day01 linux 命令

day02 python

day03 web前端

　　6. 在 日记.txt 中写:

2018年 3月12日

7. 把之前的创始的所有.mp3结尾的文件放入"音乐"文件

8. 把"我的电脑" 文件夹复制到桌面上

which 命令:

　　作用：

显示命令所在的可执行文件的完整路径

示例：

　　which ls

/usr/bin/ls

whereis 命令:

　　作用：

　　　　显示一个二进制文件，源码或man手册的位置

示例：

　　 whereis ls

day03笔记:

mkdir

-p

mkdir -p a/b/c/d

rmdir

-p

touch

两个作用：

　　１．　创建文件

　　２．　如果文件存在，则修改时间

rm

　　-r 递归删除

-i 删除前给提示

--help 选项

man 命令:

查看手册

格式：　man 命令名

history 命令

通配符

　　\*

　　?

vi / vim 文本编辑器

emacs

gedit

sublime text 3 文本编辑器

文本文件查看命令:

tail

head

cat

more

less

cp 命令

复制文件

-a 全部复制(包含文件夹)

mv 命令：

作用：

1. 移动文件

2. 改文件名

which 命令

查看一个命令所对应的文件

whereis 命令

**day03笔记：**

file 命令

查看文件的类型

find 命令：

作用：

根据文件名查找指定的文件

格式:

find 路径 -name "文件名"

问题：

昨天作业中"日记.txt" 在哪儿个文件夹下？

示例：

find / -name "日记.txt"

find /home -name "目录.txt"

ctrl + c 可以终止当前程序的执行

grep 命令

作用：

根据文件内容查找相应的文件

格式：

grep "内容" [选项] 文件名或路径

常用选项:

-n 显示行号

-r 递归搜索文件夹内的文件

示例：

grep "2018年" -nr /home

练习：

1. 查找 group 这个文件在哪儿个文件夹下

命令:

find /etc -name "group"

2. 查找/etc下哪儿个文件里含有 "tedu" 这个字符串

命令：

grep "tedu" -nr /

wc 命令(word counter)

作用：

统计文件中的字节数，字数，行数

格式:

wc [选项] 文件名

常用选项:

-l 只显示行数

-c 只显示字节数

示例：

wc day03.txt

问题：

昨天的练习：我的电脑这个文件夹能否复制给同学一份

gzip 命令:

作用：

用zip压缩算法对文件进行压缩,生成压缩后的文件(.gz 结尾)

格式：

gzip 文件名

示例：

cp /etc/passwd mypasswd.txt

gzip mypasswd.txt

gunzip命令

作用：

对用zip压缩算法压缩的.gz格式的数据进行解压缩

格式：

gunzip 文件名

示例：

gunzip mypasswd.txt.gz

打包和解包的命令

tar 命令

作用：

对文件或文件夹进行打包和解包的操作

格式：

tar [选项] 文件名或路径

常用选项:

-c 创建包

-x 解包

-v 显示操作文件的细节

-f 文件名 操作的文件名

-z 用gzip/gunzip 对包进行压缩和解压缩

常用的打包命令：

tar -czvf day02.tar.gz day02/

.....解包命令

tar -xzvf day02.tar.gz

练习：

1. 把昨天的作业"我的电脑"文件夹打包为mycompute.tar.gz 复制到桌面

2. 用解包命令，查看解压缩后的数据与原数据是否相同

用户权限管理命令:

sudo 命令:

作用：

用超级用户root权限执行这些命令

格式：

sudo 命令 [选项] [参数]

常用选项:

-i 选项, 切换到root 用户

示例：

$ sudo mkdir /myfolder\_haha

$ sudo -i # 切换到root用户

超级用户 root

exit 命令:

作用：

退出用户登陆

如:

# exit # 退出用户登陆

UNIX/Linux 组管理及命令:

组相关的配置文件:

/etc/group

groupadd 命令

作用:

创建一个组

groupdel 命令

作用：

删除一个组

UNIX/Linux 用户管理命令:

用户信息:

1. 用户名称

2. 用户ID

3. 用户的主组

4. 用户主目录

5. 用户密码

6. 用户登陆时的 shell 解释器

用户相关的配置文件:

/etc/passwd

/etc/shadow

useradd 命令:

作用：

添加一个用户

常用选项:

-g 组名或GID 指定主组

-p 密码 在创建时指定密码

-m 创建用户主目录

-d 目录 指定其它的用户主目录

-s shell程序路径 指定新用户的登陆shell

示例：

$ sudo useradd weimz -m -s /bin/bash

passwd 命令

作用：

修改用户密码

格式:

passwd [选项] 用户名

示例：

$ sudo passwd weimz #修改weimz的密码

$ passwd # 修改当前用户自己的密码

userdel 命令：

作用：

删除一个已存在用户

格式：

userdel [选项] 用户名

常用选项:

-r 删除用户的主目录和相关文件

示例：

$ sudo userdel -r weimz

usermod 命令：

作用：

修改用户的相关信息

格式：

usermod [选项] 用户名

常用选项见:

usermod --help

su 命令:

作用：

切换到指定的用户

格式:

su [用户名]

注：如果不指定参数，默认切换到root用户

练习：

1. 在你的Ubuntu 虚拟机内创建你名字的用户

$ sudo useradd weimz -m -s /bin/bash

2. 用passwd给你名字的用户设置初始密码是123456

3. 用su 命令切换到你名字的那个用户, 创建一个notes.txt文件，并用vi 写入你的姓名

4. 退出以你名字命名的用户,回到tarena 用户

5. 查看刚才你写过的文件

6. 偿试修改你刚才写过的文件

7. 查看/etc/shadow 下你名字帐户对应的密码值

8. 修改你名字帐户的密码,再看看/etc/shadow里有什么变化

9. 删除你名字的帐户（可以考虚不做此步)

文件权限管理及命令：

查看文件权限:

ls -l

UNIX/Linux文件类型

- 普通文件

d 文件夹

c/b 设备文件charactor / block

p 管道文件

l 链接文件

s 套接字文件

文件的权限类型：

r 读取权限

w 写权限

x 执行权限

文件权限的分组:

用户权限(user) 组权限(group) 其它(other)

rwxrwxrwx

---------

属主

每个文件必须属于某一个用户，所有者称属主

属组

每个文件必须属于某一个组，这个组称为属组

以上两个都不是的用户称为其它用户

chmod 命令

作用：

修改文件的权限

格式：

chmod 权限 文件名

权限:

u 用户(属主)

g 同组用户（属组）

o 其它用户

a 所有用户

+ 添加权限

- 去除权限

777 最高权限

000 最低权限

示例：

chmod o-r a.txt # 其它人去掉读权限

chmod g-rw a.txt # 属组用户云掉读写权限

chown 命令:

修改文件的属主(需要管理员权限)

格式：

　　chown [owner][:group] [选项] 文件

常用选项:

-R 文件名或文件夹名 对所有文件执行递归修改操作

示例：

　　$ sudo chown weimz:weimz a.txt

$ sudo chown tarena a.txt

练习：

　　1. 用tarena用户创建一个文件: myprog.sh

2. 编辑 myprog.sh 添入以下内容:

#! /bin/bash

ls /

cd ~

mkdir "xxx来过"

3. 修改此文件的权限,让tarena用户有可执行此文件的权限

4. 用tarena用户执行此文件:

$ ./myprog.sh

**day04笔记:**

grep 命令

find 命令

file 命令

wc 命令

tar 命令

z 选项

gzip 命令

gunzip 命令

　　打包:

tar -czvf 包名.tar.gz 文件夹或文件

解包:

tar -xzvf 包名.tar.gz

groupadd 添加组(创建组)

groupdel 删除组

sudo 命令：

　　用root权限来执行命令

-i 选项 切换到root用户（仅用于ubuntu linux)

用户管理命令：

　　useradd

userdel

usermod

passwd 命令

su 命令

用户信息:

用户名,ID,用户主目录(家目录), 主组(GID), 启动的shell程序，密码

/etc/passwd

/etc/shadow

文件权限

# ls -l

-rwx rwx rwx 属主 属组　文件大小 修改日期

chmod 命令，修改文件权限

chown 命令：

　　　　　　修改属主／属组

**day04笔记：**

alias 命令：

　　作用：

　　　　给命令取别名

格式：

　　alias 新命令名='旧命令名 [选项...]'

示例：

　　alias delete='rm -i'

Unix/Linux 管道

　　运算符

|

作用：

　　将一个命令的输出，作为另一个命令的输入

格式:

命令1 选项１.. | 命令2 选项２ | 命令3 选项3...

示例：

　　查看/etc/passwd 下用/bin/bash启动的 的用户有几个

grep "/bin/bash" /etc/passwd | wc -l

输出重定向:

输出分为两种:

1. 标准输出

2. 标准错误输出

标准输出重定向:

> 将一个命令的标准输出重定向到文件

>> 标准输出追加到一个文件

示例：

　　grep "/bin/bash" /etc/passwd > file.txt

标准错误输出重定向

2> 将一个标准错误输出重定向到文件

2>> 标准错误输出追加到文件

示例：

　　find /etc -name 'group' 2> erorr.txt

重定向所有输出

&>　　将所有输入重定向到同一个文件

&>> 此处略...

echo 命令:

作用：

　　将文本信息作为标准输出

格式:

echo "字符串"

示例：

echo 'hello'

练习：

　　已知有一个文件的内容有5行文本

　　aaaaaa

bbbbbb

Cadadjf

dddddd

eeeeee

请head, tail, echo 命令组成将文件的第三行改为cccccc 后存入另一个文件 abcde.txt中

shell 编程

shell的种类

bash

sh

shell程序运行方法：

方法1:

$ bash 程序文件名

方法2:

$ chmod +x 程序文件名

$ ./程序文件名

shell的注释：

# 井号开头，直至行尾

第一行注释 #!:

作用：

指定解释执行此文件的程序是谁?

示例：

#! /bin/bash

进程管理及其命令：

什么是进程：

正在执行的程序

ps 命令：

作用：

查看进程相关的信息

格式:

ps [选项]

示例：

ps

ps aux

kill 命令：

作用：

杀死进程

格式:

kill 进程的pid号

ctrl + c 终止进程

top 命令：

作用：

查看程序运行状态

退出:

Q键

练习：

已知有一个模板文件template.txt 的内容如下：

群通知

----------------

xxxxxx:

xxxxxx

----------------

群主：魏老师

另创建两个文件file1.txt 和file2.txt，在其中写入一定的内容

1. 用file1.txt的内容生成一个群通知文件n1.txt

2. 用file2.txt的内容生成一个群通知文件n2.txt

要求：写一个shell程序，生成以上两个文件

提示:

head, tail, cat, 输入重定向

网络管理及命令:

什么是IP地址:

192.168.1.100 (IPv4)

(0~255).(0~255).(0~255).(0~255)

ifconfig 命令

作用：

查看和设置网络配置

示例：

ifconfig

ifconfig eth0

注：windows下用ipconfig命令

ping 命令:

作用：

测试网络的连接状态

格式：

ping IP地址或域名

例如：

　　ping 172.60.50.78

ping www.baidu.com

hostname 命令:

作用：

　　 查看和设置主机名

　　　　示例：

　　　　　　hostname # 查看主机名

ssh 命令：

　　作用：

　　　　远程登陆到一台主机

格式：

　　ssh [用户名@]ip地址或域名

示例：

　　$ ssh weimz@172.60.50.78

exit 命令

作用：

　　退出登陆

who 命令：

　　作用：

　　　　查看当前主机有哪儿几个用户登陆

　　scp 命令:

作用：

　　远程复制文件或文件夹

格式:

scp [选项] 源文件地址 目标文件地址

远程文件地址格式:

用户名@IP地址:绝对路径

示例：

# 复制我的一个文件day03.tar.gz到172.60.50.78的 /home/weimz/下

　　$ scp ./day02.tar.gz weimz@172.60.50.78:/home/weimz/

系统的关机，重启等命令

poweroff # 现在关机

shutdown -h now # 现在关机

shutdown # 一分钟后关机

shutdown -h 小时:分钟 # 按预定时间关闭系统

shutdown -c # 取消按预定时间关闭系统

shutdown -r now # 重启

reboot 重启

练习：

　　1．把你自己今天的笔记打包成为: day04.tar.gz

　　2. 用scp 把笔记发送到你同桌的电脑上备份

　　3. 用ssh 登陆，在你同桌的电脑上创建一个你自己的目录

　　4. 把day04.tar.gz 复制到你自己的目录里

5. 解包day04.tar.gz到你自己目录．用vim 查看文件信息是否与你自己电脑上的相同

**day05笔记:**

ssh 命令：

　　远程登陆另一台Linux/Unix主机

IP 用户名 密码

scp 命令：

　　远程复制

scp 用户名@IP或域名:路径

密码

poweroff

shutdown -h now

shutdown -h 小时: 分钟

shutdown -r now

reboot

echo 制作一个标准输出

输出重定向:

>

>>

2>

2>>

&>

&>>

数据流(先后顺序关系)

　　管道 |

进程

　　启动用户 PID 运行状态 CPU占比 内存% 命令名

　　　　　　　　　　　　　　　　睡眠S

　　　　　　　　僵尸Z

　　　　　　　　停止?

　　　　　　　　运行R

kill　杀死进程

top 命令

ping 命令

who 命令

**day05笔记：**

《python 基础》

shell

if while for 正则表达式

python 创始人: Guido van Rossum

原始的二进制机器指令

010101010 二进制代码

11010001 ADD <<<--指令

11010010 SUB

11010011 MUL

.....

11010000 xxx

汇编语言 ADD SUB

编译型语言:(静态语言)

B C C++ PASCAL

解释执行的语言（动态语言):

basic, java, Python, C#

处理器的体系架构

x86 处理器 (286,386,586)

指令的种类不超过200条

arm 处理器

指令的种类五十几条

51系列 处理器

Java C C++ Python ....... C#

python的官网:

网址: https://www.python.org

python的版本:

v2.7

v3.5

v3.7

安装:

www.python.org/getit

python的解释执行器:

CPython(C语言开发)

/usr/bin/python3

/usr/bin/python

Jython(Java开发)

IronPython(.net开发)

进入python解释执行器:

$ python3

退出:

>>> exit()

>>> quit()

>>> <ctrl+d>

ctrl + L 清屏

第一个python程序：

hello.py

写入内容见:

hello.py

python 的执行：

　　　$ python3 xxxx.py

python的核心数据类型　：

　　数字(整数，浮点数, 复数，布尔型)

　　字符串等

整数:

10

-10

9999999999

0xabcd(十六进制)

0b1110(二进制形式)

0o1234(八进制形式)

浮点数:

3.14

6.18e-1 (等同于 0.618 e代表10, -1代表10的-1次方)

2.9979e8(光速)

复数　complex

分为两部分:

实部 (real)

虚部　(image)

注：虚部以j或J结尾的数

表示方式：

　　(1+2j)

(-2j)

(-100+200j)

-2j #　等同于 (0-2j)

布尔类型 bool

用来表示真和假两种状态的类型

　　　True 表示真(条件满足或成立)

False 表示假(条件不满足或不成立)

表达式和运算符

　　　表达式是由数字或数字和运算符组成，在计算机语句中用来做一件事情并返回结果

算术运算符:

+　　　加法

- 减法

\* 乘法

/ 除法

// 地板除

% 求余

\*\* 幂运算

/ 除法:

　　等同于数学中的除法

1 / 3 # 等于 0.33333333

// 地板除

除的结果去掉小数部分，向下取整

1 // 3 得 0

7 / 3 # 得 2.3333333

7 // 3 # 得 2

7.5 // 3 # 得 2

7.5 // 3.5 # 得 2

% 求余:

7 % 3 # 得 1

7.5 % 3 # 得 1.5

7.5 % 3.5 # 得 0.5

\*\* 幂运算

x \*\* y　意为 x的y次方

4\*\*2 # 得 16

3.0 \*\* 2.5 # 得

练习：

　　一个商店卖西瓜7元一个, 你带20元能买几个西瓜，找零多少？

print函数:

简单格式:

print(要打印的数据, 第二个要打印的数据,...)

练习：

　　写一个 triangle.py 程序　打印如下三角形

\*

\*\*\*

\*\*\*\*\*

\*\*\*\*\*\*\*

变量

变量名的命名方法：

　　变量名必须为字母或下划线开头，后跟字母或数字或下划线

合法变量名

a a\_1 a\_200 a\_b\_c\_d

aaaaa bcd zzz c1b2

abcd ABCD

a1 \_1234 \_ABCD \_hello\_123

a2

a3

不合法的变量名:

1a %100 %ABC% $ABCD

赋值语句:

变量名 =　表达式

del 语句:

删除变量

格式：

　　del 变量名

复合赋值运算:

+= -= \*= /= //= %= \*\*=

格式:

变量 += 表达式　　# x += y 等同于 x = x + y

练习：

　　　温度转换

摄氏温度 = 5/9\*(华氏温度-32)

开氏温度 = 摄氏温度 + 273.15

问：

　　100华氏温度，转为摄氏温度是多少度?

　　转为开氏温度是多少度?

比较运算符

< 小于

<= 　小于等于

> 大于

>=　　　大于等于

== 等于

!= 不等于

语法:

左表达式 < 右表达式

数值对象构造(创建)函数

　　float(obj) 用字符串或数字转换为浮点数，如果不给同参数则返回0.0

int(x, base=10) 用数字或字符串转换为整数,如果不给出参数则返回0

complex(r=0.0, i=0.0) 用数字创建一个复数(实部为r,虚部为i)

bool(x) 用x创建一个布尔值

函数的调用：

　　　函数名(传参列表)

基本输入函数:

input('提示字符串')

返回输入的字符串(仅python3)

示例:

输入一个数字，代表圆的半径

打印出这个圆的面积

见: area.py

练习：

　　输入一个人的北京社保基数(3082~23118)元：

计算这个人的五险一金

　　社保分为:

养老，医疗，失业，工伤，生育

一金:

公积金:

个人缴费比例 单位缴费比例

养老　　　　　 8%　　　　　　　　　　　　　19%

失业(城镇) 0.2%　　　　　　　　　　　0.8%

失业(农村) 　0%　　　　　　　　　　　　0.8%

工伤　　　　　 0%　　　　　　　　　　　　0.5%

生育　　　　　 0%　　　　　　　　　　　　0.8%

医疗　　　　　 2%+3元　　　　　　　　 10%

公积金　　　　　 12%　　　　　　　　　　　12%

要求:

写程序，输入您的社保基数，打印出各项要缴纳的费用明细和总和(公司部分总和和个人部分总和)

**day06笔记:**

python

两个版本:

v2.7

v3.5

python 两种执行模式:

交互模式:

python3<回车>

解释执行吖模式:

python3 xxxxx.py

python 的注释:

# 开头直至行尾

数据类型：

　　数字类型（整型int, 浮点型float,复数complex,

布尔型bool)

字符串

列表

元组

...

3.3 == 3.3333333333

算术运算符:

+ - \* / // % \*\*

a 是一个数字

b 是被除数

x = a // b

y = a % b

b \*　x + y == a

运算符的优先级:

变量

取名字:

第一个字符[a-zA-Z\_]

第二个字符和以后[a-zA-Z0-9\_]

变量名不能是关键字:

del, if, True, False, None, while ...

赋值语句

变量名 = 表达式（数字,运算符,函数调用,类,....)

作用：

　　创建和修改变量

del 语句:

删除变量

复合赋值语句:

+= \*=

语法：

　　 变量名　+= 表达式

比较运算符

> >= < <= == !=

函数:

print函数

input函数

float

int

complex

bool

函数调用：

　　函数名(传参列表)

练习答案见:

code/shebao.py

**day06笔记:**

if 语句

作用：

让程序根据条件选择性的执行某条语句或某些语句

语法：

if 真值表达式1:

语句块1

elif 真值表达式2:

语句块2

elif 真值表达式3:

语句块3

...

else:

语句块4

示例见:

if.py

if\_elif.py

练习：

做游戏：

假设您与电脑做猜拳游戏，电脑出"石头"

您来选择输入:"石头", "剪刀", "布"

1) 石头

2) 剪刀

3) 布

请输入:

您输入1: 输出:平局，电脑是石头，您出的是石头

以下自己定义...

布尔运算:

not and or (对应C语言的 ! && || )

布尔非操作 not

语法:

not x

作用:

对 x 进行布尔取非，如bool(x) 为True,则返回False, 否则返回True

x = 100

if not x:

print("not x 为真")

else:

print("not x 为假")

示例：

not.py

布尔与操作 and

语法:

x and y

注： x, y 代表表达式

作用：

优先返回假值对象

当x的布尔值为False时，返回x,否则返回y

布尔或操作 or

语句:

x or y

作用：

优先返回真值对象

当x的布尔值为真时返回x,否则返回y

练习：

输入一个人的年龄:

如果年龄小于零提示: 输入不合法

如果年龄大于120提示: 输入不合法

否则提示: 输入年龄合法

pass 语句

作用：

用来填充语法空白

字符串 str

作用：

用来记录文本信息

表示式方法

'hello'

"hello"

"""hello"""

'''hello'''

字符串的运算:

+ += \* \*=

"你好" + "中国"

'你好中国'

练习1：

输入一段文字：

请输入：明天放假!<回车>

打印:

您刚才说的是:"明天放假!"

练习2:

输入一个整数n，打印一个长方形，n代表长方形的宽的字符数

例如:

请输入:6

打印:

+------+

| |

+------+

请输入:9

打印:

+---------+

| |

+---------+

字符串的比较:

<

<=

>

>=

==

!=

说明：

字符串的比较是按 ascii 编码值进行依次比较

见:

$ man ascii

示例:

'A' < 'B' # True

'ABB' < 'ABC' # True

'ACB' > 'ABC' # True

'AD' > 'ABC' # True

'AB' < 'ABC' # True

'ABC' == 'abc' # False

len(x) 函数

示列：

s = input("请输入:")

print("您输入", len(s), "个字符")

练习：

写一个程序分别输入三行字符串:

将这些字符串进行右对齐显示:

$ python3 input.py

请输入:abc

请输入:abcdefg

请输入:12345

打印如下：

abc

abcdefg

12345

**day07笔记:**

if 语句

判断条件，选择性的执行某些语句

语法：

if 条件1:

条件1成立时执行的语句块

elif 条件2:

条件2成立时执行的语句块

elif 条件3:

条件3成立时执行的语句块

....

else:

语句块

布尔运算

not, and, or

pass 语句

字符串

'aaa'

"aaa"

'''aaa'''

"""aaa"""

字符串的运算:

+ += \* \*=

> >= < <= == !=

**day07笔记:**

用转义序列代表特殊字符:

\' 代表单个引号

\" 代表单个双引号

\n 代表换行

让计算机输出如下字符串:

aaa'b'"c",'''d''',"""e"""

字符串索引: index

字符串索引上以下标方式获取字符串的某个字符

语法：

字符串[整数值]

说明：

0代表第一个字符,1代表第二个字符，以此类推

-1 代表倒数第一个字符，-2 代表倒数第二个,以此类推

示例：

s = 'abcdefg'

print(s[0]) # 'a'

print(s[-1]) #'g'

python3 中常用的序列函数

len(seq) 返回字符串的长度

max(x) 返回字符串的最大值字符

min(x) 返回字符串最小值元素

字符串编码转换函数:

ord(c) 返回一个字符串的 UNICODE 值

chr(i) 返回i这个值对应的一个字符

字符串相关的方法：

如何用python3帮助查看字符串的方法：

>>> help(类型名) # 查看某个类型的文档

例:

>>> help(str) # 查看字符串类型的文档

方法用法:

字符串.方法名(方法传参)

s =input("请输入一个整数:")

i = int(s)

if i < 0:

print("您输入提负数")

练习：

用键盘输入任意一个字符串，判断您输入的字符串内有几'i'字符并打印

打印出您输入的字符串的总长度

(3分钟时间)

字符串格式化表达式:

语法：

格式字符串 % 参数值

或

格式字符串 % (参数值1, 参数值2, ...)

作用：

生成一定格式的字符串

格式化字符串中的占位符的类型码

%s 字符串

%c 整数传为单个字符串

%d 整数

%f 浮点数

练习：

写一个程序，定义一个合同的格式:

甲方:\_\_\_\_\_ 乙方:\_\_\_\_\_\_\_

合同金额:\_\_\_\_\_\_

......

日期:\_\_\_\_年\_\_月\_\_日

在程序输入时，多次输入相应的参数,打印出完整的合同文本信息

如:

请输入甲方姓名: xxxx

请输入已方姓名: yyyy

请输入金额: 10000

请输入年: 2018

....

循环语句：

while 语句

语法:

while 真值表达式:

语句块1

else:

语句块2

说明：

else 部分可以省略

作用：

可以根据条件重复的执行语句块1

练习：

输入一个整数n，打印一个长方形，

n代表长方形的宽的字符数和高的行数

例如:

请输入:6

打印:

+------+

| |

| |

| |

| |

| |

| |

+------+

请输入:4

打印:

+----+

| |

| |

| |

| |

+----+

练习:

1. 输入一个数n，打印, 1 + 2 + 3 + 4 + .... + n的和

2. 写程序求:

1/1 + 1/3 + 1/5 + 1/7 + ... +1/99的和

break 语句:

作用：

用于循环语句中，当执行时终止循环

说明：

当break执行时，while 语句的else子句不会被执行

示例见：

break.py

列表 list

列表是一种容器，用来存储数据

列表内的数据有先后顺序关系

列表是可以被改变的容器

列表里可以存放任意类型的数据

列表的表示方式:

[]

列表的运算:

+ += \* \*=

> >= < <= == !=

# 比较的规则同字符串比较规则完全相同

示例:

[1, 2, 3] > [1, 2, 2] # True

[1, 2, 3] < [1, 3, 2] # True

列表的索引操作:

1. 同字符串的索引操作相同．

2. 列表是可变的容器，可以通过列表索引赋值来改变列表的元素

示例：

　　L = [1, 2, 3, 4]

print(L[0]) # 1

print(L[2]) # 3

print(L[-1]) # 4

L[3] = 5

print(L) # [1, 2, 3, 5]

del 语句，用于删除列表的元素

示例：

L = [1, 2, 3, 4]

　　del L[3]

print(L) # [1, 2, 3]

列表的方法见:

list\_xxxxxx.html

练习：

1. 写程序求:

　　1/1 - 1/3 + 1/5 - 1/7 + 1/9 ... +1/(2\*n-1)的和

n最大取:1000000

1. 打印这个和

2. 打印这个和 乘以4的值?(打印有惊喜)

2. 见:personal\_income\_tax\_xxxx.html

**day08笔记:**

字符串:

格式化字符串:

字符串 % (参数1, 参数2, ...)

字符串 % 参数

转义字符 \

'\'' '\"' '\n' '\\'

while 语句:

语法:

while 条件:

语句块1

else:

　　　语句块2

break 语句:

终止循环

while i < 7:

print(i)

i += 1

break

else:

print("while结束")

列表:

可变的容器

增,删,改,查

增

L.append(x)

L.insert(index, x)

删

L.clear()

del L[index]

改

L[index] = x

查

L.find(x)

函数 function

函数是可以重复执行的语句块, 可以重复调用

作用：

1. 定义用户级别的函数

2. 用于封装一些语句，将这些语句看作一个整体

语法:

def 函数名(参数列表):

语句块

示例:

def myadd(a, b):

return a + b

print("100 + 200 = ", myadd(100, 200))

练习：

写一个函数print\_even, 传入一个参数n代表终止整数，打印

2 4 6 8 ..... n 之间的所有偶数

函数定义参数如下：

def print\_even(n):

此处自己完成

# 测试调用

print\_even(10)

# 打印结果

2

4

6

8

10

None 空值对象

在函数定义的内部，如果没有return 语句，则在函数执行完毕后默认返回None

return 语句

语法:

return [表达式]

作用：

用于函数中，结束当前函数的执行，返回到调用该函数的地方，同时返回一个对象的引用关系

练习：

1. 编写函数fun, 其功能是：计算并返回下载多项式的值

Sn = 1+1/2+1/3+1/4+.....1/n

def fun(n):

....

print(fun(3)) # 1.8333333333333333

2. 编写函数，计算下列级数的和:

Sn=1/(1\*2) + 1/(2\*3) + .... + 1/n(n+1)

3. 编写函数fun, 其功能是：计算并返回下载多项式的值

Sn = 1 + 1/(1+2) + 1/(1+2+3) + 1/(1+2+3+4) + ..... 1/(1+2+3+...+n)

def fun(n):

...

print(fun(100))

看懂下面的代码:

def mymax(a, b):

if a > b:

return a

return b

作用:

给出两个参数，返回较大的一个

练习：

1. 定义一个函数，返回三个参数的最大值

def mymax2(a, b, c):

返回最大的一个(要求: 不允许用max函数)

列表相关的函数:

len(x) 返回列表或字符串的长度

max(x) 返回列表中的最大值

min(x) 返回列表中的最小值

sum(x) 返回列表中所有元素的和

随机模块:

导入随机模块:

import random as R

此时可以用R来调用此模块内的函数来生成随机数

R.choice(列表) 从列表中随机返回一个元素

练习1：

用随机模块写一个猜拳游戏,随机产生让计算机生成一个值["石头", "剪刀", "布"]中的一个

让用户输入出拳的选择:

0) 石头

1) 剪刀

2) 布

3) 退出猜拳

循环猜拳，每次打印出猜拳的结果(包含 计算机和您都出的什么拳)

直到用户不想玩了输入"3" 结束程序

提示:

L = ["石头", "剪刀", "布"]

S = [0, 1, 2]

import random as R

select = R.choice(S) # select 为计算机选的选项

print("计算机选择的是:", L[select]) # ...

**day09笔记:**

函数:

函数定义:

def 函数名(形参变量1, 形参变量2, ...):

语句块

函数调用:

函数名(实参1, 实参2, ...)

return 语句

语法:

return 表达式

示例：

return 等同于 return None

len(x)

max(x)

min(x)

sum(x) 求和

猜拳游戏答案见:

guess\_sign.py

**day09笔记：**

模块 Module

什么是模块

模块是一个包含有一系列变量，函数，类等组成的程序组

作用：

1. 让一些相关的变量，函数，类等有逻辑的组织在一起,使逻辑结构更加清晰

2. 模块中的变量，函数，类等可提供组其它模块或程序使用

模块的导入:

三种导入语句:

import 语句:

import 模块名1 [as 模块新名1]

from import 语句:

from 模块名1 import 属性名[as 属性新名1]

from import \* 语句:

from 模块名 import \*

常用模块:

math 数学模块

random 随机生成模块

time 日期时间模块

sys 系统信息模块

查看模块信息的方法：

>>> help(模块名)

如:

>>> import math

>>> help(math)

练习：

输入一个正方形的面积的是:

求其边长是多少?

math.sqrt(x)

os 模块 (operator system)

os.mkdir(path, mode=0o777, \*, dir\_fd=None) 创建一个文件夹

os.rmdir(path, \*, dir\_fd=None) 删除一个文件夹

os.chdir(path) 改变当前的工作路径

os.system(command) 运行 linux 命令

os.listdir(path=None) 返回path目录下的所有的文件和文件夹的列表

os.path.isdir(path) 判断一个路径是否是文件夹

练习：

写一个程序，任意输入一个文件名称，在此文件夹内创建如下文件夹和文件:

输入文件名称

|

+---src(文件夹)

|

+---bin(文件夹)

|

+---lib(文件夹)

|

+--- mylib.py(文件)

|

+--- \_\_init\_\_.py(文件)

面向对象编程

类 class 和 对象 object

能够实现'面向对象'编程的语言有:

Python, Java, C++, C#, Swift, Objective-C, Dephi

能够实现'面向过程'的编程语言有:

C, Basic, Pascal

class object(instance)

车(类) -------> BYD E6(京A.88888) 对象(实例)

\

\------> BMW X5(京B.00000) 对象(实例)

船(类) -----> 辽宁舰

\

\----> 货船(23987)

\

\---> 鱼船(99887)

类的创建语法:

class 类名(继承列表):

类变量的定义

实例方法的定义

类的作用：

1. 用类，可以创建一个或多个此类相关的对象(实例)

2. 类内定义的变量和方法能被此类所创建的所有对象所共同拥有

示例：

# 创建一个名为Car的类

class Car:

pass

实例创建表达式:

类名([创建传参列表])

作用：

创建这个类的实例对象,并返回此实例对象的引用关系

示例：

bydf6 = Car() # 创建一个Car类型的对象

bmwx5 = Car()

实例说明：

实例有自己的作用域和名字空间，可以为该实例添加变量(实例变量/属性)

实例可以调用类中的方法

实例可以访问类中的类变量

实例变量(也叫属性 attribute)

在python程序中，每个实例可以有自己的变量，每个实例变量有自己独立的作用域和名字空间

实例变量的调用语法:

实例.变量名

实例变量的赋值规则

1. 首次为实例变量赋值，则创建此实例变量

2. 再次为实例变量赋值，则改变变量的绑定关系

删除实例变量:

del 语句

用法:

del 实例.实例变量

dir函数:

dir([obj])

1. 如果给定参数，返回这个参数所有属性的字符串列表

2. 如果不给定参数，则返回当前作用域内所有变量的字符串列表

例如：

dir(bydf6)

dir(bmwx5)

实例方法: method

实例方法是类内定义的方法，此方法为所有此类的实例所共同拥有，每个对象都可以调用此实例方法

实例方法语法:

class 类名(继承列表):

def 实例方法名(self, 形参名1, 形参名2,...):

语句块

实例方法说明：

实例方法的实质是函数，是定义在类内的函数

实例方法的第一个参数代表调用这个方法的实例，一般命名为"self"

实例方法属于类的属性

调用语法:

实例.实例方法名(调用参数)

明日预告:

初始化方法 \_\_init\_\_(self, ....)

１．练习：

　　用数学模块中的math.factorial(x) 求如下极数的和:

Sn = 1 + 1/1! + 1/2! + 1/3! + ... + 1/100!

２．　看懂class Car类的调用顺序

**day10笔记：**

模块:

math

pi 数据

sin(x), cos(x) 函数

os 模块

mkdir(path)

rmdir(path)

chdir(path)

system(command)

...

random

choice(lst)

导入模块的语法：

import 模块名 [as 别名]

from 模块名 import 函数名,变量名,类名

from 模块名 import \*

帮助：

>>> help(模块名)

>>> help(函数名)

>>> help(对象名)

>>> help(类名)

面向对象:

类 class 和 对象 object

语法:

class 类名(继承承列表):

实例方法的定义

类变量的定义

实例变量(属性)

实例方法(行为)

class Car:

def run(self, speed):

pass

**day10笔记：**

初始化方法:

作用：

对新创建的空白对象添加实例变量

说明：

1. 实初化方法的名称必须是\_\_init\_\_ 不可改变

2. 初始化方法会在实例生成后自动调用，且将实例自身通过第一个参数self传入\_\_init\_\_方法

3. 初始化方法内如果需要return语句返回，则只能返回None

语法格式:

class 类名(继承列表):

def \_\_init\_\_(self[, 参数列表]):

语句块

示例见:

init\_method.py

init\_method2.py

析构方法:

语法:

def 类名(继承列表):

def \_\_del\_\_(self):

语句块

作用：

在对象销毁时自动调用,主要做该对象的清理工作

示例：

见 del\_method.py

\_\_class\_\_属性

\_\_class\_\_ 属性绑定创建此实例的类

作用：

1. 借助于此属性来创建同类实例

2. 借助于此属性来访问类变量

示例见: class\_attribute.py

类变量

是指类的变量,此变量属于类，不属于此类的实例

类变量可以通过该类直接访问

类变量可以通过类的实例间接访问

类变量可以通过此类的对象的\_\_class\_\_属性间接访问

例如：

class Human:

total\_count = 0 # 此变量为类变量

# 用它来记录来记录对象的个数

....

用于类的函数:

isinstance(obj, cls) 返回这个对象obj是不是某个类cls的实例，如果是返回True, 否则返回False

type(obj) 返回对象的类型

对象的属性管理函数

hasattr(obj, name) 用给定的name返回对象obj是否有此属性，此种做法可以避免getattr(obj，name)时引起错误

getattr(obj, name[, default]) 从一个对象得到对象的属性, getattr(x, 'y') 等同于x.y, 当属性不存在是，如果给出default参数，则返回default,如果没有给出default,则产生一个AttributeError错误

示例：

class Car:

pass

c1 = Car()

c1.color = "白色"

print(hasattr(c1, "color")) # True

print(hasattr(c1, "brand")) # False

c2 = Car()

c2.color = "黑色"

print(getattr(c1, "brand")) # 报错,因为c1没有brand属性

练习：

1. 看懂 Human类，及对象的相互关系.

见: human.py

**day11笔记：**

初始化方法 \_\_init\_\_ 和析构方法 \_\_del\_\_

class xxxx:

def \_\_init\_\_(self, 形参1, 形参2, ...):

...

def \_\_del\_\_(self):

....

a = xxxx(实参1, 实参2, ...)

\_\_class\_\_ 属性(对象的实例变量)

类变量 ---> 实例变量

类 对象1(实例)

对象2..

isinstance(obj, cls) 判断一个对象是否是某个类的实例

type(obj) 返回对象的类

属性相关的函数:

hasattr(obj, name)

getattr(obj, name[,default]) ...

示例：

def getattr(obj, name, default=None):

if hasattr(obj, name):

return getattr(obj, name)

else:

return default

**day11笔记:**

封装 enclosure

1. 封装的目的是让使用者通过尽可能少的变量名(或方法)操作对象

2. 封装是指隐藏类的实现细节，让使用者不用关心这些细节

私有实例变量和方法:

python类中，以双下划线'\_\_'开头,不以双下划线结尾的标识符为私有成员。

私有属性在子类和类外部无法直接使用

私有方法在子类和类外部无法直接使用

继承(inheritance) 和 派生(derived)

为什么要继承/派生

继承的目的是延续旧的类的功能

派生的目的是在旧类的基础上添加新的功能

作用：

1. 用继承派生机制，可以将一些共有功能加在基类中，实现代码共享;

2. 在不改变超类代码的基础上改变原有的功能

继承/派生名词:

基类（base class)/超类(super class)/父类(father class)

派生类(derived class) / 子类(child class)

单继承的语法：

class 类名(超类名):

...

示例： inherit.py

\_\_base\_\_属性

用来记录此类的基类

object类是一切类的原始父类

覆盖 override

覆盖是指在有继承关系的类中，子类中实现了与基类(超类)同名的方法，在子类实例调用该方法时，实际调用的是子类中的覆盖版本，这种现象叫覆盖

用于类的函数：

issubclass(cls, 类class或者多个类的元组tuple) 判断一个类是否继承自其它的类，如果此类是class或tuple中的一个派生子类，则返回True, 否则返回False

示例：

class A:

pass

class B(A):

pass

class C(B):

pass

issubclass(C, B) # True

issubclass(C, int) # False

issubclass(B, (int, str)) # False

issubclass(B, (int, str, A)) # True

super 函数

super(type, obj) 返回绑定超类的实例（要求obj必须为type类型的实例)

super() 返回绑定超类的实例，等同于super(\_\_class\_\_, 实例方法的第一个参数),必须用在方法内部

示例见: super.py

日记本管理系统项目

notebook(日记本)

|

+ 1页 (时间:年,月,日，天气, 内容)

|

+ 2页

事件：

１．添加页

２．修改当天日记

３．列出所有日记信息(不包括内容)

４．查看日记内容

对象:

1. 日记本(1个)

2. 页(多个)