**第四周任务**

## 基本语法练习

条件、循环、函数

1. 判断给定的三个数a, b, c能不能构成一个三角形,直接返回True 和 False即可。（不用考虑浮点误差）

def is\_triangle(a, b, c):

1. 给定的三个数a, b, c，判断有几个实数根,返回根的数量，以及相应的根。（不用考虑浮点误差）

def delta(a, b, c):

1. 给定一个自然数, 判断n是不是质数（可以遍历range(n)，也可以考虑遍历range(0,math.floor(n\*\*0.5)，不考虑复杂的素性判定算法）

def is\_prime(n):

1. 判断一个（整数或字符串表示的）年份是不是回文年，譬如 2002， 1991， 1221都是回文（可以采用len函数提取字符串长度，并用str[i]提取字符串中在i处的字符。这里是训练循环的写法，所以尽量不要用slicing（str[;;-1]）和reversed()函数）

def is\_palindrome\_year(year):///

1. 实现函数

def f(x):(这里最后一个是 x >= 2)

1. Write a program to read a series of numbers from the console, such that if the input number :

x == 0: terminate the input;

x == -1: read a new number;

otherwise, output the inverse of the number (19-->91, 102-->201， 123—>321) first, then read a new number.（注意类型转换）

1. Given a positive integer n, count how many zeros at its end.（这里指的是十进制下的零）
2. The Hamming distance between two integers is the number of positions at which the corresponding bits are different (即对应的二进制表示，有多少位不同)

def Hamming(a,b):

以a=3,b=7为例，先将a,b转换为2进制表达11,111,对位检查它们每个位上面是不是相等。Hamming(3,7)=1（可以考虑通过位运算、用bin()函数转换成字符串以及string对象的.count()函数计算差异位）

1. 3n+1问题。

首先判断对于一个给定整数猜想成立（即Collatz猜想）

def is\_valid(n):

对于范围的数据，判断猜想成立（这里的意思是对于每一个该范围内的数n，调用is\_valid()函数，只需要返回是否所有该范围内的数都最后能够达到1，不需要print中间过程。）

1. China Remainder Theorem (中国剩余定理)

今有物不知其数,三三数之剩二,五五数之剩三,七七数之剩二,问物几何?

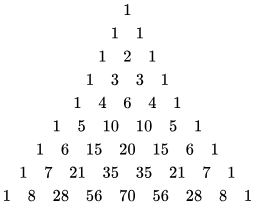
三人同行七十稀,把除以3所得的余数用70乘.

五树梅花日一枝,把除以5所得的余数用21乘.

七子团圆正半月,把除以7所得的余数用15乘.

除百零五便得知

Please write a program to find all the solutions within 0 and 100，000.（这里是指找出所有“三三数之剩二,五五数之剩三,七七数之剩二”的0到100000之间的整数）

1.  Pascal三角（中国又名杨辉三角），每一行对应于多项式的系数。

def Pascal\_triangle(n): # 按照上图输出大小为n的杨辉三角

(a). 实现def fractal(n): n! （这个需要自己手写）

(b). 数学中n个元素选取k种, ，实现def （这个可以直接通过阶乘来做（可以调用之前写的fractal函数），也可以考虑先作约分把k!或者(n-k)!约掉（这样可以减少乘法次数从而提升执行效率））

(c). 在此基础上，输出Pascal 三角

注意：可以认为每个系数只占一位，不考虑系数的具体长度

## Linux学习

在过去的Ubuntu学习中，我们讲解了浏览、创建、修改文件以及目录的一系列操作，学会了如何在Ubuntu中用命令行运行Python文件。

在一个操作系统中，同时运行有多个程序，每个程序叫做一个进程（Process）。每个进程内部可能会有多个线程（Thread）在同时运行，以充分使用现在多核CPU的性能。

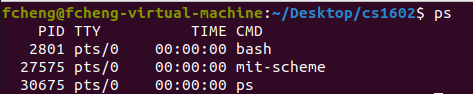
<https://en.wikipedia.org/wiki/Process_(computing)>

<https://en.wikipedia.org/wiki/Thread_(computing)>

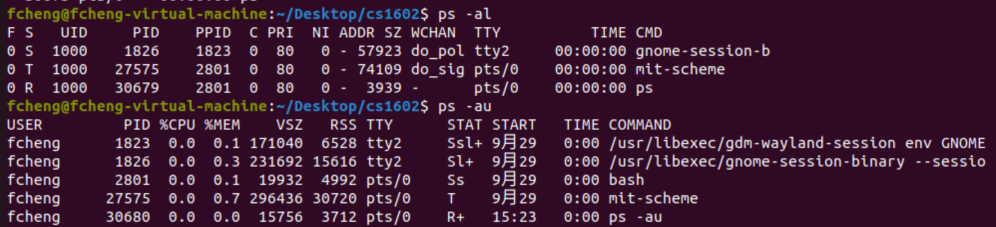
通俗的讲，每个计算机程序被执行时，都会对应于一个CPU中的进程。进程和进程之间是互相独立的。每个进程内部的多个线程，会共享代码、数据、内存和CPU。

Linux提供了ps（process status）命令，用于显示当前进程的状态，类似于 windows中的任务管理器。

插一句小知识点：Linux的shell中提供了clear命令，可以用来清楚屏幕的信息。



ps同时提供了很多参数，用来控制线程具体参数信息，常用的有ps -al， ps -au, ps -aux



ps -aux会输出很多信息，这个时候可以用clear命令清楚掉。

图片包含 图形用户界面

描述已自动生成

我们在使用Ubuntu过程中，可能会遇到程序死机、或者是运行时间过长，希望关掉程序的问题。这个时候，我们需要用到kill命令。请注意，不要随便kill掉不认识的线程，以免系统崩溃。

使用kill命令需要使用sudo权限。sudo的内容，我们前面讲过。

在文件夹中建议一个busy.py文件，编写如下程序。（nano？）

import time

for i in range(1000,-1,-1):

print(f'\r{i:4}%', end='')

time.sleep(.1)

print("\nDone")

用命令python3 busy.py执行，可以很清楚得知这个程序的意思是从1000倒数，中间停顿0.1秒。考虑到程序要执行100s，我们希望退出。退出程序的命令主要有两个

ctrl + c：结束当前程序的运行，程序不再运行，退出程序界面。

ctrl + z：挂起当前程序的运行，程序在后台运行，退出程序界面。

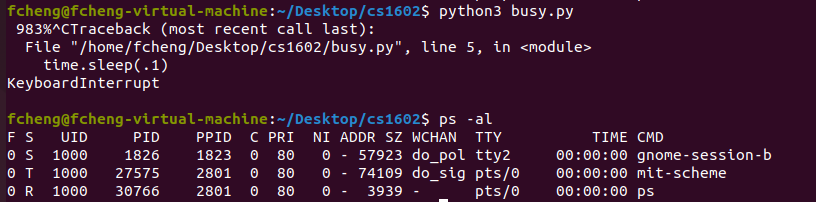
还有两个命令，可以用来控制程序在控制台的输出：

ctrl + s, 中断控制台输出

ctrl + q，恢复控制台输出

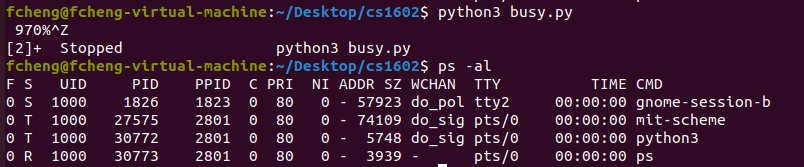
对于控制台ctrl+z挂起的程序，其并没有被终止运行，而是在背后运行。可以用fg命令恢复运行。 对busy.py程序fg后，会继续看到输出。

执行busy.py, 中间用ctrl+c退出，会收到一个Python的键盘(KeyboardInterrupt)异常。Python异常以后会学到。



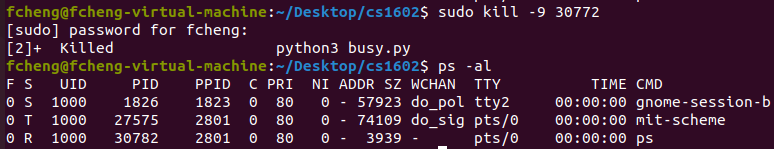
用ps -al查看，发现没有python相关的线程信息。

如果用ctrl + z退出busy.py的运行，再用ps -al查看，会看到。



出现一条python3的线程。其中，对我们有用的信息是第四列的PID。每个线程都有一个唯一的pid。可以看到python3的pid为30772。（pid： process id的意思，具体的id号会不同。）

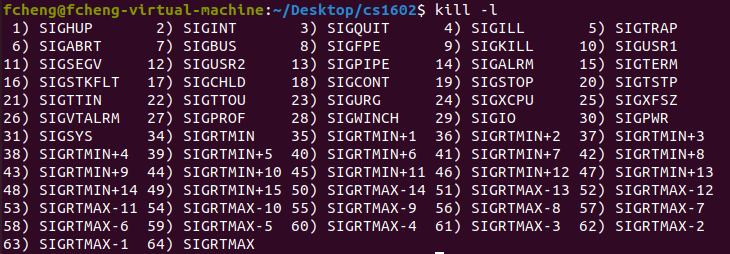
运行如下命令 sudo kill -9 30772, 就可以结束线程30772了



sudo kill -9 pid

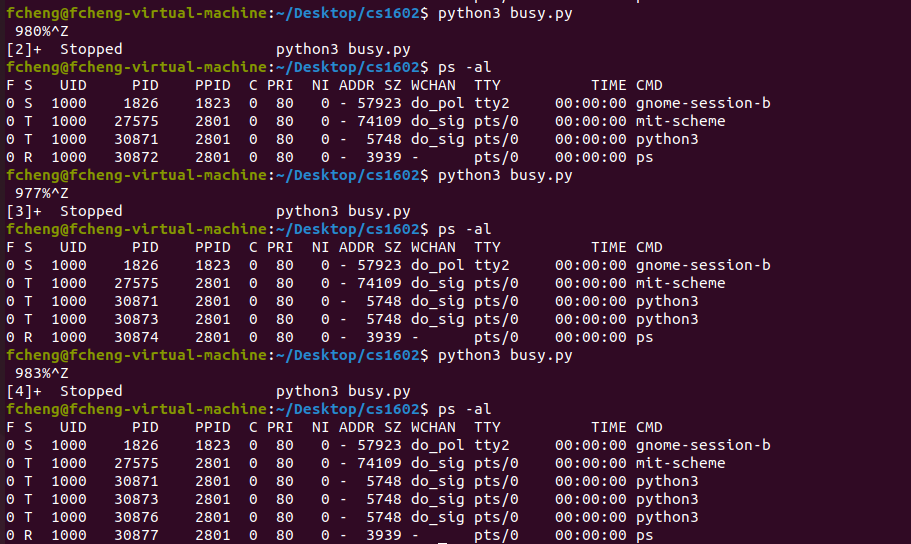
-9是ps命令的一个参数，表示彻底结束程序。可以查看kill的手册，有更多的命令。但大多数时候，用户都是非常简单粗暴的以 -9 结束进程。

kill -l （小写的字母L，不是数字1）命令可以查看kill的众多参数。

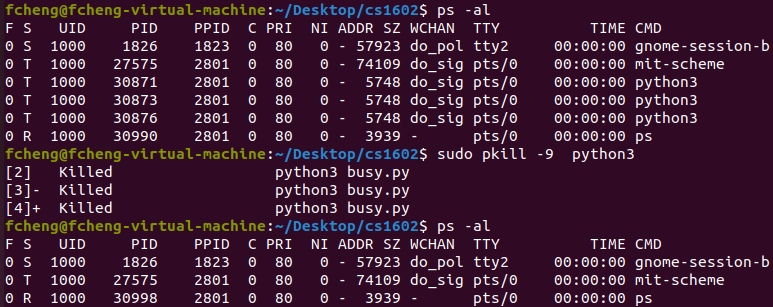


注意：在Ubuntu学习中，我们只需要记住常用的命令，对于一些具体的参数，也可以记得一些最常用的。在使用中，我们也会记住一部分。具体要使用某些特殊功能时，可以查看手册或者google。

在控制台下，运行busy.py, 然后ctrl+z退出，反复三次，最后我们可以看到三条python3的进程，它们具有相同的进程名称，但是pid不同，分别是30871, 30873, 30876。如果我们打算结束某个python3，就必须使用它对应的pid。



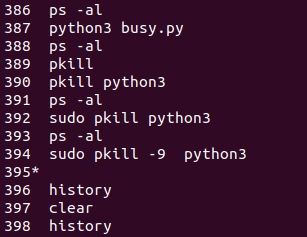
Linux中提供了pkill process\_name命令用来通过线程名来结束线程，但要慎重，以防误伤。



三条python3的线程全部被终止。

在控制台下执行命令的时候，我们可以连续用 方向键 Up 或者 方向键 Down 来重复使用过去执行过的命令，这样可以极大节省我们的时间。

history命令可以用来查看当前的命令历史



命令小结

sudo, ls, ll, cd, mkdir, touch

nano, cat, mv, cp, rm, rmdir

ps, kill, clear, history

目前为止，我们已经具备了python程序的编写、运行、维护等全套Ubuntu命令行操作的能力。初步可以用nano编辑一些python代码，体验一下最原始的程序开发的生活。

nano编辑 + python3命令行执行

可以考虑抛弃VS Code的舒适环境，在nano下纯手工编写规范的Python代码。

Linux的标准程序（例如nano, vim等等），一般提供了退出程序的标准方式。这种情况下，不建议直接ctrl+z/c退出程序。