计算概论 C 期中辅导

计算概论 C 期中辅导

Leading

Why learning computation?
Berkeley, CS 61A, 2000+ people joined

Python on song editing

Python on dealing with images

Python on Word Cloud

Python on Microsoft word, excel, etc.

Python is playing its role on everything!

Some suggestions
- Enjoy Python!
- Interact with it, instead of viewing it as only words
- **DO NOT** depend on ChatGPT or other AI tools so much! Think YOURSELF.
- Certain amount of **practice** is necessary.
- The pace would become faster in the following semester, so try to follow up. If there is any question, feel free to ask TAs or other students, or post it on website!

View Python Differently

Python 的命令行交互模式 (interactive mode)

Why interactive?
interactive 模式在熟悉概念和测试的过程中可以起到很直观的作用(python 是最好用的内置计算器！部分班级讲解了)

Step 1: 打开命令行、终端
Windows: 按 `win+R`，在弹出的小框中输入 `cmd`，就进入了命令行模式。

Mac: 点击下面的终端 `Terminal`，如果没有到应用程序中搜索终端或 `Terminal`

Step 2: 启动 **python**
在命令行/终端输入 python3 后敲回车，如果跳出 python 的安装界面或者无任何反应，则重新在命令行/终端输入 python 后敲回车。

如果输入成功，则会弹出来一系列的版本信息，然后看到下面这种符号，意味着 python 正在等待你的输入。

注：命令行交互过程不需要使用 input 函数，直接输入就可以

```
>>>
```

试一试：输入下面的结果，看看问号里面都是什么？

```
>>> 2024
?
>>> 1 + 2023
?
>>> 2 ** 10 + 10 ** 3    # **号是乘方的意思（# 号后面的内容代表注释，实际运行时不会执行到）
?
>>> 1 * (2 ** 3) + 4 * ((5 // 6) + 7 * 8 * 9)
?

>>> 6 // 3
?
>>> 6 / 3
?               # 为什么和//的结果不一样？这里一定要注意整除和除的区别 !!
>>> 7 // 3
?
>>> 7 % 3
?

>>> import math    # import 这句指的是导入了外界一个叫做 math 的库
>>> math.pi
?
>>> from math import *    # 这句的意思基本等同于上一句 import，把 math 库中所有内容导入
>>> pi             # 如果用的是 from a import b 的方式，在里面用到 b 的时候就不用 a.b 了，
比如这里不用 math.pi，直接说 pi
?
```

Expressions 表达式

表达式是 Python 的灵魂 !（CS 61A 课程中的第一个定义不是 int、float 类型，不是输入输出，而是 expression 表达式，足以看出表达式真的非常重要）

An expression **describes a computation**（描述计算过程）and **evaluates to a value**（表示一个数值）.

Python 中的表达式

上面例子中，所有问号前的一行都是表达式。事实上，表达式有很多种类型。可以是单个数字，可以带有加减乘除运算，也可以有如 max(a, b)的形式。

在 Python 中，所有表达式都可以转化为函数调用(call expressions)的模式。我们来看例子:

```
>>> max(2, 4)
4
```

这里 max 是一个内置的函数。通俗地理解，函数就是一个能实现特定功能的“黑箱子”，接受一串输入值，做一些特定的操作，然后返回另外一个值。函数名(变量 1，变量 2，...)是函数调用最基本的格式。

在这里，max 输入一串数，然后返回其中最大的数。

```
>>> from operator import add, mul
>>> add(2, 3)    # 和 2+3 起到一个作用
5
>>> mul(2, 3)
6
>>> max(1, 2, 3, 4, 5)   # max 可以输入多个值！
5
>>> mul(add(2, 3), add(3, 3))    # 表达式还可以嵌套
30
```

命名和赋值

只有表达式不够方便，Python 中是可以定义变量的。(注：在运行下面的代码之前，我们重启一下命令行。)

```
>>> pi
Traceback: ... NameError: name 'pi' is not defined   # python 报错了，告诉我们 pi 还没有定义呢
```

```
>>> from math import pi
>>> pi
3.141592...
>>> from math import sin   # 导入正弦函数
>>> sin
<built-in function sin>    # 告诉我们 sin 是一个内部的函数
>>> sin(pi / 2)
```

以上这些都是内置变量，我们可以自己定义变量吗？当然可以！

```
```

```
>>> radius = 10           # 我们定义了一个变量叫做 radius 半径，然后把它和 10“绑定”了，这个过程叫做赋值(evaluation)。
>>> radius
10
>>> 2 * radius
20
>>> area, circus = pi * radius * radius, 2 * pi * radius   # 可以同时给两个变量赋值！
```

注意两点:

1. 这里等号的本质是赋值操作，最好念做“把 radius 赋值为 10”。判断两个数是否相等用的是==符号，如 `a == b`。
2. 命名尽量能体现出变量的信息！尽量用英语单词，如果要使用多个英语单词，单词之间可以加下划线连接。这样在代码长度变长、debug 回看时能够清晰知道变量表达的含义。

不太好的命名

1. l, I, 1 之间实在是太难区分了……所以尽量不要用！
2. 和内置函数或者自己曾经定义过的变量或函数冲突的名字不要用！否则就会出现下面这种情况

```
>>> pi = 0                          # pi 和内置的 pi 冲突了！
>>> radius = 10
>>> area = pi * radius * radius
>>> area
0
```

名字之间也是可以互相赋值的。

```
>>> f = max          # 甚至内置函数之间也可以互相赋值！现在 f 就代表 max 这个函数了
>>> f(1, 2)
2
>>> max = 3          # 把 max 赋值成数了
>>> max(1, 2)        # max 此后就不再代表函数 max 了
Traceback: ... TypeError: int is not callable
```

`Challenging Task!` 请问下面一段 python 代码执行之后结果是什么？答案在 1-5 之间。

```
f = min
f = max
g, h = min, max
max = g          # 小心!!这个命名真的很危险!!
print(max(f(2, g(h(1, 5), 3)), 4))
```

Python Tutor

pythontutor.com 是一个能够按步骤执行 python 的可视化平台，简单好用。点开一个 example 吧！

example: click here

你也可以将自己的代码拷贝到上面去运行。(注：新版 pythontutor 已经支持用 input 函数进行用户输入！) 在右边的变量框里面，可以查看各个变量的当前值。

在 pythontutor 执行 challenge task(click here!),给出上面 Challenging Task 的答案吧！

Hint: 对 *max* 赋值并没有改变内置函数的 `max` 本身，可以看到 f 和 h 仍然指向 `built-in function max`，而程序中的 g 和 *max* 则都指向 `built-in function min`。这个练习提醒我们：不要随意命名为 *max* 这类的变量！！

类似的变量还有 `len`, `str`, `int`, `float`, `list`, `tuple` (这些数据结构的名称最好都不要直接拿来命名变量)等等

```
>>> len = 5
>>> my_string = "Perhaps a bug may appear..."
>>> len(my_string)
Traceback: ... TypeError: int object is not callable  # 现在 len 已经不是函数了，它是个整数！不能被调用了。
```

Debugging

除了 pythontutor 可以用来 debug，pycharm 也有 debug 的方式。

首先点击任意一行的行号，会发现行号上出现了一个红色的圆点，这个圆点被称为断点。随后点击运行程序（绿色小三角）右边的“虫子”按钮，会发现程序暂停在断点处了！

更多关于 pycharm 调试的内容见【这个网站】。

Review Session

输入和输出

在命令行交互模式中，输入和输出都是很直接的。但是如果运行 python 文件 (python file)，则我们需要用特定的方式指定程序的输入入口。

python 的输出函数是 `print`。

```
>>> print("hello!")
hello!
>>> print(5)
5
>>> print(2 ** 10)      # 这是表达式，遵循表达式的求值原则。
1024
```

```
>>> print("2 ** 10")   # 这是字符串，不能直接求值！
2 ** 10
>>> a = 3
>>> b = 2
>>> print("a", "b")
a b
>>> print(a, b)
3 2
>>> print(a, b, sep="")
32
>>> print(a, b, sep="\t")  # \t 是制表符，\n 是换行符
3   2
>>> print(a, "Welcome", sep="\n")
a
Welcome
```

python 的输入函数是 `input`。`input` 函数返回的是字符串，一次会读取一行的输入。

```
def input(message_string=None):
    if message_string:
        print(message_string)   # 如果 input("提示信息")，那么程序会先打印出提示信息
    #... 等待你的输入
    return your_input_line_of_string
```

```
# a buggy code
a = input()
print(a * 2)          # 输入 2，为什么结果是 22？而不是 4？
```

```
22
```

基本数据类型

`int`, `float`, `bool`

基本程序结构

顺序， 分支， While 循环

- 分支样例（点击查看）
- 循环样例（点击查看）

程序结构的嵌套

序列 Sequence

“A sequence is an ordered collection of values."序列是一些值的有序排列。

常见的序列种类： `list`, `tuple`, `string`, `range`

尽管序列有很多种，但是他们有很多共同的属性。

- `len`: 序列的长度
- `[]`: 选择元素。
- `in`: 判断元素是否在 sequence 中
- `for`: `for item in sequence`, 遍历 sequence 中的元素进行循环

```
a = "123"
for i in a:
    print(i)
print(len(a))    # 输出 a 的长度
```

```
1
2
3
3
```

字符串的格式化

方式 1: 用 %

```
str1 = "%s, hello!" % ("Alex")        # %s 代表字符串
print(str1)
```

```
Alex, hello!
```

```
str2 = "Year: %d" % (2024)            # %d 代表整数
print(str2)
```

```
Year: 2024
```

```
str3 = "1 / 3 = %f" % (1 / 3)         # %f 代表浮点数
print(str3)
```

```
'1 / 3 = 0.333333'
```

```
str4 = "1 / 3 = %.2f" % (1 / 3)       # %.nf 代表保留 n 位小数的浮点数
print(str4)
```

```
1 / 3 = 0.33
```

方法 2: 用 format 或者 f

```
print("ab{}cd{}".format(0, 2))
```

```
ab0cd2
```

```
print("ab{:.2f}cd{}".format(0,2))
```

```
ab0.00cd2
```

```
name = "calvin"
print(f"welcome, {name}")
print("welcome, {}".format(name))
```

```
welcome, calvin
welcome, calvin
```

序列的变长

`list`, `tuple`, `string` 都支持加法和乘法运算来做数据合并。

题目实战

解题的标准步骤

- 想出解题思路
- 翻译成代码语言

注:

1. 审题要仔细，不要遗漏题干中的关键信息，尤其是特殊条件
2. 一定要先有大致思路再去写代码！

游戏时间（2023 某班期末机考 No.2）

(我 5:20 睡觉，13:14 起床打游戏 x)

描述

李华进入 P 大后，坚持每天打游戏，娱乐身心。请帮他计算某天的游戏时间。

读取四个整数 A,B,C,D，用来表示游戏的开始时间和结束时间。$0 \le A,C \le 23$, $0 \le B,D \le 59$。

其中 A 和 B 为 开始时刻的小时和分钟数，C 和 D 为结束时刻的小时和分钟数。

游戏可以在一天开始并在第二天结束。游戏最短持续 1 分钟，最长持续 24 小时。如果 A 与 C 相等，B 与 D 相等，则视为持续了 24 小时。

输入
共一行，包含四个整数 A,B,C,D。

输出
输出格式为 GAME TIME X HOUR(S) Y MINUTE(S)，表示游戏共持续了 X 小时 Y 分钟。

样例输入
7 8 9 10

样例输出
GAME TIME 2 HOUR(S) 2 MINUTE(S)

```
# 答案待更新
```

与 7 无关的数
(你相信这道题用一行代码就可以解决吗!)

描述
一个正整数,如果它能被 7 整除,或者它的十进制表示法中某一位上的数字为 7,则称其为与 7 相关的数.现求所有小于等于 n(n < 100)的与 7 无关的正整数的平方和.

输入
输入为一行,正整数 n(n < 100)

输出
输出一行，包含一个整数，即小于等于 n 的所有与 7 无关的正整数的平方和。

样例输入

```
21
```

样例输出

```
2336
```

```
# A BUGGY version
list=[]
for i in range(1,int(input())+1):
    list.append(i)

for number in list:
    if int(number)%7==0:
        list.remove(number)
    elif "7" in str(number):
```

```
        list.remove(number)

sum=0
for num in list:
    sum+=num**2
print(sum)
```

上面的代码里有一个 bug，你能找出来吗？

```
# 答案待更新
```

处理答题数据集（2023 某班期中机考 No.4）

(学计概 C 真的是有用的！)

描述

Y 同学在学习科研中经常需要处理一些数据文件。自从在学习了计算概论 C 之后，他一直希望使用 Python 提高自己数据集处理的效率，但由于水平有限，所以需要你的帮助。这次，他收集了一批考试答题的数据，希望将其整理成更容易阅读的格式，方便审阅。

原格式题面、答案、回答都分别储存，为了更方便的阅读对照，打算把每行处理为每个回答的“题面"的格式。具体输入输出见下面的要求。

输入

1. 第一行是两个数字 N 和 M，N 表示考题数目，M 表示考生数目。以空格间隔。
2. 接下来 N 行，每行为一个句子，表示 N 道题目的题面。
3. 再往后 N 行，每行为一个句子，表示 N 道题目的答案。
4. 接下来一共 N*M 行，每 M 行为一组。第 i 组（i=1，2，……，N）依次有序表示 M 个学生对第 i 题回答，每个回答是一个句子。每组 M 个回答分别表示 1，2，……，M 号考生的回答，保持相同的顺序。（假设题面、答案、回答都保持正确顺序）

输出

1. 输入有 N*M 行，N 组的顺序是题面输入的顺序（其实也是答案输入的顺序），组内 M 行，为作答考生回答的顺序（即为每道题 1，2，……，M 号的回答）。
2. 每行由 3 部分组成，第一部分为题面，第二部分为考生回答，第三部分为答案。三部分以 \t 间隔。注意：每组内的题面和答案是相同的。

样例输入

```
3 2
How are you doing?
What is your favorite sport?
Do you know 1+1=?
```

```
Great.
My favorite sport is tennis.
1+1=2
Not bad.
Good.
Sorry?
I like swimming.
1+1=3.
The answer is 2.
```

样例输出

```
How are you doing?    Not bad.    Great.
How are you doing?    Good.    Great.
What is your favorite sport?    Sorry?    My favorite sport is tennis.
What is your favorite sport?    I like swimming.    My favorite sport is tennis.
Do you know 1+1=?    1+1=3.    1+1=2
Do you know 1+1=?    The answer is 2.    1+1=2
```

```
# 答案待更新
```

选课意愿点（2023 某班期中机考 No.5）

（我的课为什么哗哗掉啊）

描述

“计算概论 Plus"吸引了大量同学的选课申请，系统收到了众多的选课申请和投点信息，这些信息的格式通常为“学号，意愿点”。

“计算概论 Plus"主要面向高年级学生，但低年级学生也可选课。课程教师计划开发一套筛选系统，根据申请选课的学生的年级和意愿点来决定选课名单。

作为筛选系统的开发者，你将接收到大量格式为“学号,意愿点”的字符串（例如“2100017333,99”，意愿点范围为 0 ~ 99），这些信息将逐行输入。课程教师设定的选课规则如下:

给每个同学累计一个选课分数值，初始为 0。* 先看该同学的年级。若是大四及更高年级的同学（学号显示的年级为 20 级及以上的），则分数值加 75，大三同学（21 级）分数值加 50，大二同学（22 级）分数值加 25，大一同学分数值加 15（最低为 23 级今年大一的同学）。* 再看意愿点。意愿点范围为 0 ~ 99，若所投意愿点为 x，则分数值加 x。* 综合以上三条规则，得到一个最终的分数值（分数值等于年级对应的加分和所投意愿点的总和）。

将分数值从高到低排序，进行选课抽签，分数值越高的学生被选中的概率越大。

一位预选了这门课的朋友找到你，希望你能帮他查看有多少人的分数值比他高。（朋友提供了他的学号，希望你能查看有多少人的分数值比他高。）

输入
n+2 行输入。

其中第 1 行为朋友的学号，想让你查询下有多少人分数值比 TA 高。

后面接着 n 行输入为预选这门课的人的学号和意愿点，格式为: “2100017333,99”，仅用逗号隔开，保证符合格式，保证无重复，一定包括朋友自己的学号。

最后一行是一个字符串”end”，代表输入完毕。

输出
一个非负整数 m，表示有多少个人的选课分数值比这位朋友的分数值高。

注: 若他是分数最高者则输出 0.

样例输入

```
2200019362
2100017333,99
2100015332,0
2200013400,47
2000013600,0
1900012435,12
2300017623,88
2300015333,20
2200019362,74
end
```

样例输出

```
2
```

```
# 答案待更新
```

数字密码
（这道题略微偏难……一位计概 B 的同学问我的，对计概 C 不超纲，但比较考验思维）

描述
小明和同学一起去玩密室逃脱，这个密室的老板特别喜欢数学，所以他设置了一个密码门。在搜集了房间里的有价值信息后，小明总结出的信息为:三个互不相同的正整数和为 231，那么这三个互不相同正整数的最大公因数就是密码。小明很快的算出了密码。

回到学校后，小明希望写一个程序，对于给定的三个互不相同的正整数的和，快速求出这三个正整数的最大公因数。

输入
输入为三个互不相同的正整数的和(6 ≤ N ≤ 10^9).

输出
输出三个数的最大公因数.

```
# 答案待更新
```