

Assignment #2: 语法练习

Updated 0126 GMT+8 Sep 24, 2024

2024 fall, Compiled by ==同学的姓名、院系==

姓名：李彦臻

学号：2300010821

学院：数学科学学院

****说明：****

1) 请把每个题目解题思路（可选），源码 Python，或者 C++（已经在 Codeforces/Openjudge 上 AC），截图（包含 Accepted），填写到下面作业模版中（推荐使用 typora <https://typoraio.cn>，或者用 word）。AC 或者没有 AC，都请标上每个题目大致花费时间。

3) 课程网站是 Canvas 平台，<https://pku.instructure.com>，学校通知 9 月 19 日导入选课名单后启用。****作业写好后，保留在自己手中，待 9 月 20 日提交。****

提交时候先提交 pdf 文件，再把 md 或者 doc 文件上传到右侧“作业评论”。Canvas 需要有同学清晰头像、提交文件有 pdf、“作业评论”区有上传的 md 或者 doc 附件。

4) 如果不能在截止前提交作业，请写明原因。

1. 题目

263A. Beautiful Matrix

<https://codeforces.com/problemset/problem/263/A>

思路：先建立一个列表 `matrix`，其内部包含五个小列表，每个列表包含五个数字；随后对每个列表求和，从哪个列表内部的数之和不为 0 便可看出 1 在哪一行；接着再对五个小列表中相同位置的数求和，哪个位置求和不为 0 便可看出 1 在哪一列；确定了 1 在哪一行哪一列就容易算出最少要移动多少次了~

代码

```
```python
#
```

```

'''
matrix=[]#建立列表matrix，里面存储了五个列表，每个列表代表一个行

for i in range(0,5):
 line=list(map(int,input().split()))#不要将列表的名字命名为line_i!!!
 #（电脑不会把i视为数字，会视为一个统一的名字"line_i"）
 #（直接让第一行i在（0，5）内即可自动执行五次~）
 matrix.append(line)

for j in range(0,5):
 if sum(matrix[j]) != 0:#对第j个列表求和~
 line=j#注意如果要printj，不要把print的步骤放入第一个for循环中，否则还没
 输入完就已经开始输出了！
 #（将line记为j，这样就知道哪一行有1了~）

for k in range(0,5):
 if sum(matrix[i][k] for i in range (0,5)) != 0:#连续索引：对k列求和=对每个列
 表的第k的元素求和！！
 colume=k#将有1的这一列记为k

#（此时j为1所在的行，k为1所在的列，注意行和列均从0开始算！）

if line < 2:
 line_step=2-line
else:
 line_step=line-2

if colume < 2:
 colume_step=2-colume
else:
 colume_step=colume-2

print(line_step+colume_step)

```

代码运行截图 ==（至少包含有"Accepted"）==

[HOME](#) [TOP](#) [CATALOG](#) [CONTESTS](#) [GYM](#) [PROBLEMSET](#) [GROUPS](#) [RATING](#) [EDU](#) [API](#) [CALENDAR](#) [HELP](#)

Please read [the new rule regarding the restriction on the use of AI tools](#).

[PROBLEMS](#) [SUBMIT CODE](#) [MY SUBMISSIONS](#) [STATUS](#) [HACKS](#) [ROOM](#) [STANDINGS](#) [CUSTOM INVOCATION](#)

General

#	Author	Problem	Lang	Verdict	Time	Memory	Sent	Judged		
283127562	Practice: Jo1423	<a href="#">263A</a> - 8	Python 3	Accepted	184 ms	32 KB	2024-09-27 08:10:23	2024-09-27 08:10:23	☆	<a href="#">Compare</a>

→ Source

Copy

```
matrix=[]#建立列表matrix, 里面存储了五个列表, 每个列表代表一行
for i in range(0,5):
 line=list(map(int,input().split()))#不要将列表的名字命名为line_1!!!
 #<电脑不会把1视为数字, 会视为一个统一的名字"line_1">
 #<直接让第一行在(0, 5)内即可自动执行五次>
 matrix.append(line)
for j in range(0,5):
 if sum(matrix[i]) != 0:#对第j个列表求和
 line=print(j, sum(matrix[i]))#将line记为j, 这样就知道哪一行有1了
 else:
 line=""
for k in range(0,5):
 if sum(matrix[i][k] for i in range(0,5)) != 0:#连续索引: 对k列求和=对每个列表的第k的元素求和!
 column=k#将有1的这一列记为k
 else:
 column=""
#(此时j为1所在的行, k为1所在的列, 注意行和列均从0开始算!)
if line < 2:
 line_step=2-line
else:
 line_step=line-2
if column < 2:
 column_step=2-column
else:
 column_step=column-2
print(line_step+column_step)
```

Click to see test details

### 1328A. Divisibility Problem

<https://codeforces.com/problemset/problem/1328/A>

思路：本质上就是求每一行的带余除法中的余数，为此使用//符号算出出的结果，再拿被除数减去除数乘以结果便能得到余数

##### 代码

```
```python
#
```
```

```
number_of_test=int(input())
```

result=[]#如果需要对每个项目延迟输出结论，则可以先把结论储存在一个列表里

```
for i in range(number_of_test):#会自动验证 n_o_t 次~
 test=list(map(int,input().split()))
 a=test[0]
```

```

b=test[1]

if a % b == 0:
 result.append(0)#将 0 储存在列表里

else:
 number=b - (a % b)
 result.append(number)#将 number 储存在列表里

for j in range(number_of_test):
 print(result[j])#最后再挨个儿把储存好的结果打印出来~

```

代码运行截图 ==（至少包含有“Accepted”）==

The screenshot shows the Codeforces website interface. At the top, there's a navigation bar with links like HOME, TOP, CATALOG, CONTESTS, GYM, etc. Below this is a banner for a new rule regarding AI tool usage. The main content area shows a submission table with columns: #, Author, Problem, Lang, Verdict, Time, Memory, Sent, Judged. The submission #283129555 by user Jo1423 for problem 1328A - 12 in Python 3 is shown with a green 'Accepted' verdict. Below the table, there's a 'Source' tab showing the submitted code, which matches the code provided in the previous block. A 'Click to see test details' link is also visible.

### 427A. Police Recruits

<https://codeforces.com/problemset/problem/427/A>

思路：第一次提交的代码是简答的暴力循环法，可以解决问题但时间复杂度过高（是  $on^2$ ），因为要采用一种新的方法，即“变化的量”法。具体而言，就是设置两个变化的量，一个是 free police，一个是 untreated crimes；每招募一个警察，free 警察数就加一（随着 input 动态变化）；反过来每发生一次事件根据当下 free 警察的数量来判断是 free 警察减一个还是 untreated crimes 增加一个即可~

##### 代码

```
```python
#
```
```

#上个代码虽然对，但复杂度过高，在处理几千组数据的时候需要验证的东西太多

#因此，接下来给出一个全新的方法，即“变化的量”法：

amount=int(input())#p. s. 这个数据没用

events=list(map(int,input().split()))

free\_police=0#将有空的警察储存为一个随时变化的量（随时调用！）

untreated\_crimes=0#只储存那些永远也无法处理的案件（每当一个案件废了，垃圾堆+1）

#注意：不要把历史上的所有案件都储存起来，因为把那些永远无法处理的案件存进去了没有意义！

```
for i in range(len(events)):
```

```
 if events[i] < 0:#如果发生了一起案件
```

```
 if free_police > 0:
```

```
 free_police -= 1#一位警察殉职
```

```
 else:
```

```
 untreated_crimes += 1#多了一个废掉的案件，扔进垃圾堆！
```

```
 else:#如果招进来了一些警察
```

```
 free_police += events[i]
```

```
print(untreated_crimes)#到了最后直接把所有没处理的案件数打印出来就行啦~
```

代码运行截图 ==（AC 代码截图，至少包含有“Accepted”）==

Please read [the new rule regarding the restriction on the use of AI tools](#).

PROBLEMS SUBMIT CODE MY SUBMISSIONS **STATUS** HACKS ROOM STANDINGS CUSTOM INVOCATION

| General   |                  |                           |          |          |        |         |                     |                     |   |         |
|-----------|------------------|---------------------------|----------|----------|--------|---------|---------------------|---------------------|---|---------|
| #         | Author           | Problem                   | Lang     | Verdict  | Time   | Memory  | Sent                | Judged              |   |         |
| 283365899 | Practice: Jo1423 | <a href="#">427A</a> - 21 | Python 3 | Accepted | 108 ms | 8912 KB | 2024-09-28 14:32:10 | 2024-09-28 14:32:10 | ★ | Compare |

→ Source Copy

```
#上个代码虽然对，但复杂度过高，在处理几千组数据的时候需要验证的东西太多
#因此，接下来给出一个全新的方法，即“变化的量”法：
amount=int(input())#p.s.这个数据没用
events=list(map(int,input().split()))

free_police=0##将空闲的警察储存为一个随时变化的量（随时调用！）
untreated_crimes=0##只储存那些永远也无法处理的案件（每当一个案件废了，垃圾堆+1）
#注意：不要把历史上的所有案件都储存起来，因为把那些永远无法处理的案件存进去了没有意义！

for i in range(len(events)):
 if events[i] < 0:##如果发生了一起案件
 if free_police > 0:
 free_police -= 1##一位警察殉职
 else:
 untreated_crimes += 1##多了一个废掉的案件，扔进垃圾堆！
 else:##如果招进来了一个警察
 free_police += events[i]

print(untreated_crimes)##到了最后直接把所有没处理的案件数打印出来就行啦~
```

[Click to see test details](#)

### 02808: 校门外的树

<http://cs101.openjudge.cn/practice/02808/>

思路：先构建一个长度为 L+1，元素全部为 true 的列表，随后把在地铁修建范围内的“树”（实则为 True）全部修改为 False，最后再数列表里有多少个 True 即可~

P. S. 不要把列表里的元素一个一个删去，这样会大大增加时间复杂度！（这也就是为什么要把元素全部设为 True 而不是一个个不同的数字，因为这样可以把一整个范围内的不要的树直接修改为 False，不需要一个一个删掉）

##### 代码

```
```python
#
```
```

#现代电脑能处理的指令数大概为  $10^9$  量级，因此 n 大于  $10^5$  时最好不超过  $O(n \log n)$

```
number=list(map(int,input().split()))
L=number[0]
M=number[1]
```



思路：对区间 a 到 b 内的所有数进行验证即可：先用//100 来算出百位数，接着对上一次操作的余数//10 来算出十位数，最后得到的余数也就是个位数了。接下来，只需要验证这个数是否等于百位数的三次方+十位数的三次方+个位数的三次方即可~

##### 代码

```
```python
#
...

daffodils=[]

numbers=list(map(int,input().split()))
a=numbers[0]
b=numbers[1]

for i in range(a,b + 1):
    x=i // 100
    y=(i - 100 * x) // 10
    z=i - 100 * x - 10 * y

    if i == x**3 + y**3 + z**3:
        daffodils.append(i)

if len(daffodils) == 0:
    print("NO")
else:
    print(" ".join(map(str,daffodils)))
```

代码运行截图 == (AC 代码截图，至少包含有"Accepted") ==

晴问 课程 训练营 算法笔记 题库 比赛 语言入门教程 考研算法大题特训 New 本期速递

【从零开始的 C++ 课程】现已免费发布: <https://sunnywhy.com/course/1880>, 【C++多线程课程】现已免费发布: <https://sunnywhy.com/course/1880/model/1884?itemid=1675>

入门篇 (1) ——入门模拟

- 简单模拟
 - 3N+1猜想
 - 判断三角形
 - 单调递增序列
 - 数列奇数和
 - 三位数
 - 水仙花数
 - 水仙花数II
 - 2的幂

题目 题解

水仙花数 = $1^3 + 0^3 + 0^3$, 因此153是水仙花数。

给定两个正整数a、b, 输出在闭区间[a,b]内的所有水仙花数。

输入描述

两个正整数a、b ($100 \leq a \leq b \leq 999$)。

输出描述

在一行里输出闭区间[a,b]内的所有水仙花数, 多个水仙花数按从小到大的顺序输出, 中间用空格隔开, 行末不允许有多余的空格。如果区间内没有水仙花数, 那么输出NO。

样例1

输入 复制

360 380

输出 复制

370 371

样例2

代码书写 Python

```
1 daffodils=[]
2
3 numbers=list(map(int,input().split()))
4 a=numbers[0]
5 b=numbers[1]
6
7 for i in range(a,b+1):
8     x=i//100
9     y=(i-100*x)//10
10    z=i-100*x-10*y
11
12    if i==x**3+y**3+z**3:
13        daffodils.append(i)
14
15 if len(daffodils)==0:
16     print("NO")
17 else:
18     print(" ".join(map(str,daffodils)))
```

测试输入 提交结果 历史提交

完美通过 100% 数据通过测试 运行时长: 0 ms 查看题解

收起面板 运行 提交

01922: Ride to School

<http://cs101.openjudge.cn/practice/01922/>

思路: 出发的人有以下四种可能,

1. 在 0 时刻之前出发并且在所有 0 时刻及之后出发的人之前到学校, 这样 charley 永远也追不上他;
2. 在 0 时刻之前出发但是被某个 0 时刻及之后出发的人在路上被超过了, 这样 charley 也永远不会坐上他的车 (因为是被超过的一方);
3. 在 0 时刻之后出发但是被某个 0 时刻及之后出发的人在路上被超过了, 这样 charley 就算某一次坐上去了也迟早会离开他;
4. 在 0 时刻之后出发并且在所有 0 时刻及之后出发的人之前到学校, 这样 charley 迟早会坐上他并且永远不会离开!

因此, charley 到达学校的时间就是所有 0 时刻及之后出发的人里最快到达学校的时间! 所以只需要计算出来每个 0 时刻及之后出发的人到达学校的时间, 并取其最小值即可~

代码

```
```python
#
```
```

#出发的人分为四种情况:

#第一种, 在 0 时刻之前出发并且在所有 0 时刻及之后出发的人之前到学校, 这样 charley 永远也追不上他;

#第二种, 在 0 时刻之前出发但是被某个 0 时刻及之后出发的人在路上被超过了, 这样 charley 也永远不会坐上他的车 (因为是被超过的一方);

#第三种, 在 0 时刻之后出发但是被某个 0 时刻及之后出发的人在路上被超过了, 这样 charley 就算某一次坐上去了也迟早会离开他;

#第四种, 在 0 时刻之后出发并且在所有 0 时刻及之后出发的人之前到学校, 这样 charley 迟早会坐上他并且永远不会离开!

#因此得出结论: charley 到达学校的时间就是所有 0 时刻及之后出发的人里最快到达学校的时间!

#所以只需要计算出来每个 0 时刻及之后出发的人到达学校的时间, 并取其最小值即可~

```
result=[]
```

```
def ride_function(number):#函数后面一定要有括号! 括号内可以没有东西, 这样就会直接进入下一步
```

```
    current_result=[]
```

```
    for i in range(number):
```

```
        info=list(map(int,input().split()))
```

```
        speed=info[0]
```

```
        start_time=info[1]
```

```
        if start_time >= 0:#别忘了验证出发时间是否大于等于 0~如果为否则可直接排除!
```

```
            overall_time=(16200/speed) + start_time
```

```
            current_result.append(overall_time)
```

```
    a=min(current_result)
```

```
    if a % 1 == 0:#判断 a 是否为整数的最佳方法实际上为 isinstance(a,int); 如果用 a%1==0 的方法, 1.0 就会算进去
```

```
        # (但这道题还是需要把 780.0 算进去, 所以还是用%方法~)
```

```
        result.append(int(a))
```

```
    else:
```

```
        result.append(int(a) + 1)
```

```
while True:
```

```
    number=int(input())#一定要先命名 number, 不要 if(int(input))==0! 这样会不知道原本输入了什么!!
```

```
    if number != 0:
```

```
        ride_function(number)#先命名 number, 这样一来直接对 number 进行判断即可; 如果不为 0 则进入函数程序, 不断往 result 里加数
```

```
    else:
```

```
        break#如果为 0 则整个程序结束, 进入输出模式~
```

```
for b in result:
```

```
    print(b)
```

代码运行截图 ==（AC 代码截图，至少包含有“Accepted”）==


OpenJudge

题目ID, 标题, 描述

23n2300010821

信箱

账号

 CS101 / 题库（包括计概、数算题目）

题目

排名

状态

提问

#46321057提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
#出发的人分为四种情况:
#第一种, 在0时刻之前出发并且在所有0时刻及之后出发的人之前到学校, 这样charley永远也追
#第二种, 在0时刻之前出发但是被某个0时刻及之后出发的人在路上被超过了, 这样charley也永
#第三种, 在0时刻之后出发但是被某个0时刻及之后出发的人在路上被超过了, 这样charley就算
#第四种, 在0时刻之后出发并且在所有0时刻及之后出发的人之前到学校, 这样charley迟早会坐
#因此得出结论: charley到达学校的时间就是所有0时刻及之后出发的人里最快到达学校的时间!
#所以只需要计算出来每个0时刻及之后出发的人到达学校的时间, 并取其最小值即可~

result=[]

def ride_function(number): #函数后面一定要有括号! 括号内可以没有东西, 这样就会直接报错
    current_result=[]

    for i in range(number):
        info=list(map(int,input().split()))
        speed=info[0]
        start_time=info[1]

        if start_time >= 0: #别忘了验证出发时间是否大于等于0~如果为否则可直接排除
            overall_time=(16200/speed) + start_time
            current_result.append(overall_time)

    a=min(current_result)
    if a % 1 == 0: #判断a是否为整数的最佳方法实际上为 isinstance(a,int); 如果用a
        # (但这道题还是需要把780.0算进去, 所以还是用%方法~)
```

基本信息

#: 46321057
题目: 01922
提交人: 23n2300010821
内存: 3796kB
时间: 42ms
语言: Python3
提交时间: 2024-10-05 21:29:10

2. 学习总结和收获

==如果作业题目简单, 有否额外练习题目, 比如: OJ“计概 2024fall 每日选做”、CF、LeetCode、洛谷等网站题目。==

经过最近一段时间的练习（每天至少做两道每日习题），我感觉我对基础的语法结构已经逐渐熟悉了起来，很少像刚开始学的时候经常因为语法不够熟悉而得回去翻书查看每个语法结构该怎么使用了，或者犯一些很简单的语法错误。随着我对基础语法的熟悉，我做难度 900 的题目基本也没太大问题了，但再往上一点难度的题目就经常出现 time limit exceeded 的问题，即时间复杂度太高，我觉得这归根结底还是因为对算法不够熟悉，还没有进入计算机思维；我个人认为这段瓶颈期需要做更多的题目来解决。另外，在学习编程的过程中，我也越来越发现善用 ai 的重要性，因为 gpt 可以大幅提高知识的摄入效率，并且 ai 在大部分情况下也能精确的指出代码中的错误或者可以优化的地方，这对我 python 的学习很有帮助~