

1. 题目

E27653: Fraction类

<http://cs101.openjudge.cn/pctbook/E27653/>

请练习用OOP方式实现。

思路：使用math包里的最大公约数gcd(第一次写OOP感觉不是很顺手，希望考试没有这个要求)

代码：

```
import math
class fraction:
    def __init__(self,s):
        self.s=s
    def solve(self):
        a=s[1]*s[3]
        b=s[0]*s[3]+s[1]*s[2]
        c=math.gcd(a,b)
        return(f'{int(b/c)}/{int(a/c)}')
if __name__ == "__main__":
    s=list(map(int,input().split()))
    checker = fraction(s)
    print(checker.solve())
```

代码运行截图 (至少包含有"Accepted")

#49951193提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
import math
class fraction:
    def __init__(self,s):
        self.s=s
    def solve(self):
        a=s[1]*s[3]
        b=s[0]*s[3]+s[1]*s[2]
        c=math.gcd(a,b)
        return(f'{int(b/c)}/{int(a/c)}')
if __name__ == "__main__":
    s=list(map(int,input().split()))
    checker = fraction(s)
    print(checker.solve())
```

基本信息

#: 49951193
题目: E27653
提交人: 2400011474
内存: 3624kB
时间: 20ms
语言: Python3
提交时间: 2025-09-09 12:18:22

binary search, <https://leetcode.cn/problems/minimum-limit-of-balls-in-a-bag/>

思路：直接二分，第一次见这个方法还是牛跳石头那个题，当时第一次见完全懵逼了，现在就比较得心应手了

代码：

```
class Solution:
    def minimumSize(self, nums: List[int], maxOperations: int) -> int:
        def solve(y):
            ans=0
            for x in nums:
                ans+=-(-x//y)-1
            if ans<=maxOperations:
                return 1
            return 0
        l=0
        r=max(nums)
        while r-l>1:
            if solve((l+r)//2):
                r=(l+r)//2
            else:
                l=(l+r)//2
        return(r)
```

代码运行截图 (至少包含有"Accepted")



```
1 class Solution:
2     def minimumSize(self, nums: List[int], maxOperations: int) -> int:
3         def solve(y):
4             ans=0
5             for x in nums:
6                 ans+=-(-x//y)-1
7             if ans<=maxOperations:
8                 return 1
9             return 0
10        l=0
11        r=max(nums)
12        while r-l>1:
13            if solve((l+r)//2):
14                r=(l+r)//2
15            else:
16                l=(l+r)//2
17        return(r)
18
```

M04135: 月度开销

binary search, <http://cs101.openjudge.cn/pctbook/M04135/>

思路：all the same

代码：

```
n,m=map(int,input().split())
l=[ ]
for _ in range(n):
    l.append(int(input()))
def solve(y):
    ans=0
    s=0
    for x in l:
        s+=x
        if s>y:
            ans+=1
            s=x
    ans+=1
    if ans>m:
        return 0
    return 1
e=max(l)-1
r=sum(l)
while r-e>1:
    f=(e+r)//2
    if solve(f):
        r=f
    else:
        e=f
print(r)
```

代码运行截图 (至少包含有"Accepted")

状态: Accepted

源代码

```

n,m=map(int,input().split())
l=[]
for _ in range(n):
    l.append(int(input()))
def solve(y):
    ans=0
    s=0
    for x in l:
        s+=x
        if s>y:
            ans+=1
            s=x
    ans+=1
    if ans>m:
        return 0
    return 1
e=max(l)-1
r=sum(l)
while r-e>1:
    f=(e+r)//2
    if solve(f):
        r=f
    else:
        e=f
print(r)

```

基本信息

#: 49951382
 题目: M04135
 提交人: 2400011474
 内存: 7456kB
 时间: 317ms
 语言: Python3
 提交时间: 2025-09-09 13:44:10

M27300: 模型整理

sortings, AI, <http://cs101.openjudge.cn/pctbook/M27300/>

思路: 用字典

代码:

```

n=int(input())
d={}
for _ in range(n):
    l=input().split("-")
    p=float(l[1][:-1])*[1e6,1e9][l[1][-1]=="B"]
    if l[0] not in d:
        d[l[0]]=[]
    d[l[0]].append([p,l[1]])
y=sorted(d.keys())
for key in y:
    x=sorted(d[key])
    print(key+": "+", ".join(y[1] for y in x))

```

代码运行截图 (至少包含有"Accepted")

#49953680 提交状态

查看

提交

统计

提问

状态: Accepted

基本信息

#: 49953680

题目: M27300

提交人: 2400011474

内存: 3656kB

时间: 23ms

语言: Python3

提交时间: 2025-09-09 21:44:56

源代码

```
n=int(input())
d={}
for _ in range(n):
    l=input().split("-")
    p=float(l[1][-1])*[1e6,1e9][l[1][-1]== "B"]
    if l[0] not in d:
        d[l[0]]=[]
    d[l[0]].append([p,l[1]])
y=sorted(d.keys())
for key in y:
    x=sorted(d[key])
    print(key+": "+", ".join(y[1] for y in x))
```

©2002-2022 POJ 京ICP备20010980号-1

English 帮助 关于

Q5. 熟悉云虚拟机Linux环境与大语言模型（LLM）本地部署

本项目包括两个任务：

- 1) 通过云虚拟机（如 <https://clab.pku.edu.cn/> 提供的资源）熟悉Linux系统操作环境。
- 2) 完成大语言模型（LLM）的本地部署与功能测试。

LLM 部署可选择使用图形化工具（如 LM Studio, <https://lmstudio.ai>）以简化配置流程，提升部署效率。部署完成后，需对模型进行实际能力测试。

测试内容包括：从主流在线编程评测平台（如 OpenJudge、Codeforces、LeetCode 或洛谷等）选取若干编程题目，提交由本地部署的 LLM 生成的代码解决方案，并确保其能够通过全部测试用例，获得“Accepted”状态。选题时应避免与已知可被 AI 正确解答的题目重复。当前已确认可通过的 AI 解题列表可参考以下 GitHub 仓库：

https://github.com/GMyhf/2025spring-cs201/blob/main/AI_accepted_locally.md

请提供你的项目进展，内容应该包括：关键操作步骤的截图以及遇到的技术问题及相应的解决方法。这将有助于全面掌握项目推进情况，并为后续优化与扩展提供依据。

最近比较忙，这个题可能就先跳了（

Q6. 阅读《Build a Large Language Model (From Scratch)》第一章

作者：Sebastian Raschka

请整理你的学习笔记。这应该包括但不限于对第一章核心概念的理解、关键术语的解析、学习过程中的思考与启发，以及尚存的疑问与反思。通过系统梳理，不仅有助于巩固自身理解，也希望为其他学习者提供有价值的参考。

1.1 什么是大型语言模型（LLM）？

- LLM 是一种基于深度神经网络的模型，能够理解、生成和回应类人文本。
- “大型”体现在两个方面：参数规模大（数十亿至数千亿）、训练数据规模大（涵盖互联网大量文本）。
- 核心训练任务：预测下一个词（next-token prediction），利用语言序列性学习上下文关系。

1.2 LLM 的应用场景

- 文本生成（如写诗、代码、文章）
- 机器翻译、情感分析、文本摘要
- 聊天机器人（如 ChatGPT）、搜索引擎增强
- 专业领域知识检索（如医疗、法律）

1.3 LLM 的构建与使用阶段

1. 预训练（Pretraining）：
 - 使用大规模无标注文本进行自监督学习（下一词预测）。
 - 输出为基础模型（foundation model），具备文本补全和少量示例学习能力。
2. 微调（Fine-tuning）：
 - 在标注数据上进一步训练，适应特定任务（如分类、指令遵循）。
 - 分为指令微调（instruction tuning）和分类微调（classification fine-tuning）。

1.4 Transformer 架构简介

- 源于2017年论文《Attention Is All You Need》，最初用于机器翻译。
- 包含编码器（Encoder）和解码器（Decoder）两部分。
- 自注意力机制（Self-Attention）：能捕捉长距离依赖关系，是Transformer的核心。
- 衍生模型：
 - BERT：基于编码器，擅长文本分类。
 - GPT：基于解码器，擅长生成任务。

1.5 数据集与训练成本

- GPT-3 训练数据包含 CommonCrawl、WebText、Books、Wikipedia 等，token 数量达数千亿。
- 预训练成本极高（例如GPT-3约需460万美元），因此常使用开源预训练模型进行微调。

1.6 GPT 架构特点

- 仅使用解码器，采用自回归方式生成文本。
- 表现出涌现能力（emergent behavior）：能执行未明确训练的任务（如翻译）。
- 模型规模巨大（GPT-3有1750亿参数），但仍基于相同的基本架构理念

2. 学习总结和个人收获

如果发现作业题目相对简单，有否寻找额外的练习题目，如“数算2025spring每日选做”、LeetCode、Codeforces、洛谷等网站上的题目。

每日选做一直在跟，大部分算法计概都学过（或者至少查一下马上能理解并实现，比如马拉车和并查集）比较顺手，直到碰到那道花了我一个晚上，写了3千多B代码（思路是问ai的，但代码完全是我自己实现的）才ac的求中位数的题