

Assignment #7: 20250402 Mock Exam

Updated 1624 GMT+8 Apr 2, 2025

2025 spring, Compiled by 袁奕 2400010766 数院

说明:

1. **月考:** AC5。考试题目都在“题库（包括计概、数算题目）”里面，按照数字题号能找到，可以重新提交。作业中提交自己最满意版本的代码和截图。
2. **解题与记录:**
对于每一个题目，请提供其解题思路（可选），并附上使用Python或C++编写的源代码（确保已在OpenJudge, Codeforces, LeetCode等平台上获得Accepted）。请将这些信息连同显示“Accepted”的截图一起填写到下方的作业模板中。（推荐使用Typora <https://typoraio.cn> 进行编辑，当然你也可以选择Word。）无论题目是否已通过，请标明每个题目大致花费的时间。
3. **提交安排:** 提交时，请首先上传PDF格式的文件，并将.md或.doc格式的文件作为附件上传至右侧的“作业评论”区。确保你的Canvas账户有一个清晰可见的头像，提交的文件为PDF格式，并且“作业评论”区包含上传的.md或.doc附件。
4. **延迟提交:** 如果你预计无法在截止日期前提交作业，请提前告知具体原因。这有助于我们了解情况并可能为你提供适当的延期或其他帮助。

请按照上述指导认真准备和提交作业，以保证顺利完成课程要求。

1. 题目

E05344:最后的最后

<http://cs101.openjudge.cn/practice/05344/>

```
1  from collections import deque
2
3  n, k = map(int, input().split())
4
5  que = deque(range(1, n + 1))
6  res = []
7
8  for _ in range(n - 1):
9      for _ in range(k - 1):
10         que.append(que.popleft())
11         res.append(que.popleft())
12
13  print(*res, sep = " ")
```

状态: Accepted

源代码

```
from collections import deque

n, k = map(int, input().split())

que = deque(range(1, n + 1))
res = []

for _ in range(n - 1):
    for _ in range(k - 1):
        que.append(que.popleft())
    res.append(que.popleft())

print(*res, sep = " ")
```

基本信息

#: 48802175
题目: 05344
提交人: 24n2400010766
内存: 3616kB
时间: 22ms
语言: Python3
提交时间: 2025-04-02 17:45:15

M02774: 木材加工

binary search, <http://cs101.openjudge.cn/practice/02774/>

```
1  n, k = map(int, input().split())
2  nums = []
3
4  for _ in range(n):
5      nums.append(int(input()))
6
7  def isvalid(l, k):
8      return sum(x // l for x in nums) >= k
9
10 def cut(k):
11     length = sum(x for x in nums)
12     if length < k:
13         return 0
14     left, right = 1, 10000
15     while left + 1 < right:
16         mid = (left + right) // 2
17         if isvalid(mid, k):
18             left = mid
19         else:
20             right = mid - 1
21     if left + 1 == right and isvalid(right, k):
22         left = right
23     return left
24
25 print(cut(k))
```

状态: Accepted

源代码

```
n, k = map(int, input().split())
nums = []

for _ in range(n):
    nums.append(int(input()))

def isValid(l, k):
    return sum(x // l for x in nums) >= k

def cut(k):
    length = sum(x for x in nums)
    if length < k:
        return 0
    left, right = 1, 10000
    while left + 1 < right:
        mid = (left + right) // 2
        if isValid(mid, k):
            left = mid
        else:
            right = mid - 1
    if left + 1 == right and isValid(right, k):
        left = right
    return left

print(cut(k))
```

基本信息

#: 48802242
题目: 02774
提交人: 24n2400010766
内存: 3928kB
时间: 38ms
语言: Python3
提交时间: 2025-04-02 17:47:41

M07161:森林的带度数层次序列存储

tree, <http://cs101.openjudge.cn/practice/07161/>

```
1 class TreeNode:
2     def __init__(self, val):
3         self.val = val
4         self.children = []
5
6     def build(Nodes_name, degrees):
7         root = TreeNode(Nodes_name[0])
8         curr_father_id = 0
9         Nodes = [root]
10        for i in range(1, len(degrees)):
11            node = TreeNode(Nodes_name[i])
12            curr_father = Nodes[curr_father_id]
13            while len(curr_father.children) == degrees[curr_father_id]:
14                curr_father_id += 1
15                curr_father = Nodes[curr_father_id]
16            curr_father.children.append(node)
17            Nodes.append(node)
18        return root
19
20    def postorder_traversal(root):
21        for child in root.children:
22            postorder_traversal(child)
23        res.append(root.val)
24
25    T = int(input())
26    res = []
27    for _ in range(T):
28        read = list(input().split())
29        n = len(read) // 2
```

```

30     Nodes_name = [read[2 * i] for i in range(n)]
31     degrees = [int(read[2 * i + 1]) for i in range(n)]
32     root = build(Nodes_name, degrees)
33     postorder_traversal(root)
34
35 print(" ".join(res))

```

#48803043提交状态

[查看](#) [提交](#) [统计](#) [提问](#)

状态: Accepted

源代码

```

class TreeNode:
    def __init__(self, val):
        self.val = val
        self.children = []

def build(Nodes_name, degrees):
    root = TreeNode(Nodes_name[0])
    curr_father_id = 0
    Nodes = [root]
    for i in range(1, len(degrees)):
        node = TreeNode(Nodes_name[i])
        curr_father = Nodes[curr_father_id]
        while len(curr_father.children) != degrees[curr_father_id]:
            curr_father_id += 1
            curr_father = Nodes[curr_father_id]
        curr_father.children.append(node)
        Nodes.append(node)
    return root

def postorder_traversal(root):
    for child in root.children:
        postorder_traversal(child)
    res.append(root.val)

T = int(input())
res = []
for _ in range(T):
    read = list(input().split())
    n = len(read) // 2
    Nodes_name = [read[2 * i] for i in range(n)]
    degrees = [int(read[2 * i + 1]) for i in range(n)]
    root = build(Nodes_name, degrees)
    postorder_traversal(root)

print(" ".join(res))

```

基本信息

#: 48803043
 题目: 07161
 提交人: 24n2400010766
 内存: 3660kB
 时间: 21ms
 语言: Python3
 提交时间: 2025-04-02 18:48:05

M18156:寻找离目标数最近的两数之和

two pointers, <http://cs101.openjudge.cn/practice/18156/>

```

1 t = int(input())
2 nums = sorted(list(map(int, input().split())))
3
4 left, right = 0, len(nums) - 1
5 s, dis = 1e9, 1e9
6
7 while left < right:
8     new_s = nums[left] + nums[right]
9     new_dis = abs(new_s - t)
10    if new_dis < dis:
11        dis = new_dis
12        s = new_s
13    if new_dis == dis:
14        s = min(s, new_s)

```

```

15     if new_s <= t:
16         left += 1
17     else:
18         right -= 1
19
20 print(s)

```

状态: Accepted

源代码

```

t = int(input())
nums = sorted(list(map(int, input().split())))

left, right = 0, len(nums) - 1
s, dis = 1e9, 1e9

while left < right:
    new_s = nums[left] + nums[right]
    new_dis = abs(new_s - t)
    if new_dis < dis:
        dis = new_dis
        s = new_s
    if new_dis == dis:
        s = min(s, new_s)
    if new_s <= t:
        left += 1
    else:
        right -= 1

print(s)

```

基本信息

#: 48802311
 题目: 18156
 提交人: 24n2400010766
 内存: 15248kB
 时间: 119ms
 语言: Python3
 提交时间: 2025-04-02 17:49:45

M18159:个位为 1 的质数个数

sieve, <http://cs101.openjudge.cn/practice/18159/>

注意:

```

1  res = [True] * 100000
2
3  for i in range(2, 100000):
4      if not res[i]:
5          continue
6      k = 2 * i
7      while k < 100000:
8          res[k] = False
9          k += i
10
11 n = int(input())
12
13 for i in range(1, 1 + n):
14     print(f"Case{i}:")
15     m = int(input())
16     ans = [i for i in range(11, m) if res[i] and i % 10 == 1]
17     if not ans:
18         print("NULL")
19     else:
20         print(*ans, sep = " ")

```

状态: Accepted

源代码

```
res = [True] * 100000

for i in range(2, 100000):
    if not res[i]:
        continue
    k = 2 * i
    while k < 100000:
        res[k] = False
        k += i

n = int(input())

for i in range(1, 1 + n):
    print(f"Case{i}:")
    m = int(input())
    ans = [i for i in range(11, m) if res[i] and i % 10 == 1]
    if not ans:
        print("NULL")
    else:
        print(*ans, sep = " ")
```

基本信息

#: 48802366
题目: 18159
提交人: 24n2400010766
内存: 12100kB
时间: 2683ms
语言: Python3
提交时间: 2025-04-02 17:51:40

M28127:北大夺冠

hash table, <http://cs101.openjudge.cn/practice/28127/>

```
1 class problem:
2     def __init__(self, name):
3         self.name = name
4         self.commit = 0
5         self.AC = set()
6     def __lt__(self, other):
7         if len(self.AC) != len(other.AC):
8             return len(self.AC) > len(other.AC)
9         if self.commit != other.commit:
10            return self.commit < other.commit
11        return self.name < other.name
12
13 m = int(input())
14 database = dict()
15
16 for _ in range(m):
17     name, prob, res = input().split(",")
18     if name not in database:
19         database[name] = problem(name)
20     name_class = database[name]
21     name_class.commit += 1
22     if res == "yes":
23         name_class.AC.add(prob)
24
25 nums = list(database.values())
26 nums = sorted(nums)
27
28 for i in range(1, min(13, len(nums) + 1)):
```

```
29         print(i, nums[i - 1].name, len(nums[i - 1].AC), nums[i - 1].commit, sep
            = " ")
```

状态: Accepted

源代码

```
class problem:
    def __init__(self, name):
        self.name = name
        self.commit = 0
        self.AC = set()
    def __lt__(self, other):
        if len(self.AC) != len(other.AC):
            return len(self.AC) > len(other.AC)
        if self.commit != other.commit:
            return self.commit < other.commit
        return self.name < other.name

m = int(input())
database = dict()

for _ in range(m):
    name, prob, res = input().split(",")
    if name not in database:
        database[name] = problem(name)
    name_class = database[name]
    name_class.commit += 1
    if res == "yes":
        name_class.AC.add(prob)

nums = list(database.values())
nums = sorted(nums)

for i in range(1, min(13, len(nums) + 1)):
    print(i, nums[i - 1].name, len(nums[i - 1].AC), nums[i - 1].commit,
```

基本信息

#: 48802519
题目: 28127
提交人: 24n2400010766
内存: 3668kB
时间: 22ms
语言: Python3
提交时间: 2025-04-02 17:56:33

2. 学习总结和收获



🔖 力扣最受刷题发烧友欢迎的 100 道题 [更多信息](#)

LeetCode 热题 100

24 / 100



🔖 力扣最受刷题发烧友欢迎的 100 道题 [更多信息](#)

LeetCode 热题 100

48 / 100

其中的双指针技巧在本次考试中用到了.

并有一个疑问, 其中

[189. 轮转数组 - 力扣 \(LeetCode\)](#)

为什么前者过不了, 后者可以.

```
1 class Solution(object):
2     def rotate(self, nums, k):
3         k %= len(nums)
4         nums = nums[-k:] + nums[:-k]
```

```
1 class Solution:
2     def rotate(self, nums, k):
3         k = k % len(nums)
4         nums[:] = nums[-k:] + nums[:-k]
```

其中着重练习了单调栈:

[739. 每日温度 - 力扣 \(LeetCode\)](#)

```
1 class Solution(object):
2     def dailyTemperatures(self, temperatures):
3         st = []
4         res = [0] * len(temperatures)
5         for i, t in enumerate(temperatures):
6             if not st:
7                 st.append(i)
8             else:
9                 while st and temperatures[st[-1]] < t:
10                     res[st[-1]] = i - st[-1]
11                     st.pop()
12                 st.append(i)
13         return res
```

[84. 柱状图中最大的矩形 - 力扣 \(LeetCode\)](#)

Key: 如何遍历?

Solution: 例如 [3, 1, 4, 1, 5, 9, 2, 6]

[3] => [1] => [1, 4] => [1] => [1, 5] => [1, 5, 9] => [1, 2] => [1, 2, 6]

其中红色的一步, 每次枚举最右侧是 9 的矩形 ([9], [5, 9]) (不会枚举 [1, 5, 9], 因为 [4, 9, 2, ...] 之后会枚举到的)

其中在 height 末尾加 0 是为了保证最后把 [1, 2, 6] 完整的枚举一遍 ([6], [2, 6], [1, 5, 9, 2, 6])


```

1 class Solution(object):
2     def largestRectangleArea(self, heights):
3         heights.append(0)
4         st = []
5         MAX = 0
6         for i in range(len(heights)):
7             while st and heights[st[-1]] > heights[i]:
8                 h = heights[st.pop()]
9                 w = i if not st else i - st[-1] - 1
10                MAX = max(MAX, h * w)
11            st.append(i)
12        return MAX

```

85. 最大矩形 - 力扣 (LeetCode)

```

1 class Solution(object):
2     def maximalColumn(self, col):
3         col.append(0)
4         st = []
5         MAX = 0
6         for i, x in enumerate(col):
7             while st and col[st[-1]] > x:
8                 if len(st) >= 2:
9                     MAX = max(MAX, col[st[-1]] * (i - st[-2] - 1))
10                else:
11                    MAX = max(MAX, col[st[-1]] * i)
12                st.pop()
13            st.append(i)
14        return MAX
15    def maximalRectangle(self, matrix):
16        m, n = len(matrix), len(matrix[0])
17        pre = [0] * n
18        MAX = 0
19        for i in range(m):
20            for j in range(n):
21                pre[j] = pre[j] + 1 if matrix[i][j] == "1" else 0
22            MAX = max(MAX, self.maximalColumn(pre.copy()))
23        return MAX

```