

Assignment #9: Mock Exam立冬

Updated 1856 GMT+8 Nov 7, 2025

2025 fall, Complied by 杨浩、化院

说明:

1. Nov月考: AC5 (请改为同学的通过数)。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本人头像，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. 1. 题目

1.1 M02255: 重建二叉树

<http://cs101.openjudge.cn/practice/02255/>

思路:

- 前序顺序遍历构造树根，中序序列以树根分割左右子树

代码

```

1  class Treenode:
2      def __init__(self, name):
3          self.name = name
4          self.left = None
5          self.right = None
6      def buildTree(begin, end,cnt):
```

```

7     if begin > end:
8         return None,cnt-1
9     root = Treenode(pre_order[cnt])
10    for i in range(begin, end+1):
11        if in_order[i]==root.name:
12            root.left,cnt = buildTree(begin, i-1,cnt+1)
13            root.right,cnt = buildTree(i+1, end,cnt+1)
14    return root,cnt
15 def postorder(root):
16     if root:
17         postorder(root.left)
18         postorder(root.right)
19         res.append(root.name)
20
21
22 while True:
23     try:
24         pre_order,in_order = input().split()
25         pre_order = list(pre_order)
26         in_order = list(in_order)
27         root,cnt=buildTree(0,len(in_order)-1,0)
28         res=[]
29         postorder(root)
30         print(''.join(res))
31     except EOFError:
32         break

```

Fence 1

代码运行截图 (至少包含有"Accepted")

#50740792提交状态

		查看	提交	统计	提问
状态:	Accepted				
源代码	<pre> class Treenode: def __init__(self, name): self.name = name self.left = None self.right = None def buildTree(begin, end,cnt): if begin > end: return None,cnt-1 </pre>	基本信息	#: #: 50740792 题目: 题目: M02255 提交人: 提交人: 25n2400011769 内存: 内存: 3548kB 时间: 时间: 20ms 语言: 语言: Python3 提交时间: 提交时间: 2025-11-07 17:27:20		

Figure 1

1.2 M02774: 木材加工

<http://cs101.openjudge.cn/practice/02774/>

思路:

- 二分查找, 0单独讨论

代码

```

1  def check(max_len,k,n):
2      count = 0
3      for i in range(n):
4          count += len_list[i]//max_len
5          if count >= k:
6              return True
7      if count<k:
8          return False
9      else:
10         return True
11 def fen(k,n):
12     if sum(len_list)<k:
13         return 0
14     left=1
15     right=max(len_list)
16     while left<=right:
17         mid=(left+right)//2
18         if check(mid,k,n):
19             left=mid+1
20         else:
21             right=mid-1
22     return right
23
24
25
26 n,k=map(int,input().split())
27 len_list=[]
28 for i in range(n):
29     len_list.append(int(input()))
30 print(fen(k,n))

```

Fence 2

代码运行截图 (至少包含有"Accepted")

#50741023提交状态

[查看](#) [提交](#) [统计](#) [提问](#)

状态: Accepted

源代码

```

def check(max_len,k,n):
    count = 0
    for i in range(n):
        count += len_list[i]//max_len
        if count >= k:
            return True
    if count<k:

```

基本信息
#:
题目:
提交人:
内存:
时间:
语言:
提交时间:

50741023
M02774
25n2400011769
3936kB
49ms
Python3
2025-11-07 17:36:46

Figure 2

1.3 M02788: 二叉树 (2)

<http://cs101.openjudge.cn/practice/02788/>

思路：

- 此问题中的一个满二叉树，树根为 x ，最左边的叶子节点为 $x \cdot 2^{n-1}$ ，最右边的叶子节点为 $(x+1) \cdot 2^{n-1} - 1$

代码

```

1 def find_line(k):
2     for i in range(len(check_list)):
3         if check_list[i]>k:
4             return i+1
5
6 check_list=[2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048,
7           4096, 8192, 16384, 32768, 65536, 131072, 262144, 524288,
8           1048576, 2097152, 4194304, 8388608, 16777216, 33554432,
9           67108864, 134217728, 268435456, 536870912, 1073741824]
10 while True:
11     m,n=map(int,input().split())
12     if m==0 and n==0:
13         break
14     m_line=find_line(m)
15     n_line=find_line(n)
16     count_line=n_line-m_line
17     if count_line==0:
18         print(1)
19         continue
20     count=2**count_line-1
21     left=m*2**count_line
22     right=(m+1)*2**count_line-1
23     if right<=n:
24         print(2**(count_line+1)-1)
25         continue
26     elif left<=n:
27         print(count+(n-left)+1)
28         continue
29     else:
30         print(count)
31         continue

```

Fence 3

代码运行截图 (至少包含有"Accepted")

#50741375提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
def find_line(k):
    for i in range(len(check_list)):
        if check_list[i]>k:
            return i+1

check_list=[2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048, 4096, 8192,
while True:
    ...
```

基本信息

#: 50741375
题目: M02788
提交人: 25n2400011769
内存: 3672kB
时间: 48ms
语言: Python3
提交时间: 2025-11-07 17:52:44

Figure 3

1.4 M04081: 树的转换

<http://cs101.openjudge.cn/practice/04081/>

思路:

- 利用 `stack` 构造一个树，分别记录树的深度和对应二叉树深度。根据父节点是否已有子节点判断二叉树深度为上一个子节点+1还是父节点+1。

代码

```
1 class Treenode:
2     def __init__(self, depth, twoWayDepth):
3         self.children = []
4         self.depth = depth
5         self.twoWayDepth = twoWayDepth
6 str_lisr=input()
7 root = Treenode(0,0)
8 stack=[root]
9 max_depth=0
10 max_twoWayDepth=0
11 for i in str_lisr:
12     if i=='d':
13         pr=Treenode(stack[-1].depth+1,0)
14         if stack[-1].children:
15             twoWayDepth=stack[-1].children[-1].twoWayDepth+1
16         else:
17             twoWayDepth=stack[-1].twoWayDepth+1
18         pr.twoWayDepth=twoWayDepth
19         max_depth=max(max_depth,pr.depth)
20         max_twoWayDepth=max(max_twoWayDepth,twoWayDepth)
21         stack[-1].children.append(pr)
22         stack.append(pr)
23     else:
24         stack.pop()
25 print(f'{max_depth} => {max_twoWayDepth}')
```

Fence 4

代码运行截图 (至少包含有"Accepted")

#50742213提交状态

状态: Accepted

源代码

```
class Treenode:
    def __init__(self, depth, twoWayDepth):
        self.children = []
        self.depth = depth
        self.twoWayDepth = twoWayDepth
    strList=input()
root = Treenode(0, 0)
stack=[root]
```

基本信息

#: 50742213
题目: M04081
提交人: 25n2400011769
内存: 3652kB
时间: 26ms
语言: Python3
提交时间: 2025-11-07 18:35:08

Figure 4

1.5 M04117: 简单的整数划分问题

dfs, dp, <http://cs101.openjudge.cn/practice/04117/>

思路:

- 题干说有多组数据，虽然样例只有一组，但不用 `try` `except` 会WA。
- 需要不重不漏的找拆分。先将一个数分成大小两个数 `A, B` (可以取等)，然后将大的数 `A` 递归，继续分解且要求大的数 `A` 分解出来的两个数 `C, D` 满足：较小的 `D` 大于等于 `B`。最初拆分的时候可以取 `B==0` 且这组拆分不进行递归，但之后的拆分均要求 `B>=1`。
- 这个拆分的正确性判断思路如下：
 - 维持拆分数列的有序性是这个方法的关键
 - 举 `n==5` 的例子说明，首先可以分解为 `[5]` (这组不参与后续拆分)，`[4, 1]`，`[3, 2]` 拆成2个数做到了不重不漏。
 - 拆解为3个数的所有拆法一定可以从2个数中分解1个数实现（逆向更好说明，3个数中任意找2个数合并后剩下的2个数一定存在于拆成2个数的拆法中）。
 - 限制大数分解出来的小数大小（即 `A` 分解出来的两个数 `C, D` 满足：较小的 `D` 大于等于 `B`），可以实现不重。例如，若将 `[3, 2]` 拆为 `[2, 1, 2]` 即 `[2, 2, 1]` 一定可以合并为 `[4, 1]`。即不满足大数拆解条件一定会出现重复计数。
 - 拆解到3个数时有 `[3, 1, 1], [2, 2, 1]`。可以想象假设有更多的数 `[..., a, b, ...]` (`k`个数) 不拆分第一个数，拆分后面的数为 `[..., a, m, n, ...]` (`k+1`个)，重新排序，并将最大的两个数合并，这个拆法一定会先 `k`个数中第一个数更大的情况中出现，也就说明将后面 `b`拆分为 `m+n`也会重复。即不拆第一个数也会重复。
 - 综上，将第一个数拆分，并满足限制大数分解出来的小数大小（即 `A` 分解出来的两个数 `C, D` 满足：较小的 `D` 大于等于 `B`）条件就不重复，依次遍历就可以得到全部答案。

代码

```

1  from functools import lru_cache
2  @lru_cache(maxsize=None)
3  def chaifen(num,min_line):
4      cnt=0
5      if num < 2*min_line:
6          return 0
7      pr=num-min_line
8      if min_line==0:
9          pr-=1
10     cnt+=1
11     while pr>=num-pr:
12         cnt+=1+chaifen(pr,num-pr)
13         pr-=1
14     return cnt
15     while True:
16         try:
17             n=int(input())
18             print(chaifen(n,0))
19         except EOFError:
20             break

```

Fence 5

代码运行截图 (至少包含有"Accepted")

#50743110提交状态

状态: Accepted		查看	提交	统计	提问
源代码	<pre> from functools import lru_cache @lru_cache(maxsize=None) def chaifen(num,min_line): cnt=0 if min_line==0: cnt+=1 if num < 2*min_line: ... </pre>	基本信息	# : 50743110 题目: 04117 提交人: 25n2400011769 内存: 5980kB 时间: 25ms 语言: Python3 提交时间: 2025-11-07 19:23:42		

Figure 5

1.6 M04137:最小新整数

monotonous-stack, <http://cs101.openjudge.cn/practice/04137/>

思路:

- 贪心: 假设数为abcdefg, 需要去除2个, 即需要留下5个, 则在前3个 (需要保证可以取完5个数) 数字abc中选一个最小的数作为最高位, 可以使得这步解最优, 如果出现相等的数字, 则选取最先出现的数字, 使得后续的选择得到更优解。

代码

```

1  def find_min(n,begin,left):
2      if left == 0:
3          return ''

```

```

4     mini='9'
5     index=begin
6     for i in range(begin,len(n)-(left)+1):
7         if n[i] < mini:
8             mini = n[i]
9             index=i
10    return n[index]+find_min(n,index+1,left-1)
11 t=int(input())
12 for i in range(t):
13     n,k=input().split()
14     k=int(k)
15     left=len(n)-k
16     print(find_min(n,0,left))

```

Fence 6

代码运行截图 (至少包含有"Accepted")

#50742065提交状态

		查看	提交	统计	提问
状态:	Accepted				
源代码	<pre> def find_min(n,begin,left): if left == 0: return '' mini='9' index=begin for i in range(begin,len(n)-(left)+1): if n[i] < mini: </pre>				
基本信息	# : 50742065 题目: M04137 提交人: 25n2400011769 内存: 3604kB 时间: 24ms 语言: Python3 提交时间: 2025-11-07 18:25:57				

Figure 6

2. 2. 学习总结和收获

考试题目AC5个，M04117: 简单的整数划分问题考试的时候没有看见说多组数据，看样例给了一个以为是只有1个，一直WA，下来写一个try就AC了。本周考试题目总体难度一般，没有出现T20576: printExp那种庞大书写量的题目。考题思维量也较为正常，不过考题中需要构造树对象的题目不多，一定程度上超出了前期训练时的意料。

树中典型的题目基本都训练过了，接下来的训练可以加大练习题目的书写量和思考量。