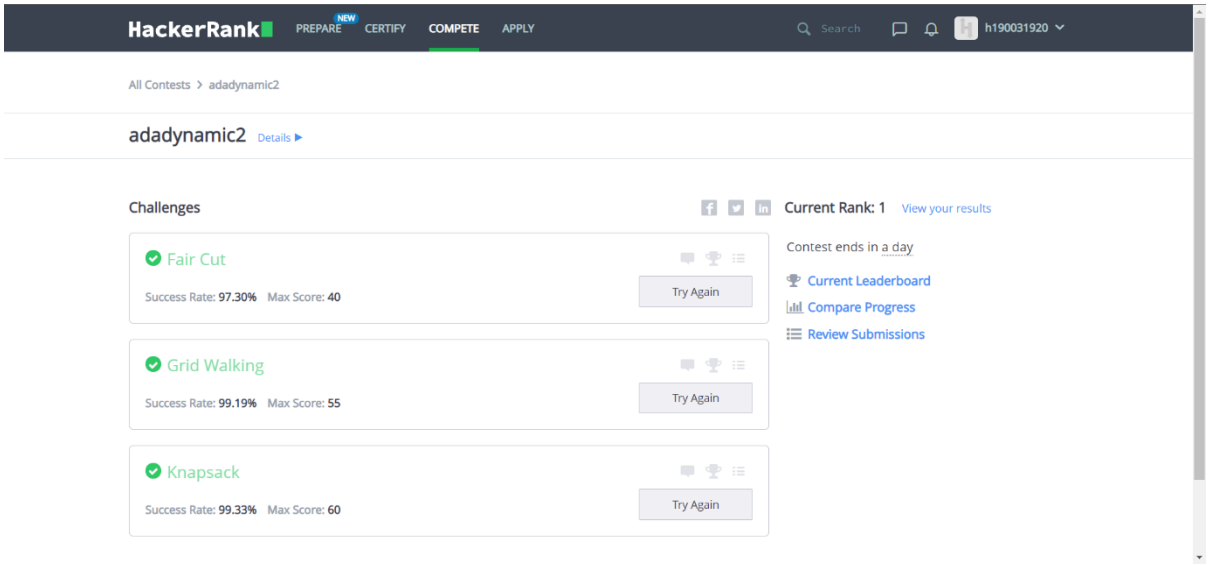
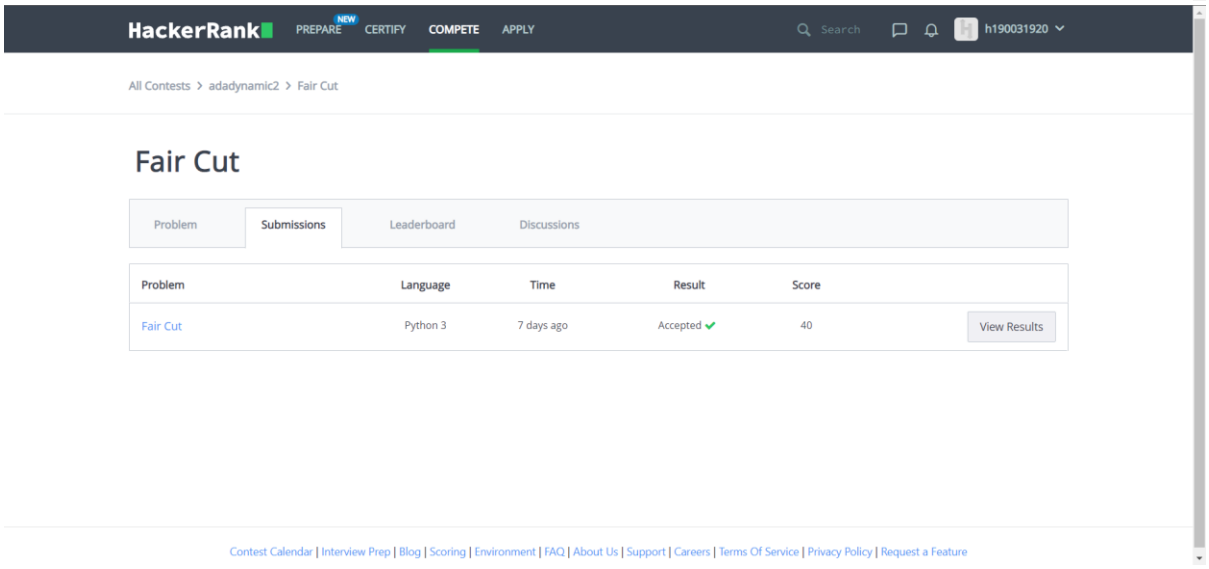
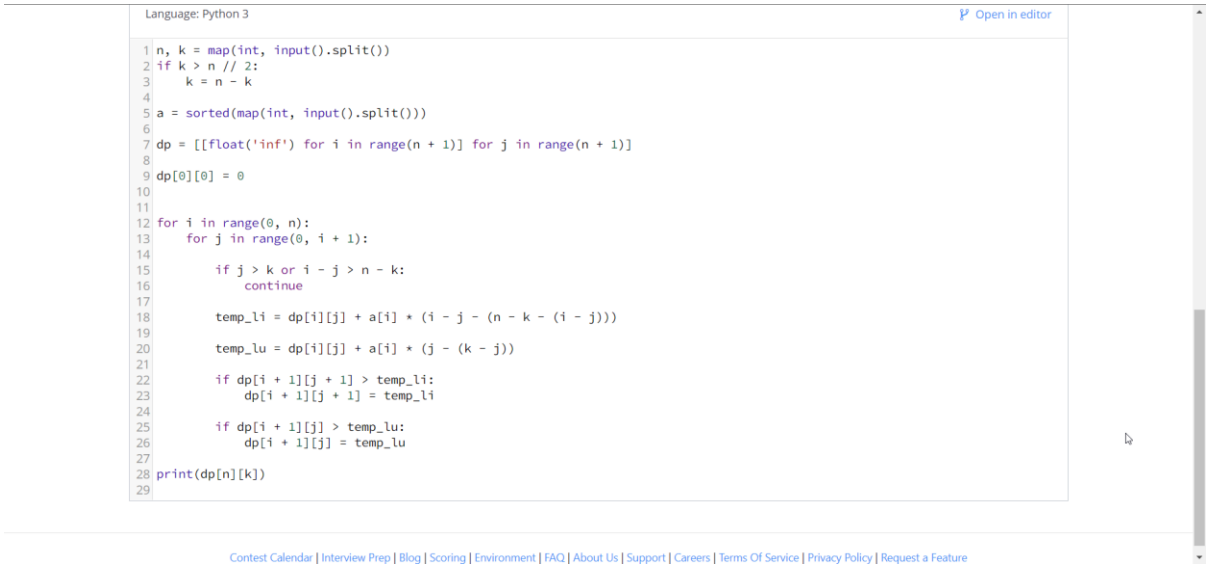
190031920 Nikhil Reddy Avuthu ADA Skill – 10







Grid Walking

Problem | Submissions | Leaderboard | Discussions

Submitted 7 days ago • Score: 55.00 Status: Accepted

✔ Test Case #0	✔ Test Case #1	✔ Test Case #2
✔ Test Case #3	✔ Test Case #4	✔ Test Case #5
✔ Test Case #6	✔ Test Case #7	✔ Test Case #8
✔ Test Case #9	✔ Test Case #10	✔ Test Case #11
✔ Test Case #12		

Submitted Code

Language: Python 3 Open in editor

```
import re
import sys

def gridWalking(m, x, D):
    MOD = 10 ** 9 + 7
    n = len(D)
    md = [[0 for _ in range(n + 1)] for _ in range(m + 1)]
    for i in range(n):
        M = [[0 for _ in range(m + 1)] for _ in range(D[i] + 1)]
        for j in range(1, D[i] + 1):
            M[j][0] = 1
        for j in range(1, m + 1):
            for k in range(1, D[i] + 1):
                M[k][j] = 0
                if k - 1 > 0:
                    M[k][j] = (M[k][j] + M[k - 1][j - 1]) % MOD
                if k + 1 <= D[i]:
                    M[k][j] = (M[k][j] + M[k + 1][j - 1]) % MOD
        md[0][i + 1] = 1
        for j in range(1, m + 1):
            md[j][i + 1] = M[x[i]][j]
    c = [[0 for _ in range(m + 1)] for _ in range(m + 1)]
    for i in range(m + 1):
        c[i][0] = 1
        c[i][i] = 1
    for i in range(1, m + 1):
        for j in range(1, i):
            c[i][j] = (c[i - 1][j - 1] + c[i - 1][j]) % MOD
    mdt = [[0 for _ in range(n + 1)] for _ in range(m + 1)]
    for i in range(m + 1):
        mdt[i][1] = md[i][1]
    for i in range(n + 1):
        mdt[0][i] = 1
    for i in range(2, n + 1):
        for j in range(1, m + 1):
            mdt[j][i] = 0
            for k in range(j + 1):
                mdt[j][i] = (
                    mdt[j][i] + ((c[j][j - k] * ((mdt[k][i - 1] * md[j - k][i]) % MOD)) % MOD)) % MOD
    return mdt[m][n]

if __name__ == '__main__':
    fptr = open(os.environ['OUTPUT_PATH'], 'w')

    t = int(input().strip())

    for t_itr in range(t):
        first_multiple_input = input().rstrip().split()

        n = int(first_multiple_input[0])

        m = int(first_multiple_input[1])

        x = list(map(int, input().rstrip().split()))

        D = list(map(int, input().rstrip().split()))

        result = gridWalking(m, x, D)

        fptr.write(str(result) + '\n')

    fptr.close()
```

Knapsack

Problem | Submissions | Leaderboard | Discussions

Submitted 3 minutes ago • Score: 60.00 Status: Accepted

✔ Test Case #0	✔ Test Case #1	✔ Test Case #2
✔ Test Case #3	✔ Test Case #4	✔ Test Case #5
✔ Test Case #6	✔ Test Case #7	✔ Test Case #8
✔ Test Case #9	✔ Test Case #10	✔ Test Case #11

Submitted Code

Language: Python 3 Open in editor

```python
import random
import re
import sys

#
# Complete the 'unboundedKnapsack' function below.
#
# The function is expected to return an INTEGER.
# The function accepts following parameters:
#  1. INTEGER k
#  2. INTEGER_ARRAY arr
#

def maxSum(arr,k):
    subset = [[False for i in range(k+1)] for j in range(len(arr)+1)]
    for i in range(len(arr)+1):
        subset[i][0] = True
    for i in range(1,len(arr)+1):
        for j in range(1,k+1):
            if(arr[i-1]>j):
                subset[i][j] = subset[i-1][j]
            else:
                subset[i][j] = subset[i-1][j] or subset[i][j-arr[i-1]]
    j = k
    while j>=0:
        if subset[len(arr)][j] == True:
            return j
        j-=1
if __name__ == '__main__':
    fptr = open(os.environ['OUTPUT_PATH'], 'w')

    t = int(input().strip())

    for i in range(t):
        first_multiple_input = input().rstrip().split()

        n = int(first_multiple_input[0])

        k = int(first_multiple_input[1])

        arr = list(map(int, input().rstrip().split()))

        result = maxSum(arr, k)

        fptr.write(str(result) + '\n')

    fptr.close()
```