문제 1. 상담 ( #14501 )

상담원 정우는 곧 퇴사를 하려고 합니다.

퇴사 날은 N+1일 이라서, 남은 N일 동안 상담을 해서 돈을 많이 벌어서 나가고 싶습니다.

가장 돈을 많이 받도록 상담을 골라서 신청했을 때, 그 금액을 출력하는 프로그램을 작성하세요.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1일 | 2일 | 3일 | 4일 | 5일 | 6일 | 7일 |
| T | 4 | 1 | 3 | 1 | 2 | 3 | 2 |
| P | 10 | 20 | 30 | 40 | 35 | 150 | 10 |

7

4 10

1 20

3 30

1 40

2 35

3 150

2 10

95

문제 2. 냅색 ( #12865 )

물건의 수 A와 배낭의 무게 B가 주어집니다.

순서대로 물건의 무게 W와 가치 V가 주어집니다.

배낭에 가장 가치있게 담았을 경우의 가치를 출력하는 프로그램을 작성하세요.

4 7

6 13

4 8

3 6

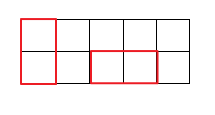
5 12

14

문제 3. 피보나치 타일링 ( #11726 )

2xN 크기의 직사각형이 있습니다.

1x2, 2x1 크기의 타일로 채우는 방법의 수를 구하는 프로그램을 작성하시오.



9

55

문제 4. RGB 색칠하기 ( #1149 )

디자이너 정우는 한 거리의 건물들의 색을 바꾸려고 한다.

Red, Green, Blue 세가지 색중에 하나를 선택해서 색을 바꾸려고 하는데

연속된 색으로 두 건물을 칠하면 건물들의 경계가 모호해져서 아름답지 않다.

건물의 개수 N과

건물을 R,G,B 로 색칠하는 비용이 주어졌을 때,

모든 집을 칠하는 비용의 최솟값을 구하는 프로그램을 작성하세요.

3

26 40 83

49 60 57

13 89 99

96

정답코드

(1)

N = int(input())

table = [[] for \_ in range(N)]

for i in range(N):

a, b = map(int, input().split())

table[i] = [a, b]

dp = [0 for \_ in range(N+1)]

for idx in range(N)[::-1]:

if idx + table[idx][0] > N :

dp[idx] = dp[idx + 1]

else :

dp[idx] = max(dp[idx + table[idx][0]] + table[idx][1], dp[idx + 1])

print(dp[0])

(2)

import sys

input = sys.stdin.readline

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

items = []

# y축 = 물건의 무게, 가치 / x축 = 배낭의 무게

# +1 달아서 0,0은 0으로 추가 (물건을 못넣을 때)

dp = [[0]\*(k+1) for \_ in range(n+1)]

for y in range(1, n+1): # 물건이

weight, value = map(int,input().split())

for x in range (1, k+1): # 가방 무게가 x일 때

# 물건을 담을 수 있을 때 (x보다 무게가 작거나 같을 때)

if x < weight :

dp[y][x] = dp[y-1][x]

# 물건을 담을 수 없을 때 ( 전에 넣어뒀던 무게로 유지)

else:

dp[y][x] = max(dp[y-1][x],dp[y-1][x-weight] + value )

print(dp[n][k])

(3)

n = int(input())

dp = [-1 for \_ in range(n)]

for idx in range(n):

if idx < 3 :

dp[idx] = idx+1

else:

dp[idx] = dp[idx-1] + dp[idx-2]

print(dp[-1]%10007)

(4)

n = int(input())

graph = [list(map(int,input().split())) for \_ in range(n)]

dp = [[0 for \_ in range(3)] for \_ in range(2)]

for k in range(3):

dp[0][k] = graph[0][k]

for i in range(1,n):

for j in range(3):

if j == 0:

dp[1][j] = min(dp[0][1] + graph[i][j], dp[0][2] + graph[i][j])

if j == 1:

dp[1][j] = min(dp[0][0] + graph[i][j], dp[0][2] + graph[i][j])

if j == 2:

dp[1][j] = min(dp[0][1] + graph[i][j], dp[0][0] + graph[i][j])

for f in range(3):

dp[0][f] = dp[1][f]

print(min(dp[-1]))