문제 1. 수열 ( #2559 )

수열의 길이 A와, 간격 B가 주어집니다.

그리고 수열이 하나 주어집니다.

주어진 간격만큼의 합을 구해서, 가장 큰 수를 출력하는 프로그램을 작성하세요.

**10 2**

**3 -2 -4 -9 0 3 7 13 8 -3**

**21**

문제 2. 수열 가장 크게 만들기 ( #1912 )

수열의 길이 N과 수열이 주어집니다.

연속된 몇 개의 수를 선택하여 구할 수 있는 합 중에서, 가장 큰 합을 구하는 프로그램을 작성하세요.

**10**

**10 -4 3 1 5 6 -35 12 21 -1**

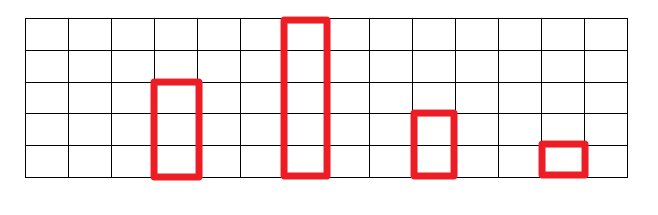
**33**

문제 3. 텐트 세우기 ( #2304, #14719 )

막대 기둥의 개수 N이 주어집니다.

그리고 기둥의 위치와 높이가 X,Y 좌표로 주어집니다.

가장 높은 기둥을 중심으로 텐트의 천을 씌워, 그 텐트가 차지하는 크기를 계산하는 프로그램을 작성하시오



**7**

**2 4**

**11 4**

**15 8**

**4 6**

**5 3**

**8 10**

**13 6**

**98**

문제 4. 2차원 누적합 ( #11660 )

4X4 크기의 표가 주어집니다.

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| 2 | 3 | 4 | 5 |
| 3 | 4 | 5 | 6 |
| 4 | 5 | 6 | 7 |

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 3 | 6 | 10 |
| 3 |  |  |  |
| 6 |  |  |  |
| 10 |  |  |  |

직사각형의 좌측 위 좌표와 우측 아래 좌표가 주어졌을 때, 그 범위의 합을 구하는 프로그램을 작성하시오.

**1 2 3 4**

**2 3 4 5**

**3 4 5 6**

**4 5 6 7**

**2 2 3 4**

**27**

문제 5. 이모스의 장애물 경주 ( #3020, #17611 )

**이모스법** : <https://imoz.jp/algorithms/imos_method.html>

5명의 사람들이 장애물 경주를 하려고 합니다.

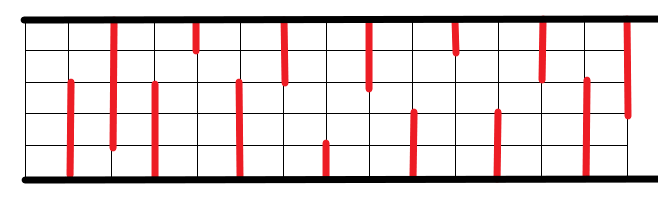
각각의 레인에는 장애물이 설치되는데, 이 장애물은 길이가 제각각입니다.

어떤 장애물은 1번 레인에서 3번 레인을 가로막고

어떤 장애물은 4번 레인에서 5번 레인을 가로막습니다.

장애물은 순서대로 1번 레인에서 설치되거나, 5번 레인에서 설치됩니다.

이때, 가장 장애물이 적은 레인의 장애물은 몇 개인지 계산하시오.



**6 7**

**1**

**5**

**3**

**3**

**5**

**1**

**2 3**

정답코드

(1)

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

array = list(map(int,input().split()))

prefix = [0 for \_ in range(a+2)]

for i in range(0,a):

prefix[i+1] = prefix[i] + array[i]

answer = []

for k in range(b,a+1):

answer.append(prefix[k] - prefix[k-b])

print(max(answer))

(2)

import sys

input = sys.stdin.readline

n = int(input())

numbers = list(map(int,input().split()))

prefix = [-1001]\*(n+1)

for i in range(n):

prefix[i+1] = max(prefix[i] + numbers[i], numbers[i])

print(max(prefix))

(3)

n = int(input())

graph = [0]\*10001

x\_list = []

y\_list = []

for i in range(n):

x, y = map(int,input().split())

graph[x] = y

x\_list.append(x)

y\_list.append(y)

maxHeight = max(y\_list)

maxWidth = max(x\_list)

prefix = [0]\*(maxWidth+2)

suffix = [0]\*(maxWidth+2)

maxPoint = []

#prefix계산

h = 0

for f in range(1,maxWidth+3):

if(graph[f] == maxHeight):

maxPoint.append(f)

break

h = max(h, graph[f])

prefix[f] = prefix[f-1] + h

h = 0

for b in range(maxWidth,0,-1):

if(graph[b] == maxHeight):

maxPoint.append(b)

break

h = max(h, graph[b])

suffix[b] = suffix[b+1] + h

#정답 합치기

answer = max(prefix) + max(suffix)

answer += (maxPoint[1] - maxPoint[0] + 1 )\*maxHeight

print(answer)

(4)

import sys

input = sys.stdin.readline

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

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

prefix = [[0 for \_ in range(n+1)] for \_ in range(n+1)]

for y in range(0,n):

for x in range(0,n):

prefix[y+1][x+1] = prefix[y][x+1] + prefix[y+1][x] - prefix[y][x] + graph[y][x]

for \_ in range(m):

y1, x1, y2, x2 = map(int,input().split())

answer = prefix[y2][x2] - prefix[y2][x1-1] - prefix[y1-1][x2] + prefix[y1-1][x1-1]

print(answer)

(5)

import sys

input = sys.stdin.readline

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

line = [0 for \_ in range(h)]

for t in range(n):

height = int(input())

if t%2 == 0:

line[0] += 1

line[height] -= 1

if t%2 == 1:

line[h-height] += 1

prefix = [0 for \_ in range(h+1)]

for i in range(h):

prefix[i+1] = prefix[i] + line[i]

prefix = prefix[1:]

print(min(prefix), prefix.count(min(prefix)))