Fall 2021
ECE30017-01 Problem Solving through Computational Thinking

Week 5

- C4. Tower of Hanoi
  Deadline: 11:59 PM, 1 October (Fri)

- P4. Tower of Hanoi
  Deadline: 11:59 PM, 5 October (Tue)

C4. Tower of Hanoi

The Tower of Hanoi is a game to play with 3 rods (rod 1, rod 2, and rod 3) and n disks of different sizes (disk 1 disk 2, …, disk n). Disk i is lager than disk j if only if $i > j$. The game starts with the disks distributed upon three rods. A disk can be stacked upon another disk in a rod if the former is smaller than the latter. Thus, the disks in each rod are stacked in the ascending order of their sizes, that is, the top-most one is the smallest one in the rod.

The goal of the game is to place all disks in a certain rod (calling it “destination” rod) with the minimum number of disk moves. The game player is allowed to move one disk at a time, from rod to rod as long as the above-stated condition is satisfied.

Note that this game is a generalized version of the traditional Tower of Hanoi game which starts with all disks stacked in one specific rod. It is known that, in the traditional game, the minimum num (ber of disk moves is 2^n -1 for n disks.

Write a program that reads the initial state of the game and then finds the minimum number of disk moves to finish the game. (continued)

Requirements

Input data

- The first line from the standard input has two numbers n and k. n stands for the number of disks where $1 \leq n \leq 50$. k represents the destination rod where $1 \leq k \leq 3$.
- The second to forth lines represent how initially the disks are stacked over three rods. Each line starts with n_i, the number of the disks in rod $i, 1 \leq i \leq 3$ followed by n_i integers that represent the disks initially stacked from bottom to top in rod i.

Output

- write the number of the minimum disks moves to the standard output
- your program must generate the result within 0.5 second.

Examples of test cases

Input 1

```
4 2
2 4 3
1 2
1 1
```

Output 1

```
13
```

Input 2

```
4 3
4 4 3 2 1
0
0
```

Output 2

```
15
```

Team for C4

401	이인석	최시령
402	정성목	강석운
403	박건희	강동인
404	이혜림	안제현
405	박은찬	이수아
406	전영우	차경민
407	김영표	홍순규
408	남진우	이찬효
409	김해린	권혁찬

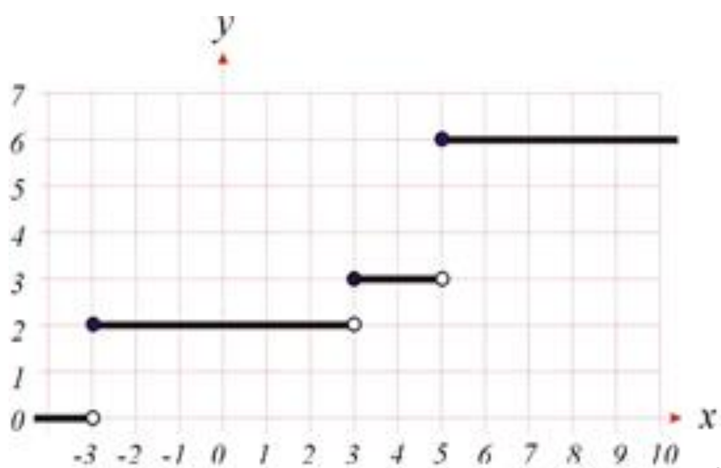
P4. Step Function (1/2)

A step function is a function f: $X \rightarrow C$ where the domain X is a real line, R^1 = (-∞, +∞) and C={c_0, c_1, ….. , c_k} is a finite set of constants such that

1. $X = A_0 \cup A_1 \cup \cdots \cup A_k$ where $A_0 = (-\infty, a_1)$, $A_k = [a_k, \infty)$ and
   $A_i = [a_i, a_{i+1})$ where $a_i < a_{i+1}$ for $0 < i < k$.
2. $f(x) = c_i$ for all x in A_i for $0 \leq i \leq k$.

You may assume that f is non-negative and non-decreasing in X,

that is, $c_0 = 0$ and $c_i < c_{i+1}$ for $0 \leq i < k$. An example of a non-negative, non-decreasing step function is shown below:



$A_0 = (-\infty, -3)$, $A_1 = [-3, 3)$, $A_2 = [3, 5)$,
$A_3 = [5, \infty)$,

$c_0 = 0,\ c_1 = 2,\ c_2 = 3,\ c_3 = 6$

(Continued)

P4. Step Function (2/2)

A step function f can be represented as a sequence of starting points,

$((a_{1,} f(a_1)), (a_{2,} f(a_2)), \ldots\ldots, (a_{k,} f(a_k)))$.

For example, the function in the previous page can be specified by

((−3, 2), (3, 3), (5, 6)).

Given two such step functions f and g, together with two integers p and q such that $p \le q$, write a program that evaluate the following expression:

$$\left(\sum_{i=p}^{q} \max\{f(i), g(i)\}\right) \quad mod \quad 10007$$

(continued)

Requirement

- **Input**
  - Input data are given from the standard input
  - The first part of input data is a representation of function f.
    - The definition starts with value k_f for $1 \le k_f \le 1{,}000{,}000$ which is the number of points to specify function f.
    - Subsequently, k_f lines follow where the i-th line has a_i and $f(a_i)$ for $-2{,}000{,}000{,}000 \le a_i \le 2{,}000{,}000{,}000$ and $1 \le f(a_i) \le 2{,}000{,}000{,}000$.
  - After that, the specification of function g is provided in the same manner.
    - The first line gives value k_g, that is, the number of points for function g. After that, k_g lines follow where the i-th line has a_i and $g(a_i)$.
  - The last line gives p and q for $-2{,}000{,}000{,}000 \le p \le q \le 2{,}000{,}000{,}000$.

- **Output**
  - Your program should print one number to the standard output within 0.5 second.

Test Case Example

Input data

```
3
-3 2
3 3
5 6
1
4 5
1 5
```

output

```
18
```