HackerRank | Prepare Certify Compete Apply Search 2

All Contests > CSE-g-week5 > Equal Stacks 2

Equal Stacks 2

Problem Submissions Leaderboard Discussions

You have three stacks of cylinders where each cylinder has the same diameter, but they may vary in height. You can change the height of a stack by removing and discarding its topmost (starting from index 0) cylinder any number of times.
Find the maximum possible height of the stacks such that all of the stacks are exactly the same height. This means you must remove zero or more cylinders from the top of zero or more of the three stacks until they are all the same height, then return the height.
Example:
h1=[1,2,1,1]
h2=[1,1,2]
h3= [1,1]
There are 4, 3 and 2 cylinders in the three stacks, with their heights in the three arrays. Remove the top 2 cylinders from h1 (heights = [1, 2]) and from h2 (heights = [1, 1]) so that the three stacks all are 2 units tall. Return it as the answer. Note: An empty stack is still a stack.

Function Description

int equalStacks(int h1_count, int* h1, int h2_count, int* h2, int h3_count, int* h3);
Strictly use the above function prototype.

Input Format

The first line contains three space-separated integers, n1, n2, and n3, the numbers of cylinders in stacks 1,2 and 3. The subsequent lines describe the respective heights of each cylinder in a stack from top to bottom:
The second line contains n1 space-separated integers, the cylinder heights in stack. The first element is the top cylinder of the stack.
The third line contains n2 space-separated integers, the cylinder heights in stack. The first element is the top cylinder of the stack.
The fourth line contains n3 space-separated integers, the cylinder heights in stack. The first element is the top cylinder of the stack.

Constraints

0 < n1, n2, n3 <=105
0 < height of any cylinder <=100

Output Format

int: the height of the stacks when they are equalized

Sample Input 0

```
5 3 4
3 2 1 1 1
4 3 2
1 1 4 1
```

Sample Output 0

```
5
```

Contest ends in 15 minutes

Submissions: 53
Max Score: 10
Difficulty: Medium

Rate This Challenge:

☆☆☆☆☆

More

C

```c
#include <stdio.h>
#include <string.h>
#include <math.h>
#include <stdlib.h>

#define MAX_STACK_SIZE 200

typedef struct {
  int buffer[MAX_STACK_SIZE];
  int top;
} STACK;

int equalStacks(STACK *s1, STACK *s2, STACK *s3);
void push(int e, STACK *s);
int pop(int *e, STACK *s);
void peek(int *e, STACK *s);
void reverse(STACK *s);
int count(STACK *s);

int main() {
  int n1, n2, n3;
  scanf("%d %d %d", &n1, &n2, &n3);

  STACK s1, s2, s3;
  s1.top = 0;
  s2.top = 0;
  s3.top = 0;
  int e;

  for (int i=0; i<n1; i++) {
    scanf("%d", &e);
    push(e, &s1);
  }
  reverse(&s1);

  for (int i=0; i<n2; i++) {
    scanf("%d", &e);
    push(e, &s2);
  }
  reverse(&s2);

  for (int i=0; i<n3; i++) {
    scanf("%d", &e);
    push(e, &s3);
  }
  reverse(&s3);

  int result = equalStacks(&s1, &s2, &s3);
  printf("%d", result);

  return 0;
}

void push(int e, STACK *s) {
  if (s->top == MAX_STACK_SIZE)
    return;

  s->buffer[s->top] = e;
  s->top++;
}

int pop(int *e, STACK *s) {
  if (s->top == 0)
    return 1;

  *e = s->buffer[s->top-1];
  s->top--;
  return 0;
}

void peek(int *e, STACK *s) {
  *e = s->buffer[s->top-1];
}

void reverse(STACK *s) {
  STACK t1, t2;
  t1.top = 0;
  t2.top = 0;
  int e;

  while (pop(&e, s) == 0)
    push(e, &t1);

  while (pop(&e, &t1) == 0)
    push(e, &t2);

  while (pop(&e, &t2) == 0)
    push(e, s);
}

int count(STACK *s) {
  int height = 0;

  STACK temp;
  temp.top = 0;
  int e;

  while (pop(&e, s) == 0) {
    height += e;
    push(e, &temp);
  }

  while (pop(&e, &temp) == 0)
    push(e, s);

  return height;
}

int equalStacks(STACK *s1, STACK *s2, STACK *s3) {
  int h1 = count(s1);
  int h2 = count(s2);
  int h3 = count(s3);

  while (h1 != h2 || h1 != h3 || h2 != h3) {
    int e1, e2, e3;

    peek(&e1, s1);
    peek(&e2, s2);
    peek(&e3, s3);

    if (h1 >= h2 && h2 >= h3) {
      pop(&e1, s1);
    }
    else if (h2 >= h1 && h2 >= h3) {
      pop(&e2, s2);
    }
    else {
      pop(&e3, s3);
    }

    h1 = count(s1);
    h2 = count(s2);
    h3 = count(s3);
  }

  return count(s1);
}

```

Line: 138 Col: 1

Upload Code as File □ Test against custom input Run Code Submit Code