#include <assert.h>

#include <limits.h>

#include <math.h>

#include <stdbool.h>

#include <stddef.h>

#include <stdint.h>

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

char\* readline();

char\*\* split\_string(char\*);

// Complete the cutTheSticks function below.

// Please store the size of the integer array to be returned in result\_count pointer. For example,

// int a[3] = {1, 2, 3};

//

// \*result\_count = 3;

//

// return a;

//

int ans\_arr[10005];

int\* cutTheSticks(int arr\_count, int\* arr, int\* result\_count) {

int i,j,min;

int count;

ans\_arr[0] = arr\_count;

int counter;

counter =0;

min = 100000;

for(i=0;i<arr\_count;i++)

{

count = 0;

/\* if(ans\_arr[i] == 1)

{

break; // if only one stick left end prog

}\*/

min =100000;

for(j=0;j<arr\_count;j++)

{

if(arr[j]!=0 && arr[j] <=min)

{

min = arr[j];

// counter ++;

}

}

printf("#########%d",min);

printf("\n");

for(j=0;j<arr\_count;j++)

{

if(arr[j]!=0 )

{

arr[j] = arr[j] - min; // to subtract min from all array elements

}

/\* if(arr[j]==min)

{

arr[j] =0;

}

\*/

}

for(j=0;j<arr\_count;j++)

{

if(arr[j]!=0)

{

count++;

}

}

ans\_arr[i+1] = count;

// counter++;

for(j=0;j<arr\_count;j++)

{

printf("%d\t",arr[j]);

}

printf("\n");

}

for(j=0;j<arr\_count;j++)

{

if(ans\_arr[j]!=0)

{

counter++;

}

}

\*result\_count = counter;

return ans\_arr;

}

int main()

{

FILE\* fptr = fopen(getenv("OUTPUT\_PATH"), "w");

char\* n\_endptr;

char\* n\_str = readline();

int n = strtol(n\_str, &n\_endptr, 10);

if (n\_endptr == n\_str || \*n\_endptr != '\0') { exit(EXIT\_FAILURE); }

char\*\* arr\_temp = split\_string(readline());

int\* arr = malloc(n \* sizeof(int));

for (int i = 0; i < n; i++) {

char\* arr\_item\_endptr;

char\* arr\_item\_str = \*(arr\_temp + i);

int arr\_item = strtol(arr\_item\_str, &arr\_item\_endptr, 10);

if (arr\_item\_endptr == arr\_item\_str || \*arr\_item\_endptr != '\0') { exit(EXIT\_FAILURE); }

\*(arr + i) = arr\_item;

}

int arr\_count = n;

int result\_count;

int\* result = cutTheSticks(arr\_count, arr, &result\_count);

for (int i = 0; i < result\_count; i++) {

fprintf(fptr, "%d", \*(result + i));

if (i != result\_count - 1) {

fprintf(fptr, "\n");

}

}

fprintf(fptr, "\n");

fclose(fptr);

return 0;

}

char\* readline() {

size\_t alloc\_length = 1024;

size\_t data\_length = 0;

char\* data = malloc(alloc\_length);

while (true) {

char\* cursor = data + data\_length;

char\* line = fgets(cursor, alloc\_length - data\_length, stdin);

if (!line) { break; }

data\_length += strlen(cursor);

if (data\_length < alloc\_length - 1 || data[data\_length - 1] == '\n') { break; }

size\_t new\_length = alloc\_length << 1;

data = realloc(data, new\_length);

if (!data) { break; }

alloc\_length = new\_length;

}

if (data[data\_length - 1] == '\n') {

data[data\_length - 1] = '\0';

}

data = realloc(data, data\_length);

return data;

}

char\*\* split\_string(char\* str) {

char\*\* splits = NULL;

char\* token = strtok(str, " ");

int spaces = 0;

while (token) {

splits = realloc(splits, sizeof(char\*) \* ++spaces);

if (!splits) {

return splits;

}

splits[spaces - 1] = token;

token = strtok(NULL, " ");

}

return splits;

}