#include <assert.h>

#include <ctype.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\* ltrim(char\*);

char\* rtrim(char\*);

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

int parse\_int(char\*);

/\*

\* Complete the 'diagonalDifference' function below.

\*

\* The function is expected to return an INTEGER.

\* The function accepts 2D\_INTEGER\_ARRAY arr as parameter.

\*/

int diagonalDifference(int arr\_rows, int arr\_columns, int\*\* arr) {

int i,j;

int sum1;

int sum2;

sum1 =0;

sum2=0;

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

{

sum1 = sum1 + arr[i][i];

}

//finding sum2

int k;

k=arr\_columns-1;

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

{

sum2 = sum2 + arr[i][k];

k= k-1;

}

int diff;

diff = sum1-sum2;

if((sum1-sum2)>0)

{

return diff;

}else{

return diff\*-1;

}

}

int main()

{

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

int n = parse\_int(ltrim(rtrim(readline())));

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

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

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

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

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

int arr\_item = parse\_int(\*(arr\_item\_temp + j));

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

}

}

int result = diagonalDifference(n, n, arr);

fprintf(fptr, "%d\n", result);

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;

}

alloc\_length <<= 1;

data = realloc(data, alloc\_length);

if (!data) {

data = '\0';

break;

}

}

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

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

data = realloc(data, data\_length);

if (!data) {

data = '\0';

}

} else {

data = realloc(data, data\_length + 1);

if (!data) {

data = '\0';

} else {

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

}

}

return data;

}

char\* ltrim(char\* str) {

if (!str) {

return '\0';

}

if (!\*str) {

return str;

}

while (\*str != '\0' && isspace(\*str)) {

str++;

}

return str;

}

char\* rtrim(char\* str) {

if (!str) {

return '\0';

}

if (!\*str) {

return str;

}

char\* end = str + strlen(str) - 1;

while (end >= str && isspace(\*end)) {

end--;

}

\*(end + 1) = '\0';

return str;

}

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;

}

int parse\_int(char\* str) {

char\* endptr;

int value = strtol(str, &endptr, 10);

if (endptr == str || \*endptr != '\0') {

exit(EXIT\_FAILURE);

}

return value;

}