#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 sockMerchant function below.

int sockMerchant(int n, int ar\_count, int\* ar) {

int left\_socks[ar\_count];

int i,j;

int pairs;

int count;

//initialize

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

{

left\_socks[i] = -1;

}

int k;

k=0;

pairs =0;

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

{//for number of socks

count = 0;

for(j=0;j<i;j++)

{//for matching if that color single sock is left

if(ar[i]==left\_socks[j])

{

left\_socks[j] = -1;

pairs = pairs + 1;

break;

}

else {

count++;

}

}

if(count >=i)

{

left\_socks[k] = ar[i];

k++;

}

}

return pairs;

}

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\*\* ar\_temp = split\_string(readline());

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

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

char\* ar\_item\_endptr;

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

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

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

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

}

int ar\_count = n;

int result = sockMerchant(n, ar\_count, ar);

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; }

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;

}