**/\*ES 4 ESAME 03/02/14:**

Si definisca una funzione int check(int a[], int dim) che restituisce il valore di verit`a della seguente formula:

∃j ∈ [1, dim − 1).#{i|i ∈ [0, j) ∧ a[j] < a[i]} = #{i|i ∈ [j + 1, dim) ∧ a[j] > a[i]} \*/

#include<stdio.h>

int check(int a[], int dim){

int i, j=1, conta1=0, conta2=0, trovato=0;

while(j<dim-1 && trovato==0){

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

if(a[i]>a[j]) conta1++;

}

for(i=j+1; i<dim; i++){

if(a[i]<a[j]) conta2++;

}

if(conta1==conta2) trovato=1;

else{

conta1=0;

conta2=0;

j++;

}

}

return trovato;

}

int check1(int a[], int dim){ //soluzione del prof

int trovato=0, j=1, i, conta\_prec, conta\_succ;

while(j<dim-1 && !trovato){

conta\_prec=0;

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

if(a[j]<a[i]) conta\_prec++;

conta\_succ=0;

for (i=j+1; i<dim; i++)

if(a[j]>a[i]) conta\_succ++;

trovato=conta\_prec==conta\_succ;

j++;

}

return trovato;

}

int main(){

int a[100], i=0, dima=0;

scanf("%d", &a[i]);

while(a[i]!=0){

i++;

scanf("%d", &a[i]);

}

dima=i;

for(i=0; i<dima; i++) printf("%d ", a[i]);

printf("\n");

printf("%d\n",check(a,dima));

printf("%d\n",check1(a,dima));

return 0;

}

**/\*ES 4 ESAME 03/06/14:**

Si definisca una funzione int check(int a[], int dim) che restituisce il valore di verit`a della seguente formula:

∃j ∈ [1, dim − 1).(∀i ∈ [0, j). a[j] < a[i]) ∨ (∀i ∈ [j + 1, dim). a[j] > a[i]) \*/

#include<stdio.h>

int check(int a[], int dima){

//oppure se cerco il massimo poi devo confrontare che gli elementi che vengono dopo di esso fino alla fine siano minori

int i=0, minimo=a[1], posmin=1, conta=0;

for(i=1; i<dima-1; i++){

if(a[i]<minimo){

minimo=a[i];

posmin=i;

}

}

i=0;

while(i<posmin){

if(a[i]>minimo){

conta++;

i++;

}

}

if(conta==i) return 1;

return 0;

}

int check1(int a[], int dim){ //soluzione del prof

int trovato=0, j=1, i=0, ok=1;

while(j<dim-1 && !trovato){

i=0;

ok=1;

while(i<j && ok)

if(a[i]<=a[j]) ok=0;

else i++;

if(!ok){

i=j+1;

ok=1;

while(i<dim && ok)

if(a[i]>=a[j]) ok=0;

else i++;

}

trovato=ok;

j++;

}

return trovato;

}

int main(){

int a[100], i=0, dima=0;

scanf("%d", &a[i]);

while(a[i]!=0){

i++;

scanf("%d",&a[i]);

}

dima=i;

for(i=0; i<dima; i++) printf("%d ", a[i]);

printf("\n");

printf("%d\n",check(a,dima));

printf("%d\n",check1(a,dima));

return 0;

}

**/\*ES 4 ESAME 25/06/14**:

Scrivere una funzione int check(int a[], int dima) che restituisca il valore di verita' della seguente formula: Esiste j appartenente all'intervallo (0,dim-1) per cui valga che la # {di k appartenente all'intervallo [0,j) & a[k]=a[j]}=1 & la # {di k appartenente a (j,dim) & a[k]=a[j]}=1. \*/

#include<stdio.h>

#include<stdlib.h>

/\*int check(int a[], int dima){

int trovato=0, j=1;

while(j<dima-1 && trovato==0){

int k=0, conta=0;

while(k<j && conta<2){

if(a[k]==a[j]){

conta++;

}

k++;

}

if(conta==1){

conta=0, k=j+1;

while(k<dima && conta<2){

if(a[k]==a[j]){

conta++;

}

k++;

}

if(conta==1)

trovato=1;

else if(conta!=1)

j++;

}

else if(conta!=1)

j++;

}

printf("%d\n",trovato);

return trovato;

}\*/ //oppure invece di usare i 2 while che si ripetono definiamo una funzione d'ausilio

int conta(int n, int a[], int inizio, int fine){

int conta=0;

while(inizio<=fine && conta<2){

if(a[inizio]==n)

conta++;

inizio++;

}

return conta;

}

int check(int a[], int dima){

int trovato=0, j=1;

while(j<dima-1 && trovato==0){

int c=conta(a[j],a,0,j-1);

if(c==1){

c=conta(a[j],a,j+1,dima-1);

if(c==1)

trovato=1;

else

j++;

}

else

j++;

}

printf("%d\n",trovato);

return trovato;

}

int main(){

int \*arr=malloc(sizeof(int)),i=0,dim=1;

scanf("%d",&arr[i]);

while(arr[i]!=0){

i++;

scanf("%d",&arr[i]);

dim++;

}

//for(i=0; i<dim-1; i++) printf("%d\n",arr[i]);

check(arr,dim-1);

return 0;

}

**/\*ES 4 ESAME 14/07/14:**

Si definisca una funzione int check(int a[], int b[], int dima, int dimb) che, dati due array di interi e le loro dimensioni, restituisce il valore di verit`a della seguente formula: ∃j ∈[0, dima).(#{i|i ∈[0, dimb) ∧ a[j]=b[i]}=3). \*/

#include<stdio.h>

int check(int a[], int b[], int dima, int dimb){

int j=0, i=0, trovato=0, conta;

while(j<dima && trovato==0){

conta=0;

for(i=0; i<dimb; i++){

if(a[j]==b[i]) conta++;

}

if(conta==3) trovato=1;

else j++;

}

return trovato;

}

int main(){

int a[100], b[100], i=0, dima=0, dimb=0;

scanf("%d", &a[i]);

while(a[i]!=0){

i++;

scanf("%d",&a[i]);

}

dima=i, i=0;

scanf("%d", &b[i]);

while(b[i]!=0){

i++;

scanf("%d",&b[i]);

}

dimb=i;

for(i=0; i<dima; i++) printf("%d ",a[i]);

printf("\n");

for(i=0; i<dimb; i++) printf("%d ",b[i]);

printf("\n");

printf("%d\n",check(a,b,dima,dimb));

return 0;

}

**/\*ES 3 ESAME 05/09/14:**

Si definisca una funzione int check(int a[], int dim) che, dato un array di interi e la sua dimensione, controlla che ogni valore dell’array, tranne l’ultimo, sia seguito nell’array da almeno un elemento strettamente minore. Si giochi d’astuzia!. \*/

#include<stdio.h>

/\* //questa non va bene perche' controlla che tutti i successori siano minori

int check(int a[], int dim){

int i=0, trovato=0;

while(i<dim-1 && trovato==0){

if(a[i]<=a[i+1]) trovato=1;

else if(a[i]>a[i+1]) i++;

}

return !trovato;

} \*/

int check(int a[], int dim){

int i=0, j=0, found=0, trovato=0, conta=0;

while(i<dim-1 && j<dim && found==0){

j=i+1;

if(a[i]>a[j]) i++;

else if(a[i]<=a[j]){

if(j==dim-1)

found=1;

else{

while(j<dim && trovato==0){

if(a[i]>a[j]) trovato=1;

else if(a[i]<=a[j]){

conta++;

j++;

}

}

if(trovato==1){

trovato=0;

i++;

}

else if(trovato==0 && conta==j-i-1) found=1;

}

}

}

return !found;

}

//giocando d'astuzia

int check1(int a[], int dim){

int i, index\_min=0;

for(i=1; i<dim; i++)

if(a[i]<a[index\_min])

index\_min=i;

return (index\_min==dim-1);

}

int main(){

int a[10], i;

for(i=0; i<10; i++) scanf("%d", &a[i]);

printf("%d\n", check(a, 10));

printf("%d\n", check1(a, 10));

return 0;

}

**/\*ES 4 ESAME 6/11/14:**

Scrivere una funzione int check(int a[], int b[], int dima, int dimb) che, dati due array di interi a e b, di dimensione dima e dimb rispettivamente, restituisce il seguente valore di verita': per ogni i appartenente a [0,dima) tale che (#{j|j appartiene a [0,dimb) & a[i]=b[j]}=0 oppure #{j|j appartiene a [0,dimb) & a[i]=b[j]}>=2. \*/

#include<stdio.h>

int check(int a[], int b[], int dima, int dimb){

int i=0, j=0, conta=0;

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

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

if(a[i]==b[j])

conta++;

if(conta==1) return 0;

else if(conta!=1) return 1;

}

int main(){

int a[100], b[100], i=0, j=0, dima=0, dimb=0;

scanf("%d",&a[i]);

while(a[i]!=0){

i++;

scanf("%d",&a[i]);

dima++;

}

scanf("%d",&b[j]);

while(b[j]!=0){

j++;

scanf("%d",&b[j]);

dimb++;

}

for(i=0; i<dima; i++) printf("%d\n",a[i]);

for(j=0; j<dimb; j++) printf("%d\n",b[j]);

printf("%d\n",check(a,b,dima,dimb));

return 0;

}

**/\*ES 5 ESAME 6/11/14:**

Scrivere una funzione int foo(int a[], int dima) che, dato un array di interi a, di dimensione dima, restituisce il numero di elementi in a che precedono la seconda occorrenza del primo elemento ripetuto piu' volte in a. La funzione restituisce 0 se ogni elemento occorre in a esattamente una volta. es a={60,20,30,10,20,10,20,30}=4, a={60,20,30,10,50,70,40,80}=0. \*/

#include<stdio.h>

int foo(int a[], int dim){

int i, j, conta=0, max=0, el=0, ris=0;

for(i=0; i<dim; i++){

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

if(a[i]==a[j])

conta++;

if(conta>max){

max=conta;

el=a[i];

}

conta=0;

}

// printf("%d\n",max);

// printf("%d\n",el);

if(max==1 || max==0) return 0;

i=0, j=1;

while(j<=2){

if(a[i]==el)

j++;

ris++;

i++;

}

ris--;

return ris;

}

int main(){

int a[100], i=0, dima=0;

scanf("%d", &a[i]);

while(a[i]!=0){

i++;

scanf("%d",&a[i]);

}

dima=i;

for(i=0; i<dima; i++) printf("%d ", a[i]);

printf("\n");

printf("%d\n",foo(a,dima));

return 0;

}

**/\*ES 4 ESAME 12/01/15:**

Scrivere una funzione int check(int a[], int dim) che, dato un array a di interi e la sua dimensione dim, restituisce il seguente valore di verita': per ogni i appartenente [0,dim-1) per cui vale che esiste j appartenente [i+1,dim) e a[i]>a[j]. HINT: La proprieta' e' soddisfatta solo se il valore minimo dell'array occorre una sola volta e in ultima posizione. \*/

#include<stdio.h>

/\*int check(int a[], int dim){

int i, j, trovato=0;

for(i=0; i<dim-1; i++){

trovato=0;

j=i+1;

while(j<dim && trovato==0){

if(a[i]>=a[j])

trovato=1;

else if(a[i]<a[j])

j++;

}

if(trovato==0) return 0;

}

return 1;

} \*/

/\* //oppure

int check(int a[], int dim){

int i=0, trovato=0, conta=0;

while(i<dim && trovato==0){

if(a[i]<a[dim-1])

trovato=1;

else{

if(a[i]==a[dim-1]){

conta++;

i++;

}else if(a[i]>a[dim-1])

i++;

}

}

if(trovato==1) return 0;

else{

if(trovato==0 && conta==1)

return 1;

else if(trovato==1 && conta!=1)

return 0;

}

} \*/

int check(int a[], int dim){ //soluzione proposta dal prof

int posmin=0, occmin=1, j;

for(j=1; j<dim; j++){

if(a[j]==a[posmin])

occmin++;

else if(a[j]<a[posmin]){

posmin=j;

occmin=1;

}

}

// printf("%d\n", posmin);

// printf("%d\n", occmin);

// if(posmin==dim-1 && occmin==1) return 1;

// return 0;

return (posmin==dim-1 && occmin==1);

}

int main(){

int arr[100], i=0, dim=0;

scanf("%d", &arr[i]);

while(arr[i]!=0){

i++;

scanf("%d", &arr[i]);

}

dim=i;

for(i=0; i<dim; i++) printf("%d ", arr[i]);

printf("\n");

printf("%d\n",check(arr,dim));

return 0;

}

**/\*ES 4 ESAME 09/02/15:**

Scrivere una funzione int check(int a[], int dim) che, dato un array a di interi e la sua dimensione dim, restituisce il seguente valore di verita': #{i|i appartiene a [0,dim-1) & (esiste j appartenente a [i+1,dim) per cui a[i]=a[j])}>=2.\*/

#include<stdio.h>

#include<stdlib.h>

/\*int check(int a[], int dim){

int i=0, j=0, conta=0, trovato=0;

while(i<dim && trovato==0){

conta=0;

for(j=0; j<dim; j++){

if(a[i]==a[j])

conta++;

}

if(conta>=2){

trovato=1;

}

else if(conta==0 || conta==1){

i++;

}

}

if(trovato==1)

return 1;

else if(trovato==0)

return 0;

} \*/

int check(int a[], int dim){ //soluzione del prof.

int conta=0, i=0, esiste=0, j=i+1;

while(i<dim-1 && conta<2){

while(j<dim && esiste==0){

if(a[i]==a[j])

esiste=1;

else if(a[i]!=a[j])

j++;

}

if(esiste==1)

conta++;

i++;

}

return conta>=2;

}

int main(){

int \*a=malloc(100\*sizeof(int)),i=0, dim=0;

scanf("%d",&a[i]);

while(a[i]!=0){

i++;

scanf("%d",&a[i]);

dim++;

}

//for(i=0; i<dim; i++) printf("%d\n",a[i]);

printf("%d\n",check(a,dim));

return 0;

}

**/\*ES 5 ESAME 23/06/15:**

Scrivere una funzione che, dati due array di interi a e b e le loro dimensioni dima e dimb, restituisce il seguente valore di verita': se I non e' vuoto restituisce min I dove I={i|i appartiene a [0,dima) & #{j|j appartiene a [0,dimb) & a[i]=b[j]}mod2=0}, altrimenti se I e' vuoto restituisce dima. \*/

#include<stdio.h>

int check(int a[], int b[], int dima, int dimb){

int i, j, arr[dima], min=0, conta=0;

if(dimb==0) return dima;

else if(dimb!=0){

for(i=0; i<dima; i++){

for(j=0; j<dimb; j++){

if(a[i]==b[j])

conta++;

}

arr[i]=conta;

//printf("%d\n",conta);

conta=0;

}

min=a[0];

for(i=0; i<dima; i++){

if(arr[i]%2==0 && a[i]<=min) //VARIANTE: restituisce min dell'a[i] se la # degli elementi in comune sono dispari

min=a[i];

}

if(arr[i]%2!=0 && a[0]==min)

return dima;

else

return min;

}

}

int main(){

int a[100], b[100], i=0, dima=0, dimb=0;

scanf("%d", &a[i]);

while(a[i]!=0){

i++;

scanf("%d", &a[i]);

}

dima=i;

i=0;

scanf("%d", &b[i]);

while(b[i]!=0){

i++;

scanf("%d", &b[i]);

}

dimb=i;

for(i=0; i<dima; i++) printf("%d ", a[i]);

printf("\n");

for(i=0; i<dimb; i++) printf("%d ", b[i]);

printf("\n");

printf("%d\n",check(a,b,dima,dimb));

return 0;

}

**/\*ES 5 ESAME 15/07/15:**

Scrivere una funzione che, dati due array di interi a e b e le loro dimensioni dima e dimb, restituice il seguente valore di verita': esiste i appartenente a [0, dima) tale che #{j|j appartiene a [0, dimb) & a[i]=b[j]}>#{k|k appartiene a [0, dima) & a[i]=a[k]}. \*/

#include<stdio.h>

#include<stdlib.h>

/\*int check(int a[], int b[], int dima, int dimb){

int i=0, j=0, conta=0, comune=0, max=0, max1=0, \*arr=malloc(dima\*sizeof(int)), \*arr1=malloc(dima\*sizeof(int));

for(i=0; i<dima; i++){

for(j=0; j<dimb; j++){

if(a[i]==b[j])

conta++;

}

arr[i]=conta;

conta=0;

}

for(i=0; i<dima; i++){

if(arr[i]>max)

max=arr[i];

}

for(i=0; i<dima; i++){

for(j=0; j<dima; j++){

if(a[i]==a[j])

comune++;

}

arr1[i]=comune;

comune=0;

}

for(i=0; i<dima; i++){

if(arr1[i]>max1)

max1=arr1[i];

}

if(max1>max)

return 0;

else

return 1;

} \*/

int check(int a[], int b[], int dima, int dimb){ //soluzione del prof.

int i=0, trovato=0, conta\_a=0, conta\_b=0, j=0, k=0;

while(i<dima && trovato==0){

for(j=0; j<dimb; j++){

if(a[i]==b[j])

conta\_b++;

}

for(k=0; k<dima; k++){

if(a[i]==a[k])

conta\_a++;

}

trovato=conta\_b>conta\_a;

i++;

}

return trovato;

}

int main(){

int \*a=malloc(100\*sizeof(int)), \*b=malloc(100\*sizeof(int)), i=0, j=0, dima=0, dimb=0;

scanf("%d",&a[i]);

while(a[i]!=0){

i++;

scanf("%d",&a[i]);

dima++;

}

scanf("%d",&b[j]);

while(b[j]!=0){

j++;

scanf("%d",&b[j]);

dimb++;

}

//for(i=0; i<dima; i++) printf("%d\n",a[i]);

//for(j=0; j<dimb; j++) printf("%d\n",b[j]);

printf("%d\n",check(a,b,dima,dimb));

return 0;

}

**/\*ES 4 ESAME 2/11/15:**

Scrivere una funzione con intestazione int check(int a[], int dim, int n, int m) che, dati un array, la sua dimensione dim e due interi n e m, restituisce 1 se il valore massimo in a occorre nell'array piu' di n volte e meno di m volte, 0 altrimenti. N.B.: Non e' consentito utilizzare comandi come break e goto, ne' utilizzare il comando return all'interno di un ciclo while e for.

#include<stdio.h>

/\*int check(int a[], int dim, int n, int m){

int max=a[0],i,conta=0;

for(i=1; i<dim; i++){

if(a[i]>=max){

max=a[i];

}

}

for(i=0; i<dim; i++){

if(a[i]==max){

conta++;

}

}

if(conta>n && conta<m) return 1;

else return 0;

} \*/

int check(int a[], int dim, int n, int m){ //soluzione proposta dal prof

int i, massimo=a[0], occ\_massimo=1;

for(i=1; i<dim; i++){

if(a[i]>massimo){

massimo=a[i];

occ\_massimo=1;

}

else if(massimo==a[i])

occ\_massimo=occ\_massimo+1;

}

return (occ\_massimo>n && occ\_massimo<m);

}

int main(){

int arr[100],i=0,dima=1,valmin,valmax;

scanf("%d",&arr[i]);

while(arr[i]!=0){

i++;

scanf("%d",&arr[i]);

}

dima=i;

for(i=0; i<dima; i++) printf("%d\n",arr[i]);

printf("Leggi il valore minimo di occorrenza: \n");

scanf("%d",&valmin);

printf("Leggi il valore massimo di occorrenza: \n");

scanf("%d",&valmax);

printf("%d\n",check(arr,dima-1,valmin,valmax));

return 0;

}

**/\*ES 5 ESAME 2/11/15:**

Scrivere una funzione con intestazione int maxocc(int a[], int dim) che restituisce il valore che occorre nell'array il maggior numero di volte. Nel caso siano piu' di uno i valori che occorrono il maggior numero di volte, la funzione restituisce quello che occorre per primo. Ad esempio se dato l'array {60,20,30,10,20,40,50,80} la funzione deve restituire il valore 20. Dato invece l'array {60,20,30,10,20,40,60,80} la funzione deve restituire il valore 60. N.B.: Non e' consentito utilizzare comandi come break e goto, ne' utilizzare il comando return all'interno di un ciclo while e for.

#include<stdio.h>

int maxocc(int a[], int dim){

int \*ar=calloc(dim,sizeof(int)),i=0,j=0,conta=0,max,trovato=0,risultato=0;

for(i=0; i<dim; i++){

for(j=0; j<dim; j++){

if(a[i]==a[j]){

conta++;

}

}

ar[i]=conta;

conta=0;

}

//for(i=0; i<dim; i++) printf("%d\n",ar[i]);

i=0, max=ar[0];

for(i=1; i<dim; i++){

if(ar[i]>max){

max=ar[i];

}

}

conta=0;

for(i=0; i<dim; i++){

if(ar[i]==max)

conta++;

}

if(conta!=1){

i=0;

while(i<dim && trovato==0){

if(ar[i]==max){

trovato=1;

risultato=i;

}

else

i++;

}

//if(trovato==1)

//printf("%d\n",a[risultato]);

}

return risultato;

}

int main(){

int arr[100],i=0,dima=1;

scanf("%d",&arr[i]);

while(arr[i]!=0){

i++;

scanf("%d",&arr[i]);

}

dima=i;

//for(i=0; i<dima; i++) printf("%d\n",arr[i]);

printf("%d\n",arr[maxocc(arr,dima-1)]);

return 0;

}

**/\*ES 4 ESAME 14/01/16:**

Scrivere una funzione che, dato un array a di dimensione dim, rstituisce il seguente valore di verita': la #{j|j appartiene a [0,dim) & a[j]<=0} > #{j|j appartiene [0,dim) & a[j]>0}. \*/

#include<stdio.h>

#include<stdlib.h>

/\*int check(int a[], int dim){

int i=0, contaneg=0, contapos=0;

for(i=0; i<dim; i++){

if(a[i]<=0)

contaneg++;

else if(a[i]>0)

contapos++;

}

if(contaneg>contapos)

return 1;

else if(contaneg<=contapos)

return 0;

} \*/

int check(int a[],int dim){ //correzione proposta

int contapos=0, j;

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

if(a[j]>0)

contapos++;

return (dim>2\*contapos);

}

int main(){

int \*arr=malloc(100\*sizeof(int)),dim=1,i=0;

scanf("%d",&arr[i]);

while(arr[i]!=0){

i++;

scanf("%d",&arr[i]);

dim++;

}

//for(i=0; i<dim-1; i++) printf("%d\n",arr[i]);

printf("%d\n",check(arr,dim-1));

return 0;

}

**/\*ES 4 ESAME 4/02/16:**

Scrivere una funzione int check(int a[], int dima, int b[], int dimb) che, dati due array di interi a, b e le loro dimensioni dima, dimb, restituisce il seguente valore di verita': esiste i appartenente all'intervallo [0,dima) per cui vale che ((per ogni j appartenente a [0,dimb).a[i]!=b[j]) & #{k|k appartiene a [0,dima) & a[i]=a[k]}=1). \*/

#include<stdio.h>

int contadiversi(int a[], int init, int dim, int elem){ //funzione che verifica la presenza di un valore in un array

int i=init, trovato=0;

while(i<dim && trovato==0){

if(a[i]==elem) trovato=1;

i++;

}

return trovato;

}

int check(int a[], int dima, int b[], int dimb){

int i;

for(i=0; i<dima; i++){

if(contadiversi(a, i+1, dima, a[i])==0){

if(contadiversi(b, 0, dimb, a[i])==0) return 1;

}

}

return 0;

}

/\* // oppure soluzione proposta dal prof

int check(int a[], int dima, int b[], int dimb){

int i=0, trovato=0, contaocc=0, j=0;

while(i<dima && trovato==0){

if(contadiversi(b,0,dimb,a[i])==0){ //oppure if(!contadiversi(..));

while(j<dima && contaocc<2){

if(a[j]==a[i]) contaocc++;

j++;

}

trovato=(contaocc==1);

}

i++;

}

return trovato;

} \*/

int main(){

int a[100], b[100], dima=0, dimb=0, n=0;

scanf("%d", &a[n]);

while(a[n]!=0){

n++;

scanf("%d", &a[n]);

}

dima=n;

n=0;

scanf("%d", &b[n]);

while(b[n]!=0){

n++;

scanf("%d", &b[n]);

}

dimb=n;

for(n=0; n<dima; n++) printf("%d ", a[n]);

printf("\n");

for(n=0; n<dimb; n++) printf("%d ", b[n]);

printf("\n");

printf("%d\n",check(a,dima,b,dimb));

return 0;

}

**/\*ES 5 ESAME 8/06/16:**

Scrivere una funzione che, dato un array di interi a e la sua dimensione dim restituisce il seguente valore di verita': esiste i tale che i appartiene a [0,dim) & per ogni j apprtenente [0,i) vale che a[j]>a[i] & #{k|k appartiene a [i+1, dim) & a[k]>[i]}=1. \*/

#include<stdio.h>

#include<stdlib.h>

/\*int cerca(int a[], int dim){

int j=0, i=j+1, k=dim, conta=0, trovato=0, ifin=0;

while(i<dim && trovato==0){

if(a[i]<a[j]){

i++;

j++;

}

else if(a[i]>=a[j]){

ifin=i-1;

trovato=1;

}

}

if(trovato==0)

return 0;

else if(trovato==1){

while(a[ifin]<k){

if(a[ifin]<a[k-1]){

conta++;

k--;

}

else if(a[ifin]>=a[k-1])

k--;

}

if(conta==1)

return 1;

else

return 0;

}

} \*/

int cerca(int a[], int dima){ //soluzione proposta dal prof

int true=1, false=0; //se non le dichiaro cosi' crede che non siano mai dichiarate poi

int i=0, trovato=0, j=0, tuttimaggiori=true, k=i+1, conta=0;

while(i<dima && trovato==0){

while(j<i && tuttimaggiori){

if(a[j]>a[i])

j++;

else

tuttimaggiori=false;

}

if(tuttimaggiori){ //if(tuttimaggiori=true)

while(k<dima && conta<2){

if(a[k]>a[i])

conta++;

k++;

}

if(conta==1)

trovato=1;

}

i++;

}

return trovato;

}

int main(){

int \*a=(int \*)malloc(100\*sizeof(int)), i=0,dim=0;

scanf("%d",&a[i]);

while(a[i]!=-1){

i++;

scanf("%d",&a[i]);

dim++;

}

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

printf("%d\n",a[i]);

printf("%d\n",cerca(a,dim));

return 0;

}

**/\*ES 1 ESAME 31/10/16:**

Scrivere una funzione che, dato un array a di dimensione dim, restituisce il seguente valore di verita':

#{j|j appartiene [0,dim-1) && per ogni i|i appartiene a [j+1,dim) allora a[i]!=a[j]}=dim-1, ovvero che TUTTI gli elementi dell'array siano tra loro diversi. \*/

#include<stdio.h>

#include<stdlib.h>

int funzione(int a[], int dim){

int i, j=0, conta=0, temp=0;

for(j=0; j<dim; j++){

conta=0;

for(i=0; i<dim; i++){

if(a[j]!=a[i]) conta++;

}

if(conta==dim-1) temp++;

}

if(temp==dim) return 1;

else return 0;

}

int main(){

int dima, \*a, i;

scanf("%d", &dima);

a=(int \*)malloc(dima\*sizeof(int));

for(i=0; i<dima; i++) scanf("%d", &a[i]);

printf("%d\n", funzione(a, dima));

return 0;

}

**/\*ES 2 ESAME 31/10/16:**Scrivere una funzione int contaUnico(int a[], int b[], int dima, int dimb) che, dati due array a e b di dimensioni rispettivamente dima e dimb, restituisce il numero di elementi diversi di a che sono contenuti in b.

Input: 8//dima 60 20 30 10 20 40 60 80 6//dimb 60 30 30 10 20 60 Output:4//valori 60 20 30 10 di b appartenenti ad a \*/

#include<stdio.h>

#include<stdlib.h>

int conta(int elem, int a[], int start, int end){

int i, conta=0;

for(i=start; i<end; i++)

if(a[i]==elem) conta++;

return conta;

}

int contaUnico(int a[], int b[], int dima, int dimb){

int i, j=0, temp=0;

for(j=0; j<dimb; j++){

if(conta(b[j], b, j, dimb)==1){

if(conta(b[j], a, 0, dima)>0)

temp++;

}

}

return temp;

}

int main(){

int dima, \*a, i, dimb, \*b;

scanf("%d", &dima);

a=(int \*)malloc(dima\*sizeof(int));

for(i=0; i<dima; i++) scanf("%d", &a[i]);

scanf("%d", &dimb);

b=(int \*)malloc(dimb\*sizeof(int));

for(i=0; i<dimb; i++) scanf("%d", &b[i]);

printf("%d\n", contaUnico(a, b, dima, dimb));

return 0;

}

**/\*ES 5 ESAME 6/07/16:**

Scrivere una funzione che, dato un array di interi a e la sua dimensione dim, restituisce il seguente valore di verita': esiste i, j appartenenti a [0,dim) tale che a[i]!=a[j] & #{k|k appartiene a [0,dim) & a[k]=a[i]}=#{h|h appartiene a [0,dim) & a[h]=a[j]}. \*/

#include<stdio.h>

/\*int contaocc(int a[], int dim, int val){ //funzione ausiliaria che conta quante volte e' presente un elemento nell'array

int i, count=0;

for(i=0; i<dim; i++) if(a[i]==val) count++;

return count;

}

int check(int a[], int dim){

int i=0, j=1;

while(j<dim){

if(a[i]!=a[j]){

if(contaocc(a,dim,a[i])==contaocc(a,dim,a[j])) return 1;

}

j++;

}

return 0;

} \*/

/\*int check(int a[], int dim){

int i=0, j=0;

for(i=0; i<dim; i++){

for(j=i+1; j<dim; j++){

if(a[i]!=a[j]){

if(conta(a,dim,a[i])==conta(a,dim,a[j]))

return 1;

}

}

}

return 0;

}\*/

int check(int a[], int dim){ //soluzione proposta dal prof

int i=0, trovato=0, j=i+1, conta=0, k=0;

while(i<dim && trovato==0){

while(j<dim && trovato==0){

if(a[i]==a[j])

j++;

else if(a[i]!=a[j])

trovato=1;

if(trovato==1){

for(k=0; k<dim; k++)

if(a[k]==a[i])

conta++;

else if(a[k]==a[j])

conta--;

if(conta!=0){

trovato=0;

j++;

}

}

}

i++;

}

return trovato;

}

int main(){

int a[100], i=0, dim=0;

scanf("%d", &a[i]);

while(a[i]!=0){

i++;

scanf("%d", &a[i]);

}

dim=i;

for(i=0; i<dim; i++) printf("%d ", a[i]);

printf("\n");

printf("%d\n",check(a,dim));

return 0;

}

/\*ES 5 ESAME 1/09/15. Scrivere una funzione che, dato un array di interi a, la sua dimensione dima ed un intero r, restituisce il seguente valore di verita': (per ogni i, j appartenente a [0, dima) vale che j>i & a[i]=a[j] allora j-i=r). \*/

#include<stdio.h>

/\*int check(int a[], int dima, int r){

int i=0, j=0, trovato=0;

while(i<dima-1){

j=i+1;

while(j<dima){

if(a[i]==a[j] && j-i==r) return 1;

else if(a[i]!=a[j]) j++;

}

i++;

}

return 0;

} \*/

int check(int a[], int dima, int r){ //soluzione del prof

int i=0, trovato=0, j=0;

while(i<dima && trovato==0){

j=i+1;

while(j<dima && trovato==0){

if(a[i]==a[j] && (j-i)!=r)

trovato=1;

else

j++;

}

i++;

}

return (!trovato);

}

int main(){

int a[100], i=0, dima=0, r;

scanf("%d", &a[i]);

while(a[i]!=0){

i++;

scanf("%d",&a[i]);

}

dima=i;

scanf("%d",&r);

for(i=0; i<dima; i++) printf("%d ", a[i]);

printf("\n");

printf("%d\n",check(a,dima,r));

return 0;

}

**/\*ES 5 ESAME 06/09/16:**

Scrivere una funzione che, dato un array a di dimensione dim e un intero m appartenente a [2,dim], restituisce il valore di verita' della seguente formula: esiste i,j appartenente a [0,dim) tale che i<j && per ogni k appartenente a [i,j] tale che a[k]=a[i] && j-i+1>=m. \*/

#include<stdio.h>

int funzione(int a[], int dim, int m){

if(m>=2 && m<=dim){

int i=0, j=i+1, k=i, trovato=0;

while(i<dim && j<dim && trovato==0){

k=i;

a[k]=a[i];

if(j-i+1>=m) trovato=1;

if(a[k]==a[j]) j++;

else if(a[k]!=a[j]){

i++;

j=i+1;

}

}

return trovato;

}

else return 0;

}

int check(int a[], int dim, int m){

int i=0, trovato=0;

while(i<=dim-m && !trovato){

int j=m+i-2;

int k=i+1;

while(k<=j && !trovato){

if(a[k]!=a[i]) trovato=1;

else k++;

}

trovato=!trovato;

i++;

}

return trovato;

}

int main(){

int a[10], i, m;

for(i=0; i<10; i++) scanf("%d", &a[i]);

scanf("%d", &m);

printf("fun%d\n",funzione(a, 10, m));

printf("che%d\n",check(a, 10, m));

return 0;

}

**/\*ES 1 ESAME 19/01/17:**

Si scriva una funzione c che, dato un array a di dimensione dim e un numero naturale n tale che 1<=n<=dim, restituisce il seguente valore di verita': per ogni j appartenente a [n,dim) vale che a[j]=sommatoria di a[i] per i che va da (j-n) a j. Il testo chiede in sostanza di scrivere una procedura che verifica se ogni elemento dell'array a dall'indice n in poi ha un valore pari alla somma degli n elementi precedenti.

Input: 7//dim 1 2 3 6 11 20 37 3//n Output:1 \*/

#include<stdio.h>

#include<stdlib.h>

/\* //seguendo la struttura della formula logica, una prima soluzione corretta prevede l'uso di 2 cicli annidati

int somma(int a[], int dim, int n){

int i, j=n, somma, trovato=0;

while(j<dim && trovato==0){

somma=0;

for(i=j-n; i<j; i++) somma+=a[i];

if(somma==a[j]) j++;

else trovato=1;

}

// if(trovato==1) return 0;

// else return 1;

return !trovato;

} \*/

/\* //oppure in maniera piu' modulare, mediante l'uso di una funzione ausiliaria

int somma(int a[], int init, int end){

int somma=0, k;

for(k=init; k<end; k++) somma+=a[k];

return somma;

}

int check(int a[], int dim, int n){

if(n<1 || n>dim) return 0;

int i;

for(i=n; i<dim; i++) if(a[i]!=somma(a,i-n,i)) return 0;

return 1;

} \*/

//oppure ancora, osservando che ad ogni iterazione del ciclo i valori da sommare sono quasi gli e stessi rispetto a quelli sommati nell'iterazione precedente, una soluzione alternativa piu' efficiente e' la seguente

int check(int a[], int dim, int n){

int i, somma=0;

for(i=0; i<n; i++) somma+=a[i];

for(i=n; i<dim; i++){

if(somma!=a[i]) return 0;

somma=somma-a[i-n]+a[i];

}

return 1;

}

int main(){

int dim, i, \*a, n;

scanf("%d", &dim);

a=(int \*)malloc(dim\*sizeof(int));

for(i=0; i<dim; i++) scanf("%d", &a[i]);

scanf("%d", &n);

// printf("%d\n", somma(a,dim,n));

printf("%d\n", check(a,dim,n));

return 0;

}

**/\*ES 1 ESAME 09/02/17:**

Scrivere una funzione che, dato un array a di dimensione dim, restituisce il seguente valore di verita':

Per ogni i appartenente a [2,dim).a[i]=(sommatoria j:j appartiene a [1,i-1].a[j])+(sommatoria k:k appartiene a [0,i-2].a[k]) \*/

#include<stdio.h>

int check(int a[], int dim){

if(dim<3) return 0;

if(dim==3){

if(a[0]+a[1]==a[2]) return 1;

else return 0;

}else if(dim>3){

int i=2, contak=a[0], contaj=a[1], trovato=0;

while(i<dim && trovato==0){

if(a[i]==contak+contaj){

i++;

contak=contak+a[i-2];

contaj=contaj+a[i-1];

}

else trovato=1;

}

return !trovato;

}

}

int main(){

int a[6], i;

for(i=0; i<6; i++) scanf("%d", &a[i]);

printf("%d\n", check(a, 6));

return 0;

}