#include<bits/stdc++.h>

using namespace std;

const long long maxN = 1e5;

double arr[maxN];

double brr[maxN];

double MyMerge(int Abeg,int Aend,int Bbeg,int Bend){

//每个数组小于等于2个时，直接归并排序求中位数

if(Aend-Abeg<2&&Bend-Bbeg<2){

vector<double> acc;

while (Aend-Abeg>=0||Bend-Bbeg>=0)

{

if(Aend-Abeg>=0&&Bend-Bbeg>=0){

if(arr[Abeg]<brr[Bbeg]){

acc.push\_back(arr[Abeg]);

Abeg++;

}else{

acc.push\_back(brr[Bbeg]);

Bbeg++;

}

}else if(Aend-Abeg>=0){

acc.push\_back(arr[Abeg]);

Abeg++;

}else if(Bend-Bbeg>=0){

acc.push\_back(brr[Bbeg]);

Bbeg++;

}

}

int Vmid = acc.size()/2 + acc.size()%2;

double midNum;

if(acc.size()%2){

midNum = acc[Vmid];

}else{

midNum = (acc[Vmid-1] + acc[Vmid])/2;

}

return midNum;

}

double Amid,Bmid;

//arr

if((Aend+Abeg)%2 ){

Amid = (arr[(Aend+Abeg)/2] + arr[(Aend+Abeg)/2 + 1])/2;

}else{

Amid = arr[(Aend+Abeg)/2];

}

//brr

if((Bend+Bbeg)%2 ){

Bmid = (brr[(Bend+Bbeg)/2] + brr[(Bend+Bbeg)/2 + 1])/2;

}else{

Bmid = brr[(Bend + Bbeg)/2];

}

//选择 中位数小的取后半部分 中位数大的取前半部分;

if((Aend+Abeg)%2&&(Bend+Bbeg%2)){

//如果arr夹在中间

if(arr[(Aend+Abeg)/2]>brr[(Bend+Bbeg)/2] && arr[(Aend+Abeg)/2 + 1]< brr[(Bend+Bbeg)/2 + 1]){

return Amid;

//如果brr夹在中间

}else if(arr[(Aend+Abeg)/2]<brr[(Bend+Bbeg)/2] && arr[(Aend+Abeg)/2 + 1]> brr[(Bend+Bbeg)/2 + 1])

return Bmid;

}

if(Amid<Bmid){

return MyMerge((Aend+Abeg)/2 + (Aend+Abeg)%2,Aend,Bbeg,(Bend+Bbeg)/2);

}else if(Amid>Bmid){

return MyMerge(Abeg,(Aend+Abeg)/2,(Bend+Bbeg)/2 + (Bbeg+Bend)%2,Bend);

}else{

return Amid;

}

}

int main(){

int n;

cin>>n;

for(int i = 0;i<n;i++) cin>>arr[i];

for(int i = 0;i<n;i++) cin>>brr[i];

double ans = MyMerge(0,n-1,0,n-1);

cout<<ans<<endl;

system("pause");

return 0;

}