A Very Easy Question

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

Very soon you will learn the quicksort algorithm. In quicksort we fix a pivot and partition the array around it.We then recurse this with array to the left and to the right of pivot.

Now during partitioning every element gets compared to the pivot, however after partitioning we can be sure that indexes to the right of pivot don't get compared to indexes to the left of pivot. However pivot is chosen by random and that randomizes everything. But still the average time complexity of the algorithm is $O(nlogn)$.

All that you have to do is tell what is the probability that on average element at index i gets compared to element at index j $(i \neq j)$ while performing the quicksort algorithm. Index i, j are the indexes of the final sorted array.

For example if array is $34, 78, 22, 19$. And indexes are $i = 0$, $j = 2$ then in sorted array $19, 22, 34, 78$. You have to find the probability that 19 and 34 get compared during the sort.

Since it is a simulating problem we require around 10^6 runs of quicksort.

Note : We expect you to perform *randomized* quicksort

Input

First line contains a single integer n $(n \leq 20)$ - denoting the number of elements in array.

Second line contains n space separated integers $(1 \leq A[i] \leq 10^9)$ -elements of the array.

Third line contains 2 integers i , j , $(0 \leq (i, j) \leq n - 1), i \neq j$ - the two indexes.

Output

Output a Double number denoting the probability that on average element at index i , j get compared in quicksort.

Example

standard input	standard output

5
5 2 3 4 6
1 3
	0.666667

Note

For c++ users : put srand (time(NULL)); in your int main for random seed to use rand().