Problem Statement for CrazyBot

Problem Statement

An out-of-control robot is placed on a plane, and it takes n random steps. At each step, the robot chooses one of four directions randomly and moves one unit in that direction. The probabilities of the robot choosing north, south, east or west are north, south, east and west percent, respectively.
The robot's path is considered simple if it never visits the same point more than once. (The robot's starting point is always the first visited point.) Return a double containing the probability that the robot's path is simple. For example, "EENE" and "ENW" are simple, but "ENWS" and "WWWWSNE" are not ('E', 'W', 'N' and 'S' represent east, west, north and south, respectively).

Definition

Class:	CrazyBot
Method:	getProbability
Parameters:	int, int, int, int, int
Returns:	double
Method signature:	double getProbability(int n, int east, int west, int south, int north)
	(be sure your method is public)

Notes

Your return must have relative or absolute error less than 1E-9.

Constraints

- n will be between 1 and 14, inclusive.
- east will be between 0 and 100, inclusive.
- west will be between 0 and 100, inclusive.
- south will be between 0 and 100, inclusive.
- north will be between 0 and 100, inclusive.
- The sum of east, west, south and north will be 100.

Examples

0)

1
25
25
25
25
Returns: 1.0
The robot only takes one step, so it never visits a point more than once.

1)
2
25
25
25
25
Returns: 0.75
The robot will visit its starting point twice only if the two moves are in opposite directions.

2)
7
50
0
0
50
Returns: 1.0
Here, the only possible directions are north and east, so the robot will never visit the same point twice.

3)
14
50
50
0
0
Returns: 1.220703125E-4
Here, the only possible directions are east and west. The only two available paths are "EEEEEEEEEEEEEE" and "WWWWWWWWWWWWWW". The probability is equal to 2 / (2^14).

4)
14
25
25
25
25
Returns: 0.008845493197441101
The probability is quite small for n=14.

Problem Statement

A boot shop has received a shipment from the factory consisting of N left boots and N right boots. Each boot has some integer size, and a left and right boot will form a proper pair if they have equal sizes. Each boot can only belong to a single pair. The employees of the boot store want to create N proper pairs of boots. Fortunately, the factory has offered to exchange any number of boots in the shipment with new boots of different sizes.

You are given a int[] left and a int[] right containing the sizes of the left boots and right boots, respectively. Return the least number of boots that must be exchanged.

Definition

Class:	BootsExchange
Method:	leastAmount
Parameters:	int[], int[]
Returns:	int
Method signature:	int leastAmount(int[] left, int[] right) (be sure your method is public)

Constraints

- Each element in left will be between 1 and 1000, inclusive.
- Each element in right will be between 1 and 1000, inclusive.
- left and right will have the same number of elements.
- left will contain between 1 and 50 elements, inclusive.

Examples

0)
{1, 3, 1}
{2, 1, 3}
Returns: 1
They can exchange a size 1 left shoe for a size 2 left shoe, or they can exchange the size 2 right shoe for a size 1 right shoe.

1)
{1, 3}
{2, 2}
Returns: 2
They can exchange both left shoes for size 2 left shoes, or they can exchange the right shoes for a size 1 right shoe and a size 3 right shoe, or they can mix these two possibilities.

2)
{1, 2, 3, 4, 5, 6, 7}
{2, 4, 6, 1, 3, 7, 5}
Returns: 0
Nothing to exchange.