#include<stdio.h>

#include<stdlib.h>

#include<math.h>

typedef struct point

{

float x;

float y;

}Point;

void initializePoint(Point p[], int length)

{

int i;

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

scanf("%f %f", &p[i].x, &p[i].y);

}

float calculate(Point A, Point B)

{

float res;

res = sqrt((float)(((A.x - B.x) \* (A.x - B.x)) + ((A.y - B.y) \* (A.y - B.y))));

return res;

}

void sort(Point p[], int left, int right, int ver)

{

int i, j;

Point temp;

if (ver == 1)

{

for (i = left; i < right; i++)

{

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

{

if (p[i].x > p[j].x)

{

temp = p[i];

p[i] = p[j];

p[j] = temp;

}

}

}

}

else

{

for (i = left; i < right; i++)

{

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

{

if (p[i].y > p[j].y)

{

temp = p[i];

p[i] = p[j];

p[j] = temp;

}

}

}

}

}

float Brute\_force(Point p[], int left, int right)

{

int i,j;

float res = 0;

for (i = left; i < right; i++)

{

if (i == left)

res = calculate(p[i], p[i + 1]);

else if (res > calculate(p[i], p[i + 1]))

res = calculate(p[i], p[i + 1]);

}

return res;

}

float Closset\_pair(Point p[], int left, int right)

{

Point \* stack;

int count = 0;

int q, i;

float dl;

float dr;

float d;

stack = (Point\*)malloc(sizeof(Point)\* (right - left));

sort(p, left, right, 1); // x축기준 sort

if (right - left < 3) // 갯수가 3개이하일때

return Brute\_force(p, left, right);

q = (left + right) / 2;

dl = Closset\_pair(p, left, q);

dr = Closset\_pair(p, q, right);

d = ((dl > dr) ? dr : dl);

for (i = left; i < right; i++)

{

if (p[q].x - d <= p[i].x && p[q].x + d >= p[i].x)

stack[count++] = p[i];

}

sort(stack, 0, count, 2);

for (i = 0; i < count - 1; i++)

{

if (calculate(stack[i], stack[i + 1]) < d)

d = calculate(stack[i], stack[i + 1]);

}

return d;

}

int main(void)

{

Point \* p;

Point \* stack;

int length;

scanf("%d", &length);

p = (Point\*)malloc(sizeof(Point)\* length);

stack = (Point\*)malloc(sizeof(Point)\* length);

initializePoint(p, length);

printf("result %.2f\n",Closset\_pair(p, 0, length));

}