#include <iostream>

#include <string>

#include <sstream>

#include <iomanip>

#include<math.h>

using namespace std;

int main()

{

const int NUM\_SAMPLES = 20000;

string str, str\_value1, str\_value2, str\_value3, str\_value4, str\_value5;

double learning\_rate, start\_a, start\_b, Vmin, Vmax, sigma;

double x, t, y, a0, b0, ga, gb, Ermsd, L;

double arr1[NUM\_SAMPLES], arr2[NUM\_SAMPLES], e[NUM\_SAMPLES], etb;

int z = 0, so\_mau, K, D, Dk, i, num\_iterations, num\_folds;

getline(cin, str);

getline(cin, str);

getline(cin, str);

getline(cin, str); stringstream scin1(str);

scin1 >> str\_value1; scin1 >> num\_iterations;

getline(cin, str); stringstream scin2(str);

scin2 >> str\_value2; scin2 >> learning\_rate;

getline(cin, str); stringstream scin3(str);

scin3 >> str\_value3; scin3 >> start\_a;

getline(cin, str); stringstream scin4(str);

scin4 >> str\_value4; scin4 >> start\_b;

getline(cin, str); stringstream scin5(str);

scin5 >> str\_value5; scin5 >> num\_folds;

getline(cin, str);

getline(cin, str);

getline(cin, str);

//Dem so mau du lieu

so\_mau = 0;

while (getline(cin, str))

{

if (str.empty())continue;

else

{

so\_mau = so\_mau++;

stringstream scin(str);

scin >> x; scin >> t;

arr1[z] = x; arr2[z] = t;

z++;

}

}

//Tinh toan

K = num\_folds;

D = int(so\_mau / K);

Dk = so\_mau - D\*(K - 1);

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

{

double f[10] = { 0,0,0,0,0,0,0,0,0,0 };

a0 = start\_a; b0 = start\_b;

for (int h = 1; h <= num\_iterations; h++)

{

Ermsd = 0;

if (i != K)

{

ga = gb = 0;

for (int j = 0; j < so\_mau; j++)

{

if ((j < (i - 1)\*D) || (j > (i\*D - 1)))

{

ga = ga + (a0\*arr1[j] + b0 - arr2[j])\*arr1[j];

gb = gb + (a0\*arr1[j] + b0 - arr2[j]);

}

}

a0 = a0 - learning\_rate\*ga / (sqrt(ga\*ga + gb\*gb));

b0 = b0 - learning\_rate\*gb / (sqrt(ga\*ga + gb\*gb));

for (int j = (i - 1)\*D; j <= (i\*D - 1); j++)

{

Ermsd = Ermsd + pow((a0\*arr1[j] + b0 - arr2[j]), 2);

}

Ermsd = sqrt(Ermsd / D);

}

else

{

ga = gb = 0;

for (int j = 0; j < (i - 1)\*D; j++)

{

ga = ga + (a0\*arr1[j] + b0 - arr2[j])\*arr1[j];

gb = gb + (a0\*arr1[j] + b0 - arr2[j]);

}

a0 = a0 - learning\_rate\*ga / (sqrt(ga\*ga + gb\*gb));

b0 = b0 - learning\_rate\*gb / (sqrt(ga\*ga + gb\*gb));

for (int j = (i - 1)\*D; j < so\_mau; j++)

{

Ermsd = Ermsd + pow((a0\*arr1[j] + b0 - arr2[j]), 2);

}

Ermsd = sqrt(Ermsd / Dk);

}

}

if (i != K)

{

int dem = 0; double sum = 0;

for (int j = (i - 1)\*D; j <= (i\*D - 1); j++)

{

dem++;

y = a0\*arr1[j] + b0;

e[dem - 1] = y - arr2[j];

sum = sum + e[dem - 1];

}

etb = (sum) / double(dem);

sigma = 0;

for (int j = 0; j < D; j++)

{

sigma = sigma + pow((e[j] - etb), 2);

}

sigma = sqrt(sigma / double(D));

Vmax = double(etb + 3 \* sigma);

Vmin = double(etb - 3 \* sigma);

L = (Vmax - Vmin) / 10;

}

else

{

int dem = 0; double sum = 0;

for (int j = (i - 1)\*D; j < so\_mau; j++)

{

dem++;

y = a0\*arr1[j] + b0;

e[dem - 1] = y - arr2[j];

sum = sum + e[dem - 1];

}

etb = (sum) / double(dem);

sigma = 0;

for (int j = 0; j < Dk; j++)

{

sigma = sigma + pow((e[j] - etb), 2);

}

sigma = sqrt(sigma / double(Dk));

Vmax = double(etb + 3 \* sigma);

Vmin = double(etb - 3 \* sigma);

L = (Vmax - Vmin) / 10;

}

if (i != K)

{

for (int j = 0; j < D; j++)

{

if (e[j] >= Vmin && e[j] <= Vmax)

{

if ((e[j] >= Vmin) && (e[j] < Vmin + L))

{

f[0]++;

}

else if (e[j] < Vmin + 2 \* L)

{

f[1]++;

}

else if (e[j] < Vmin + 3 \* L)

{

f[2]++;

}

else if (e[j] < Vmin + 4 \* L)

{

f[3]++;

}

else if (e[j] < Vmin + 5 \* L)

{

f[4]++;

}

else if (e[j] < Vmin + 6 \* L)

{

f[5]++;

}

else if (e[j] < Vmin + 7 \* L)

{

f[6]++;

}

else if (e[j] < Vmin + 8 \* L)

{

f[7]++;

}

else if (e[j] < Vmin + 9 \* L)

{

f[8]++;

}

else if (e[j] <= Vmax)

{

f[9]++;

}

}

}

}

else

{

for (int j = 0; j < Dk; j++)

{

if (e[j] >= Vmin && e[j] <= Vmax)

{

if ((e[j] >= Vmin) && (e[j] < Vmin + L))

{

f[0]++;

}

else if (e[j] < Vmin + 2 \* L)

{

f[1]++;

}

else if (e[j] < Vmin + 3 \* L)

{

f[2]++;

}

else if (e[j] < Vmin + 4 \* L)

{

f[3]++;

}

else if (e[j] < Vmin + 5 \* L)

{

f[4]++;

}

else if (e[j] < Vmin + 6 \* L)

{

f[5]++;

}

else if (e[j] < Vmin + 7 \* L)

{

f[6]++;

}

else if (e[j] < Vmin + 8 \* L)

{

f[7]++;

}

else if (e[j] < Vmin + 9 \* L)

{

f[8]++;

}

else if (e[j] <= Vmax)

{

f[9]++;

}

}

}

}

double sumf = 0;

for (int j = 0; j < 10; j++)

{

sumf = sumf + f[j];

}

for (int j = 0; j < 10; j++)

{

f[j] = f[j] / sumf;

}

cout << setw(10) << fixed << setprecision(5) << right << a0

<< setw(10) << fixed << setprecision(5) << right << b0;

cout << setw(10) << fixed << setprecision(5) << right << Ermsd;

for (int w = 0; w <= 9; w++)

{

cout << setw(10) << right << f[w];

}

cout << endl;

}

}