The following code was created from this geeksforgeeks [tutorial](https://www.geeksforgeeks.org/regression-analysis-and-the-best-fitting-line-using-c/)

#include "cuda\_runtime.h"  
#include "device\_launch\_parameters.h"  
// C++ program to implement  
#include <iostream>  
#include <cstdio>  
#include <vector>  
using namespace std**;**//Prototypes  
void PrintBestFittingLine(float&**,** float&**,** vector<float>& **,** float&**,** float&**,** float&**,** float&)**;**void squareErraErr(vector<float>**,** vector<float>**,** float**,** float**,** float)**;**float predict(float**,**float&**,** float& )**;**// Driver code  
int main(int argc**,** char\*\* argv)  
{  
 //Variable Declaration  
 vector<float> x**;**vector<float> y**;** //Dynamic Arrays: Contain all (i-th x) and all (i-th y)  
 float coeff = **0;** // Store the coefficient/slope in the best fitting line  
 float constTerm = **0;** // Store the constant term in the best fitting line  
 float sum\_xy = **0;** // Contains sum of product of all (i-th x) and (i-th y)  
 float sum\_x = **0;** // Contains sum of all (i-th x)  
 float sum\_y = **0;** // Contains sum of all (i-th y)  
 float sum\_x\_square = **0;** //Contains sum of square of all (i-th x)  
 float sum\_y\_square = **0;** // Contains sum of square of all (i-th y)  
  
 freopen("input.txt"**,** "r"**,**stdin)**;** // Number of pairs of (xi, yi)  
 // in the dataset  
 int size**;** cin >> size**;** for (int i = **0;** i < size**;** i++) {  
 // In a csv file all the values of  
 // xi and yi are separated by commas  
 char comma**;** float xi**;** float yi**;** cin >> xi >> comma >> yi**;** sum\_xy += xi \* yi**;** sum\_x += xi**;** sum\_y += yi**;** sum\_x\_square += xi \* xi**;** sum\_y\_square += yi \* yi**;** x.push\_back(xi)**;** y.push\_back(yi)**;** }  
  
 PrintBestFittingLine(coeff**,** constTerm**,** x**,** sum\_x\_square**,** sum\_xy**,** sum\_x**,** sum\_y)**;** cout << "Predicted value at 2060 = " << predict(**2060,** coeff**,** constTerm) << endl**;** // Printing the best fitting line  
 squareErraErr(x**,** y**,** coeff**,** constTerm**, 2050**)**;**}  
  
// Function that print the best fitting line  
void PrintBestFittingLine(float& coe**,** float& ct**,** vector<float>& x**,** float& sxs**,** float& sxy**,** float& sx**,** float& sy)  
{  
 if (coe == **0** && ct == **0**) {  
 coe = (x.size() \* sxy - sx \* sy) / (x.size() \* sxs - sx \* sx)**;** // Calculate the coefficient slope of the best fitting line  
 ct = (sy \* sxs - sx \* sxy) / (x.size() \* sxs - sx \* sx)**;** // Calculate the constant term of the best fitting line  
 }  
 cout << "The best fitting line is y = " << coe << "x + " << ct << endl**;**}  
  
//display square error, and error of a certain number inside the vector x  
void squareErraErr(vector<float> x**,** vector<float> y**,** float coe**,** float ct**,** float num)  
{  
 float se = **0;** //Square error  
 float err = **0;** //Error of a specific number  
 for (int i = **0;** i < x.size()**;** i++) {  
 se += (((coe \* x[i] + ct) - y[i]) \* ((coe \* x[i] + ct) - y[i]))**;** if (num == x[i]) err = (y[i] - (coe \* x[i] + ct))**;** }  
 cout << "The errorSquared = " << se << endl**;** cout << "Error in " << num << " = " << err << endl**;**}  
//Predicts the respective y value for an specific x value  
float predict(float x**,** float& coe**,** float& ct)  
{  
 return coe \* x + ct**;**}