```cpp
#include <fstream>
#include <iostream>
#include <random>
#include <sstream>
#include <string>
#include <vector>
#include <algorithm>


static void readCsv(const std::string& filename,
                    std::vector<double>& x,
                    std::vector<double>& y)
{
    std::ifstream fin(filename);

    // skip header
    {
        std::string s;
        std::getline(fin, s);
    }

    double a, b;
    char delim;

    while(fin >> a >> delim >> b)
    {
        x.push_back(a);
        y.push_back(b);
    }
}

class PieceWiseLinearCDF
{
public:
    PieceWiseLinearCDF(std::vector<double> x, std::vector<double> y)
        : x_(std::move(x))
        , y_(std::move(y))
    {}

    PieceWiseLinearCDF(const std::string& filename)
    {
        readCsv(filename, x_, y_);

    }

    double generateSample(std::mt19937& gen, int c) const
    {
        std::uniform_real_distribution<double> distrib(0, 1);
        double u = distrib(gen);
        int i = 0;
        while (y_[i]<u){
            i += 1;
        }
        int j=0;
        while (0.5 > y_[j+1]){
            j++;
        }
        double the_median;

        if (c ==0){
            //the_median =(x_[j] +x_[j + 1] )/2;
            //the_median =x_[j];
            the_median = (0.5 -y_[j + 1])*(x_[j+1] - x_[j])/(y_[j+1] - y_[j]) + x_[j+1];
            std :: cout << "the theoretical median is "<< the_median << '\n';
            c+=1;
        }

        return (u - y_[i])*(x_[i + 1] - x_[i])/(y_[i + 1] - y_[i])+ x_[i];
    }

    double getMinX() const {return x_.front();}
    double getMaxX() const {return x_.back();}

private:
    std::vector<double> x_;
    std::vector<double> y_;
};


int main(int argc, char **argv)
{
    // Load the data
    PieceWiseLinearCDF cdf("cdf.csv");

    // histogram quantities
    const int nbins = 100;

    std::vector<double> histogramLoc(nbins); // center of each bin
    std::vector<int> histogramCounts(nbins); // number of samples per bin

    const double a = cdf.getMinX(); // lowest bound of the histogram
    const double b = cdf.getMaxX(); // highest bound of the histogram
    const double h = (b-a) / nbins; // bin size

    for (int i = 0; i < nbins; ++i){
        histogramLoc[i] = a + (i+0.5) * h;
    }

    // collect samples
    const int nsamples = 1'000'000;
    std::vector<double> samples(nsamples);
    std::mt19937 gen(3456789);
    int c = 0;
    for (auto& x : samples)
    {
        x = cdf.generateSample(gen, c);
        int i=0;
        while (histogramLoc[i] < x){
            i++;
        }
        histogramCounts[i] += 1;
        c = 1;
        // TODO fill in the histogram counts
    }

    // TODO compute and printout the empirical median and the analytical median
    double mid = 0.5;
    std::sort(std::begin(samples), std::end(samples));
    double emp_median = (samples[499999] + samples[500000])/2;

    std :: cout << "the empirical median is "<<emp_median;


    // write the histogram to a csv file

    std :: ofstream cout("histogram.csv");
    std :: cout << "x,count" << std::endl;

    for (int i = 0; i < nbins; ++i){
        std :: cout << histogramLoc[i] << ',' << histogramCounts[i]<< '\n';
    }
    return 0;
}
```