Help

```
extern "C" {
#include "pnl/pnl_random.h"
}

#include <ctime>
#include <cmath>
#include "rnd.h"

using namespace std;

StableRnd::StableRnd(float alpha1, float sigma1, float beta1, float mu1, int gen
{
  if (alpha != 1)
    {
      B = atan(beta * tan(M_PI * alpha / 2)) / alpha;
      S = pow((1 + beta * beta * SQR(tan(M_PI * alpha / 2))), 1. / 2 / alpha);
    }
}

float StableRnd::next()
{
  float V = M_PI * (pnl_rand_uni(generator) - 0.5);
  float W = -log(pnl_rand_uni(generator));
  float X;
  if (alpha != 1)
    {
      X = S * sin(alpha * (V + B)) / pow((double)cos(V), (double)1. / alpha) * p
      X = X * sigma + mu;
    }
  else
    {
      X = 2. / M_PI * ((M_PI / 2 + beta * V) * tan(V) - beta * log((M_PI / 2 * W
      X = X * sigma + mu + 2. / M_PI * beta * sigma * log(sigma);
    }
  return X;
}


float stablernd(float alpha, float sigma, float beta, float mu, int generator)
```

```
{
  float V = M_PI * (pnl_rand_uni(generator) - 0.5);
  float W = -log(pnl_rand_uni(generator));
  float B, S, X;
  if (alpha != 1)
    {
      B = atan(beta * tan(M_PI * alpha / 2)) / alpha;
      S = pow((1 + beta * beta * SQR(tan(M_PI * alpha / 2))), 1. / 2 / alpha);
      X = S * sin(alpha * (V + B)) / pow((double)cos(V), (double)1. / alpha) * p
      X = X * sigma + mu;
      //    std::cout << "X = "<< X << std::endl; // debug
    }
  else
    {
      X = 2. / M_PI * ((M_PI / 2 + beta * V) * tan(V) - beta * log((M_PI / 2 * W
      X = X * sigma + mu + 2. / M_PI * beta * sigma * log(sigma);
    }
  return X;
}
```