```
// Verzio: 3.1.1.cpp
// Forditas:
// g++ 3.1.1.cpp `libpng-config --ldflags` -O3 -o 3.1.1
// Futtatas:
// ./3.1.1 mandel.png 1920 1080 2040 -0.01947381057309366392260585598705802112818 -0.0194738105725413418456426484226540196687 0.798505756933826860155534177465597167611 1 0.7985057569343791961102851928444579243 66
// ./3.1.1 mandel.png 1920 1080 1020 0.4127655418209589255340574709407519549131 0.4127655418245818053080142817634623497725 0.2135387051768746491386963270997512154281 0.2135387051804975289126531379224616102874

#include <iostream>
#include "png++/png.hpp"
#include <complex>

int
main ( int argc, char *argv[] )
{

  int szelesseg = 1920;
  int magassag = 1080;
  int iteraciosHatar = 255;
  double a = -1.9;
  double b = 0.7;
  double c = -1.3;
  double d = 1.3;

  if ( argc == 9 )
    {
      szelesseg = atoi ( argv[2] );
      magassag =  atoi ( argv[3] );
      iteraciosHatar =  atoi ( argv[4] );
      a = atof ( argv[5] );
      b = atof ( argv[6] );
      c = atof ( argv[7] );
      d = atof ( argv[8] );
    }
  else
    {
      std::cout << "Hasznalat: ./3.1.1 fajlnev szelesseg magassag n a b c d" << std::endl;
      std::cout << "Most az alapbeallitasokkal futtatjuk " << szelesseg << " "
                << magassag << " "
                << iteraciosHatar << " "
                << a << " "
                << b << " "
                << c << " "
                << d << " " << std::endl;
      //return -1;
    }

  png::image < png::rgb_pixel > kep ( szelesseg, magassag );

  double dx = ( b - a ) / szelesseg;
  double dy = ( d - c ) / magassag;
  double reC, imC, reZ, imZ;
  int iteracio = 0;

  std::cout << "Szamitas\n";

  // j megy a sorokon
  for ( int j = 0; j < magassag; ++j )
    {
      // k megy az oszlopokon

      for ( int k = 0; k < szelesseg; ++k )
        {

          // c = (reC, imC) a halo racspontjainak
          // megfelelo komplex szam

          reC = a + k * dx;
          imC = d - j * dy;
          std::complex<double> c ( reC, imC );

          std::complex<double> z_n ( 0, 0 );
          iteracio = 0;

          while ( std::abs ( z_n ) < 4 && iteracio < iteraciosHatar )
            {
              z_n = z_n * z_n + c;

              ++iteracio;
            }

          iteracio %= 256;

          kep.set_pixel ( k, j,
                          png::rgb_pixel ( iteracio%255, 0, 0 ) );
        }

      int szazalek = ( double ) j / ( double ) magassag * 100.0;
      std::cout << "\r" << szazalek << "%" << std::flush;
    }

  kep.write ( argv[1] );
  std::cout << "\r" << argv[1] << " mentve." << std::endl;

}
```