```
//  file: derivative_test_simple.cpp
//
//  Program to study the error in differentiation rules using
//   the simplest algorithms.
//
//  Programmer:  Dick Furnstahl  furnstahl.1@osu.edu
//
//  Revision history:
//      01/08/07  original version, extracted from derivative_test.cpp
//
//  Notes:
//   * Based on the discussion of differentiation in Chap. 8
//      of "Computational Physics" by Landau and Paez.
//
//*****************************************************************

// include files
#include <iostream>             // basic input/output functions
#include <iomanip>              // manipulators like setprecision
#include <fstream>              // to read and write data as file streams
using namespace std;    // so that std::cout --> cout, etc.

// function prototypes
double test_function (double x);
double test_function_derivative (double x);

double forward_diff (double x, double h, double (*f) (double x));
double central_diff (double x, double h, double (*f) (double x));

//************************* main program **************************
int
main ()
{
  const double h_min = 1.e-10;  // minimum mesh size
  double x = 1.;                // find the derivative at x
  double diff_cd, diff_fd;      // central, forward difference

  ofstream deriv_out ("derivative_test_simple.dat"); // open the output file

  double exact = test_function_derivative(x);   // exact answer for test

  double h = 0.1;               // initialize mesh spacing
  while (h >= h_min)
  {
    diff_fd = forward_diff (x, h, &test_function);
    diff_cd = central_diff (x, h, &test_function);

    // print relative errors to output file
    deriv_out << scientific << setprecision (8)
      << log10(h) << " "
      << log10( fabs((diff_fd - exact)/exact) ) << " "
      << log10( fabs((diff_cd - exact)/exact) ) << " "
      << endl;

    h /= 2.;            // reduce mesh (x spacing) by 2 before repeating
  }

  deriv_out.close ();         // close the output stream
  return (0);           // successful completion
}

//************************* test_function *************************
double
test_function (double x)
{
  double alpha = 1.;    // a parameter for the function
  return (exp (-alpha * x));
}
```



```
//********************* test_function_derivative ******************
double
test_function_derivative (double x)
{
  double alpha = 1.;    // Note that we had to repeat this, which is bad!
  return (-alpha * exp (-alpha * x));
}

//************************ forward_diff ***************************
double
forward_diff (double x, double h, double (*f) (double x))
{
  return ( f(x + h) - f(x) ) / h;
}

//************************ central_diff ***************************
double
central_diff (double x, double h, double (*f) (double x))
{
  return ( f(x + h/2.) - f(x - h/2.) ) / h;
}

//*****************************************************************
```