

```
/*
 * precision.c
 *
 * This file contains the functions
 *
 *     int decimal_places_of_accuracy(double x, double y);
 *     int complex_decimal_places_of_accuracy(Complex x, Complex y);
 *
 * which are used within the kernel to determine how many decimal
 * places of accuracy the doubles (resp. Complexes) x and y have in common.
 * Typically x and y will be two estimates of the same computed quantity
 * (e.g. the volume of a manifold as computed at the last and the
 * next-to-the-last iterations of Newton's method in
 * hyperbolic_structures.c), and decimal_places_of_accuracy()
 * will be used to tell the user interface how many decimal places
 * should be printed. We compute the number of decimal places of
 * accuracy instead of the number of significant figures, because
 * this is what printf() requires in its formatting string.
 */

#include "kernel.h"

int decimal_places_of_accuracy(
    double x,
    double y)
{
    int digits;

    if (x == y)
    {
        if (x == 0.0)
            digits = DBL_DIG;
        else
            digits = DBL_DIG - (int) ceil(log10(fabs(x)));
    }
    else
        digits = - (int) ceil(log10(fabs(x - y)));

    /*
     * Typically the difference between the computed values
     * of a quantity at the penultimate and ultimate iterations
     * of Newton's method is a little less than the true error.
     * So we fudge a bit.
     */
    digits -= 4;

    if (digits < 0)
        digits = 0;

    return digits;
}

int complex_decimal_places_of_accuracy(
    Complex x,
    Complex y)
{
    int real_precision,
        imag_precision;

    real_precision = decimal_places_of_accuracy(x.real, y.real);
    imag_precision = decimal_places_of_accuracy(x.imag, y.imag);

    return MIN(real_precision, imag_precision);
}
```