Help

```
#if defined(PremiaCurrentVersion) && PremiaCurrentVersion < (2007+2) //The "#els
#else

//Routine for Golden Section Search
#include <math.h>

#include "golden.h"

#define R 0.61803399 //The golden ratios.
#define C (1.0-R)
#define SHFT2(a,b,c) (a)=(b);(b)=(c);
#define SHFT3(a,b,c,d) (a)=(b);(b)=(c);(c)=(d);

/*Given a function F, and given a bracketing triplet of abscissas ax, bx, cx (su
between ax and cx, and F(bx) is less than both F(ax) and F(cx)), this routine pe
golden section search for the minimum, isolating it to a fractional precision of
abscissa of the minimum is returned as xmin, and the minimum function value is r
golden, the returned function value.*/

double golden(PnlFunc *F, double ax, double bx, double cx, double tol, double *x
{
  double f1, f2, x0, x1, x2, x3;

  x0 = ax; //At any given time we will keep track of four points, x0,x1,x2,x3.
  x3 = cx;
  if (fabs(cx - bx) > fabs(bx - ax)) //Make x0 to x1 the smaller segment, and fi
    {
      x1 = bx;
      x2 = bx + C * (cx - bx);
    }
  else
    {
      x2 = bx;
      x1 = bx - C * (bx - ax);
    }
  f1 = PNL_EVAL_FUNC(F, x1); //The initial function evaluations. Note that we ne
  f2 = PNL_EVAL_FUNC(F, x2);
  while (fabs(x3 - x0) > tol * (fabs(x1) + fabs(x2)))
    {
```

```
      if (f2 < f1) //One possible outcome, its housekeeping, and a new function
        {
          SHFT3(x0, x1, x2, R * x1 + C * x3)
          SHFT2(f1, f2, PNL_EVAL_FUNC(F, x2))
        }
      else //The other outcome, and its new function evaluation.
        {
          SHFT3(x3, x2, x1, R * x2 + C * x0)
          SHFT2(f2, f1, PNL_EVAL_FUNC(F, x1))
        }
    } // Back to see if we are done.
  if (f1 < f2) //We are done. Output the best of the two current values.
    {
      *xmin = x1;
      return f1;
    }
  else
    {
      *xmin = x2;
      return f2;
    }

  return 0;
}


#endif //PremiaCurrentVersion
```