Help

```
#include "
href../../../mod/hullwhite1dgeneralized/hullwhite1dgeneralized_stdi/hullwhite1dg

#include "
href../../../common/math/read_market_zc/InitialYieldCurve_h_src.pdfmath/read_mar
#include "
href../../../mod/hullwhite1dgeneralized/hullwhite1dgeneralized_stdi/hullwhite1dg


#if defined(PremiaCurrentVersion) && PremiaCurrentVersion < (2010+2) //The "#els
int CALC(CF_ZCPutBondEuroHW1DG)(void *Opt, void *Mod, PricingMethod *Met)
{
  return AVAILABLE_IN_FULL_PREMIA;
}
static int CHK_OPT(CF_ZCPutBondEuroHW1DG)(void *Opt, void *Mod)
{
  return NONACTIVE;
}
#else

/*Put Option*/
static int cf_zbc1d(double flat_flag, double a, int CapletCurve, double flat_yie
{
  double strike;

  ModelHW1dG HW1dG_Parameters;
  ZCMarketData ZCMarket;
  MktATMCapletVolData MktATMCapletVol;

  /* Flag to decide to read or not ZC bond datas in "initialyields.dat" */
  /* If P(0,T) not read then P(0,T)=exp(-r0*T) */
  if (flat_flag == 0)
    {
      ZCMarket.FlatOrMarket = 0;
      ZCMarket.Rate = flat_yield;
    }

  else
    {
```

```
      ZCMarket.FlatOrMarket = 1;
      ZCMarket.filename = curve;
      ReadMarketData(&ZCMarket);

      if (S > GET(ZCMarket.tm, ZCMarket.Nvalue - 1))
        {
          printf("\ nError : time bigger than the last time value entered in ini
          exit(EXIT_FAILURE);
        }
    }

  ReadCapletMarketData(&MktATMCapletVol, CapletCurve);

  hw1dg_calibrate_volatility(&HW1dG_Parameters, &ZCMarket, &MktATMCapletVol, a);

  strike = p->Par[0].Val.V_DOUBLE;
  /*Price*/
  *price = hw1dg_zc_put_price(&ZCMarket, &HW1dG_Parameters, strike, T, S);

  DeleteZCMarketData(&ZCMarket);
  DeleteMktATMCapletVolData(&MktATMCapletVol);
  DeletModelHW1dG(&HW1dG_Parameters);

  return OK;
}

int CALC(CF_ZCPutBondEuroHW1DG)(void *Opt, void *Mod, PricingMethod *Met)
{
  TYPEOPT *ptOpt = (TYPEOPT *)Opt;
  TYPEMOD *ptMod = (TYPEMOD *)Mod;

  return cf_zbc1d(ptMod->flat_flag.Val.V_INT,
                  ptMod->a.Val.V_DOUBLE,
                  ptMod->CapletCurve.Val.V_ENUM.value,
                  MOD(GetYield)(ptMod),
                  MOD(GetCurve)(ptMod),
                  ptOpt->OMaturity.Val.V_DATE - ptMod->T.Val.V_DATE,
                  ptOpt->BMaturity.Val.V_DATE - ptMod->T.Val.V_DATE,
                  ptOpt->PayOff.Val.V_NUMFUNC_1,
                  &(Met->Res[0].Val.V_DOUBLE));
}
```

```
static int CHK_OPT(CF_ZCPutBondEuroHW1DG)(void *Opt, void *Mod)
{
  return strcmp(((Option *)Opt)->Name, "ZeroCouponPutBondEuro");
}
#endif //PremiaCurrentVersion



static int MET(Init)(PricingMethod *Met, Option *Opt)
{
  if (Met->init == 0)
    {
      Met->init = 1;
      Met->HelpFilenameHint = "cf_hullwhite1dgeneralized_zbputeuro";

    }

  return OK;
}

PricingMethod MET(CF_ZCPutBondEuroHW1DG) =
{
  "CF_HullWhite1dG_ZBPutEuro",
  {{" ", PREMIA_NULLTYPE, {0}, FORBID}},
  CALC(CF_ZCPutBondEuroHW1DG),
  {{"Price", DOUBLE, {100}, FORBID}, {" ", PREMIA_NULLTYPE, {0}, FORBID}},
  CHK_OPT(CF_ZCPutBondEuroHW1DG),
  CHK_ok,
  MET(Init)
} ;
```