Help

```
#include <stdlib.h>
#include "
href../../../mod/hullwhite1d/hullwhite1d_stdi/hullwhite1d_stdi_h_src.pdfhullwhit
#include "
href../../../mod/hullwhite1d/hullwhite1d_stdi/hullwhite1d_includes_h_src.pdfhull


#if defined(PremiaCurrentVersion) && PremiaCurrentVersion < (2007+2) //The "#els
int CALC(CF_ZCCallBondEuroHW1D)(void *Opt, void *Mod, PricingMethod *Met)
{
  return AVAILABLE_IN_FULL_PREMIA;
}
static int CHK_OPT(CF_ZCCallBondEuroHW1D)(void *Opt, void *Mod)
{
  return NONACTIVE;
}
#else

double cf_hw1d_zbcall(ZCMarketData *ZCMarket, double a, double sigma, double S,
{
  double P0T, P0S, price;
  double d1, d2, sigma_p;

  //Price of an option on a ZC

  /*Computation pure discount bond*/
  P0T = BondPrice(T, ZCMarket);
  P0S = BondPrice(S, ZCMarket);

  sigma_p = sigma * sqrt((1. - exp(-2.*a * T)) / (2.*a)) * (1. / a) * (1. - exp(
  d1 = 1. / (sigma_p) * log(P0S / (P0T * K)) + 0.5 * sigma_p;
  d2 = d1 - sigma_p;

  /*Price*/
  price = P0S * cdf_nor(d1) - K * P0T * cdf_nor(d2);

  return price;
}
```

```
/*Call Option*/
static int cf_zbc1d(double flat_flag, double a, double sigma, double r_t, char *
{
  double K;

  ZCMarketData ZCMarket;

  /* 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 = r_t;
    }

  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);
        }
    }

  K = p->Par[0].Val.V_DOUBLE;
  /*Price*/
  *price = cf_hw1d_zbcall(&ZCMarket, a, sigma, S, T, K);

  DeleteZCMarketData(&ZCMarket);

  return OK;
}

int CALC(CF_ZCCallBondEuroHW1D)(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->Sigma.Val.V_PDOUBLE,
                  MOD(GetYield)(ptMod),
                  MOD(GetCurve)(ptMod),
                  ptOpt->BMaturity.Val.V_DATE - ptMod->T.Val.V_DATE,
                  ptOpt->OMaturity.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_ZCCallBondEuroHW1D)(void *Opt, void *Mod)
{
  return strcmp(((Option *)Opt)->Name, "ZeroCouponCallBondEuro");
}
#endif //PremiaCurrentVersion



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

  return OK;
}

PricingMethod MET(CF_ZCCallBondEuroHW1D) =
{
  "CF_HullWhite1d_ZBCallEuro",
  {{" ", PREMIA_NULLTYPE, {0}, FORBID}},
  CALC(CF_ZCCallBondEuroHW1D),
  {{"Price", DOUBLE, {100}, FORBID}, {" ", PREMIA_NULLTYPE, {0}, FORBID}},
  CHK_OPT(CF_ZCCallBondEuroHW1D),
  CHK_ok,
  MET(Init)
} ;
```