

[Help](#)

```
#include "hullwhite1d_std.h"

int MOD_OPT(ChkMix)(Option *Opt,Model *Mod)
{
    TYPEOPT* ptOpt=( TYPEOPT*)(Opt->TypeOpt);
    TYPEMOD* ptMod=( TYPEMOD*)(Mod->TypeModel);
    int status=OK;

    if ((strcmp(Opt->Name,"ZeroCouponCallBondEuro")==0) || (
        strcmp(Opt->Name,"ZeroCouponPutBondEuro")==0) || (strcmp(Opt->Name,"ZeroCouponCallBondAmer")==0) || (strcmp(Opt->Name,"ZeroCouponPutBondAmer")==0))
    {
        if ((ptOpt->OMaturity.Val.V_DATE)<=(ptMod->T.Val.V_DATE))
        {
            Fprintf(TOSCREENANDFILE,"Current date greater than maturity!\n");
            status+=1;
        }
        if ((ptOpt->BMaturity.Val.V_DATE)<=(ptOpt->OMaturity.Val.V_DATE))
        {
            Fprintf(TOSCREENANDFILE,"Option maturity greater than Bond maturity!\n");
            status+=1;
        }
    }
    if ((strcmp(Opt->Name,"ZeroCouponBond")==0))
    {
        if ((ptOpt->BMaturity.Val.V_DATE)<=(ptMod->T.Val.V_DATE))
        {
            Fprintf(TOSCREENANDFILE,"Current date greater than maturity!\n");
            status+=1;
        }
    }
    if ((strcmp(Opt->Name,"PayerSwaption")==0) || (strcmp(Opt->Name,"ReceiverSwaption")==0) || (strcmp(Opt->Name,"
```

```

    PayerBermudanSwaption")==0)|| (strcmp(Opt->Name,"
    ReceiverBermudanSwaption")==0))
    if((ptOpt->BMaturity.Val.V_DATE)<=(ptOpt->OMaturity.
    Val.V_DATE))
    {
        Fprintf(TOSCREENANDFILE,"Option maturity greater than
        Bond maturity!\n");
        status+=1;
    }

    if ((strcmp(Opt->Name,"Floor")==0)|| (strcmp(Opt->Name,"    Cap")==0))
    {
        if ((ptOpt->FirstResetDate.Val.V_DATE)<=(ptMod->T.Val
        .V_DATE))
        {
            Fprintf(TOSCREENANDFILE,"Current date greater than fir
            st coupon date!\n");
            status+=1;
        }
        if ((ptOpt->FirstResetDate.Val.V_DATE)>=(ptOpt->BMatu
        rity.Val.V_DATE))
        {
            Fprintf(TOSCREENANDFILE,"First reset date greater than
            contract maturity!\n");
            status+=1;
        }
    }

    return status;
}

```

```

extern PricingMethod MET(CF_ZCBondHW1D);
extern PricingMethod MET(CF_ZCCallBondEuroHW1D);
extern PricingMethod MET(CF_ZCPutBondEuroHW1D);
extern PricingMethod MET(CF_CapHW1D);
extern PricingMethod MET(CF_FloorHW1D);
extern PricingMethod MET(CF_PayerSwaptionHW1D);
extern PricingMethod MET(CF_ReceiverSwaptionHW1D);
extern PricingMethod MET(TR_ZCBondHW1D);
extern PricingMethod MET(TR_ZBOHW1D);

```

```
extern PricingMethod MET(TR_SwaptionHW1D);
extern PricingMethod MET(TR_CapFloorHW1D);
extern PricingMethod MET(TR_BermudianSwaptionHW1D);
```

```
PricingMethod* MOD_OPT(methods)[]={
    &MET(CF_ZCBondHW1D),
    &MET(CF_ZCCallBondEuroHW1D),
    &MET(CF_ZCPutBondEuroHW1D),
    &MET(CF_CapHW1D),
    &MET(CF_FloorHW1D),
    &MET(CF_PayerSwaptionHW1D),
    &MET(CF_ReceiverSwaptionHW1D),
    &MET(TR_ZCBondHW1D),
    &MET(TR_ZBOHW1D),
    &MET(TR_SwaptionHW1D),
    &MET(TR_CapFloorHW1D),
    &MET(TR_BermudianSwaptionHW1D),
    NULL
};
DynamicTest* MOD_OPT(tests)[]={
    NULL
};
```

```
Pricing MOD_OPT(pricing)={
    ID_MOD_OPT,
    MOD_OPT(methods),
    MOD_OPT(tests),
    MOD_OPT(ChkMix)
};
```

## References