```
   Help
#include "bs1d_doublim.h"

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

    /*Custom*/
    if (ptOpt->Maturity.Val.V_DATE<=ptMod->T.Val.
    V_DATE)
        {
        Fprintf(TOSCREENANDFILE,"Current date gre
    ater than maturity!{n");
        status+=1;
        };
    if ( ((ptOpt->LowerLimit.Val.V_NUMFUNC_1)->
    Compute)((ptOpt->LowerLimit.Val.V_NUMFUNC_1)->Par,pt
    Mod->T.Val.V_DATE)>ptMod->S0.Val.V_PDOUBLE)
        {
        Fprintf(TOSCREENANDFILE,"Limit Down great
    er than spot!{n");
        status+=1;
        };

    if ( ((ptOpt->UpperLimit.Val.V_NUMFUNC_1)->
    Compute)((ptOpt->UpperLimit.Val.V_NUMFUNC_1)->Par,pt
    Mod->T.Val.V_DATE)<ptMod->S0.Val.V_PDOUBLE)
        {
        Fprintf(TOSCREENANDFILE,"Limit Up lower
    than spot!{n");
        status+=1;
        };
    /*EndCustom*/

    return status;
    }

extern PricingMethod MET(MC_OutBaldi);
extern PricingMethod MET(MC_InBaldi);
```

```
extern PricingMethod MET(MC_ParisianOut);
extern PricingMethod MET(MC_ParisianIn);
extern PricingMethod MET(AP_Out_Laplace);
extern PricingMethod MET(
    CF_CallOut_KunitomoIkeda);
extern PricingMethod MET(CF_PutIn_KunitomoIkeda);
extern PricingMethod MET(
    CF_PutOut_KunitomoIkeda);
extern PricingMethod MET(FD_Psor_Out);
extern PricingMethod MET(FD_Psor_In);
extern PricingMethod MET(FD_Cryer_Out);
extern PricingMethod MET(FD_Cryer_In);
extern PricingMethod MET(FD_Gauss_In);
extern PricingMethod MET(FD_Gauss_Out);
extern PricingMethod MET(FD_Fem_Out);
extern PricingMethod MET(
    CF_CallIn_KunitomoIkeda);
extern PricingMethod MET(TR_Ritchken_In);
extern PricingMethod MET(TR_Ritchken_Out);

PricingMethod* MOD_OPT(methods)[]={
        &MET(CF_CallOut_KunitomoIkeda),
        &MET(CF_PutOut_KunitomoIkeda),
        &MET(CF_CallIn_KunitomoIkeda),
        &MET(CF_PutIn_KunitomoIkeda),
    &MET(AP_Out_Laplace),
        &MET(FD_Psor_Out),
        &MET(FD_Psor_In),
    &MET(FD_Cryer_Out),
    &MET(FD_Cryer_In),
    &MET(FD_Gauss_In),
        &MET(FD_Gauss_Out),
    &MET(FD_Fem_Out),
        &MET(TR_Ritchken_In),
        &MET(TR_Ritchken_Out),
        &MET(MC_OutBaldi),
        &MET(MC_InBaldi),
    &MET(MC_ParisianOut),
    &MET(MC_ParisianIn),
        NULL
    };
```

```
extern DynamicTest MOD_OPT(test);
DynamicTest* MOD_OPT(tests)[]={
    &MOD_OPT(test),
    NULL
  };

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

References