```
   Help
#include  "hes1d_std.h"

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

  if ((ptOpt->Maturity.Val.V_DATE)<=(ptMod->T.Val.V_DATE))
    {
      Fprintf(TOSCREENANDFILE,"Current date greater than
    maturity!{n");
      status+=1;
    };

  return status;
}

extern PricingMethod MET(CF_CallHeston);
extern PricingMethod MET(CF_PutHeston);
extern PricingMethod MET(CF_CarrHeston);
extern PricingMethod MET(CF_AttariHeston);
extern PricingMethod MET(MC_Alfonsi_Heston);
extern PricingMethod MET(MC_Andersen_Heston);
extern PricingMethod MET(MC_Smith_Heston);
extern PricingMethod MET(MC_Zhu_Heston);
extern PricingMethod MET(MC_Pelsser_Heston);
extern PricingMethod MET(MC_KahlJackel_Heston);
extern PricingMethod MET(MC_Lord_Heston);
extern PricingMethod MET(MC_BroadieKaya_Heston);
extern PricingMethod MET(MC_GlassermanKim_Heston);
extern PricingMethod MET(MC_Joshi);
extern PricingMethod MET(MC_GlassermanKimMod_Heston);
//extern PricingMethod MET(MC_Giles_Heston);
extern PricingMethod MET(MC_RobbinsMonro_Heston);
extern PricingMethod MET(AP_Cosine_Euro);

extern PricingMethod MET(AP_Alos_Heston);
extern PricingMethod MET(AP_BGM_Heston);
extern PricingMethod MET(AP_AntonelliScarlatti_Heston);
```

```
extern PricingMethod MET(AP_SPM_Heston);
extern PricingMethod MET(TR_VELLEKOOPNIEUWENHUIS_Heston);
extern PricingMethod MET(AP_fastwhamer_hes);
extern PricingMethod MET(FD_Heston1);

extern PricingMethod MET(MC_AM_Alfonsi_LongstaffSchwartz);
extern PricingMethod MET(MC_AM_Alfonsi_AndersenBroadie);
extern PricingMethod MET(MC_AM_Alfonsi_Iterative);
extern PricingMethod MET(MC_AM_Alfonsi_MLSM);
extern PricingMethod MET(MC_MALLIAVIN_HESTON);
extern PricingMethod MET(AP_CosineBermudan);
extern PricingMethod MET(AP_SmallTime_ImpliedVolatility);
extern PricingMethod MET(AP_Asymptotics_ImpliedVolatility);
//extern PricingMethod MET(AP_fastwhamerdig_hes);

//extern PricingMethod MET(FD_Fem_Achdou);

PricingMethod* MOD_OPT(methods)[]={
  &MET(CF_CallHeston),
  &MET(CF_PutHeston),
  &MET(CF_CarrHeston),
  &MET(CF_AttariHeston),
  &MET(MC_Alfonsi_Heston),
  &MET(MC_Andersen_Heston),
  &MET(MC_Lord_Heston),
  &MET(MC_KahlJackel_Heston),
  &MET(MC_RobbinsMonro_Heston),
  &MET(MC_Pelsser_Heston),
  &MET(MC_BroadieKaya_Heston),
  &MET(MC_GlassermanKim_Heston),
  &MET(MC_GlassermanKimMod_Heston),
  &MET(MC_Joshi),
  &MET(MC_Smith_Heston),
  &MET(MC_Zhu_Heston),
  //&MET(MC_Giles_Heston),
  &MET(AP_Cosine_Euro),

  &MET(AP_Alos_Heston),
  &MET(AP_BGM_Heston),
  &MET(AP_AntonelliScarlatti_Heston),
  &MET(AP_SPM_Heston),
```

```
  &MET(TR_VELLEKOOPNIEUWENHUIS_Heston),
  &MET(AP_fastwhamer_hes),
  &MET(FD_Heston1),

  &MET(MC_AM_Alfonsi_LongstaffSchwartz),
  &MET(MC_AM_Alfonsi_AndersenBroadie),
  &MET(MC_AM_Alfonsi_Iterative),
  &MET(MC_AM_Alfonsi_MLSM),
  &MET(MC_MALLIAVIN_HESTON),
    &MET(AP_CosineBermudan),
  &MET(AP_SmallTime_ImpliedVolatility),
  &MET(AP_Asymptotics_ImpliedVolatility),
  //&MET(AP_fastwhamerdig_hes),
  //&MET(FD_Fem_Achdou),
  NULL
};

DynamicTest* MOD_OPT(tests)[]={
  NULL
};

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

References