```
   Help
#include  "mer1d_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_Call_Merton);
extern PricingMethod MET(CF_Put_Merton);
extern PricingMethod MET(AP_Carr);
extern PricingMethod MET(MC_Merton);
extern PricingMethod MET(MC_Privault);
extern PricingMethod MET(FD_ImpExp2);
extern PricingMethod MET(TR_MSS_MER);
extern PricingMethod MET(FD_AndersenAndreasen);
extern PricingMethod MET(FD_ImpExp);
extern PricingMethod MET(FD_Explicit);
extern PricingMethod MET(AP_STATICHEDGING_CARRWU);

PricingMethod* MOD_OPT(methods)[]={
  &MET(CF_Call_Merton),
  &MET(CF_Put_Merton),
  &MET(AP_Carr),
  &MET(MC_Merton),
  &MET(MC_Privault),
  &MET(FD_ImpExp2),
  &MET(TR_MSS_MER),
  &MET(FD_AndersenAndreasen),
```

```
  &MET(FD_ImpExp),
  &MET(FD_Explicit),
  &MET(AP_STATICHEDGING_CARRWU),
  NULL
};

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

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

References