```
   Help
#include "mer1d_pad.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;
    };
  if ((ptOpt->MinOrElse).Val.V_BOOL==MINIMUM)
    {
      if ((ptOpt->PathDep.Val.V_NUMFUNC_2)->Par[4].Val.V_
    PDOUBLE>ptMod->S0.Val.V_PDOUBLE)
  {
    Fprintf(TOSCREENANDFILE,"Minimum greater than spot!{n"
    );
    status+=1;
  };
    }
  if ((ptOpt->MinOrElse).Val.V_BOOL==MAXIMUM)
    {
      if ((ptOpt->PathDep.Val.V_NUMFUNC_2)->Par[4].Val.V_
    PDOUBLE<ptMod->S0.Val.V_PDOUBLE)
  {
    Fprintf(TOSCREENANDFILE,"Maximum lower than spot!{n");
    status+=1;
  };
    }
  return status;
}
extern PricingMethod MET(AP_Asian_FMMMER);
extern PricingMethod MET(AP_FixedAsian_FusaiMeucciMER);
extern PricingMethod MET(MC_Merton_FixedLookback);
extern PricingMethod MET(MC_Merton_FloatingLookback);
```

```
PricingMethod *MOD_OPT(methods)[]={
  &MET(AP_Asian_FMMMER),
  &MET(AP_FixedAsian_FusaiMeucciMER),
  &MET(MC_Merton_FixedLookback),
  &MET(MC_Merton_FloatingLookback),
  NULL
};
DynamicTest* MOD_OPT(tests)[]={
  NULL
};

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

# References