```
   Help
#include "optype.h"
#include "var.h"
#include "tools.h"
#include "random.h"

int CHK_ok(int user, Planning *pt_plan,void* dum)
  {
  return OK;
  }

int CHK_call(int user, Planning *pt_plan,void*
    dum)
  {
  NumFunc_1* payoff=(NumFunc_1*)dum;
  int status=OK;

  status+=ChkParVar(pt_plan,payoff->Par);

  return status;
  }

int CHK_callspread(int user, Planning *pt_plan,vo
    id* dum)
  {
  NumFunc_1* payoff=(NumFunc_1*)dum;
  int status=OK;

  status+=ChkParVar(pt_plan,payoff->Par);

  if (payoff->Par[1].Val.V_PDOUBLE<payoff->Par[0
    ].Val.V_PDOUBLE)
    {
    Fprintf(TOSCREENANDFILE,"%s: lower than %s{
    n",payoff->Par[1].Vname,payoff->Par[0].Vname);
    status+=1;
    }

  return status;
  }
```

```
int CHK_digit(int user, Planning *pt_plan,void*
    dum)
  {
  NumFunc_1* payoff=(NumFunc_1*)dum;
  int status=OK;

  status+=ChkParVar(pt_plan,payoff->Par);

  return status;
  }

int CHK_tree(int user, Planning *pt_plan,void*
    dum)
  {
  PricingMethod* Met=(PricingMethod*)dum;
  int status=OK;

  status+=ChkParVar(pt_plan,Met->Par);

  return status;
  }

int CHK_mc(int user, Planning *pt_plan,void* dum)
  {
  PricingMethod* Met=(PricingMethod*)dum;
  int status=OK;

  status+=ChkParVar(pt_plan,Met->Par);

  return status;
  }

int CHK_mcBaldi(int user, Planning *pt_plan,void*
     dum)
  {
  PricingMethod* Met=(PricingMethod*)dum;
  int status=OK;

  status+=ChkParVar(pt_plan,Met->Par);

  return status;
```

```
  }

int CHK_fdiff(int user, Planning *pt_plan,void*
    dum)
  {
  PricingMethod* Met=(PricingMethod*)dum;
  int status=OK;

  status+=ChkParVar(pt_plan,Met->Par);

  return status;
  }

int CHK_split(int user, Planning *pt_plan,void*
    dum)
  {
  PricingMethod* Met=(PricingMethod*)dum;
  int status=OK;

  status+=ChkParVar(pt_plan,Met->Par);

  return status;
  }

int CHK_psor(int user, Planning *pt_plan,void*
    dum)
  {
  PricingMethod* Met=(PricingMethod*)dum;
  int status=OK;

  status+=ChkParVar(pt_plan,Met->Par);

  return status;
  }



int CHK_mc_generator(int user, Planning *pt_plan,
    void* dum)
{
  PricingMethod* Met=(PricingMethod*)dum;
```

```
  int status=OK;

  int type_generator;

  type_generator= Met->Par[1].Val.V_INT;

  if(Rand_Or_Quasi(type_generator)==QMC)
    {
      Fprintf(TOSCREEN, "Type generator should
    be MC and not QMC");
      status+= 1;
    }

  return status;
}
```

References