```
   Help
#include "bsdisdiv1d.h"
#include "chk.h"
#include "error_msg.h"
#include "model.h"
#include "pnl/pnl_matrix.h"

extern char* path_sep;

static int MOD(Init)(Model *model)
{
  TYPEMOD* pt=(TYPEMOD*)(model->TypeModel);

  if (model->init == 0 )
    {

      model->init = 1;
      model->nvar=0;
      pt->T.Vname = "Current Date";
      pt->T.Vtype=DATE;
      pt->T.Val.V_DATE=0.;
      pt->T.Viter=ALLOW;
      model->nvar++;

      pt->Size.Vname = "Number of Discrete Dividends";
      pt->Size.Vtype=PINT;
      pt->Size.Val.V_PINT=1;
      pt->Size.Viter=FORBID;
      model->nvar++;

      pt->S0.Vname = "Spot";
      pt->S0.Vtype=PDOUBLE;
      pt->S0.Val.V_PDOUBLE=100.;
      pt->S0.Viter=ALLOW;
      model->nvar++;

      pt->Mu.Vname = "Trend";
      pt->Mu.Vtype=DOUBLE;
      pt->Mu.Val.V_DOUBLE=0.;
      pt->Mu.Viter=ALLOW;
      model->nvar++;
      pt->Sigma.Vname = "Volatility";
      pt->Sigma.Vtype=PDOUBLE;
      pt->Sigma.Val.V_PDOUBLE=0.2;
      pt->Sigma.Viter=ALLOW;
      model->nvar++;

      pt->R.Vname = "Annual Interest Rate";
      pt->R.Vtype=DOUBLE;
      pt->R.Val.V_DOUBLE=10.;
      pt->R.Viter=ALLOW;
      model->nvar++;

      pt->Amounts.Vname = "Dividend Amounts";
      pt->Amounts.Vtype=PNLVECT;
      pt->Amounts.Val.V_PNLVECT=NULL;
      pt->Amounts.Viter=FORBID;
      model->nvar++;

      pt->Dates.Vname = "Dividend Dates";
      pt->Dates.Vtype=PNLVECT;
      pt->Dates.Val.V_PNLVECT=NULL;
      pt->Dates.Viter=FORBID;
      model->nvar++;
    }

  if (pt->Amounts.Val.V_PNLVECT==NULL){
    if ((pt->Amounts.Val.V_PNLVECT=
         pnl_vect_create_from_double (pt->Size.Val.V_PINT,
    3.0))==NULL)
      goto err;
  }

  if(pt->Dates.Val.V_PNLVECT==NULL){
    if ((pt->Dates.Val.V_PNLVECT=
         pnl_vect_create_from_double (pt->Size.Val.V_PINT,0
    .5))==NULL)
      goto err;
  }

  /*  pt->Size.Val.V_INT = pt->S0.Val.V_PNLVECT->size;*/
  return OK;

 err:
  Fprintf(TOSCREEN,"%s{n",error_msg[MEMORY_ALLOCATION_FAILU
    RE]);
  exit(WRONG);
}


static int MOD(Get)(int user,Planning *pt_plan,Model *
    model)
{
  int nvar;
  TYPEMOD *pt=(model->TypeModel);
  VAR *var = ((VAR*) pt);
  int i;

  model->Init(model);
  nvar = model->nvar;
  if (user==TOSCREEN)
    if  ((model->Show)(user,pt_plan,model))
      do
        {
          Fprintf(TOSCREEN,"_____________________________
    Model:%s{n",model->Name);

          ScanVar(pt_plan, user, &(pt->Size));
          pnl_vect_resize_from_double(pt->Amounts.Val.V_PN
    LVECT, pt->Size.Val.V_PINT, 3.0);
          pnl_vect_resize_from_double(pt->Dates.Val.V_PNLV
    ECT, pt->Size.Val.V_PINT, 0.5);
          for (i=1; i<nvar; i++)
            {
              ScanVar(pt_plan,user,&(var[i]));
            }
        }
      while ((model->Show)(user,pt_plan,model));

  return ((model->Show)(TOSCREENANDFILE,pt_plan,model));
}
```

```
static int MOD(FGet)(char **InputFile,int user,Planning *pt
    _plan,Model *model)
{
  int nvar;
  TYPEMOD *pt=(TYPEMOD*)(model->TypeModel);
  VAR *var = ((VAR*) pt);
  int i;

  model->Init(model);
  nvar = model->nvar;
  if (user==TOSCREEN)
    if  ((model->Show)(user,pt_plan,model))
      do
        {
          Fprintf(TOSCREEN,"_____________________________
    Model:%s{n",model->Name);

          FScanVar(InputFile, pt_plan, user, &(pt->Size));

          pnl_vect_resize_from_double(pt->Amounts.Val.V_PN
    LVECT, pt->Size.Val.V_PINT,3.0);
          pnl_vect_resize_from_double(pt->Dates.Val.V_PNLV
    ECT, pt->Size.Val.V_PINT,0.5);

          for (i=1; i<nvar; i++)
            {
              FScanVar(InputFile,pt_plan,user,&(var[i]));
            }
        }
      while ((model->Show)(user,pt_plan,model));

  return ((model->Show)(TOSCREENANDFILE,pt_plan,model));

}


/**
 * Check function for BSDISDIV1D
 * @param user:
 * @param pt_plan:
 * @param model: the model to be checked
 *
 * general model check function
 */
int MOD(Check)(int user,Planning *pt_plan,Model *model)

{
  VAR *var;
  void* pt=(model->TypeModel);
  int status=OK;
  int i, nvar=0;
  char helpfile[MAX_PATH_LEN]="";

  if ((2*strlen(model->ID)+strlen("{{mod{{") +strlen("{{")
       +strlen("_doc.pdf"))>=MAX_PATH_LEN)
    {
      Fprintf(TOSCREEN,"%s{n",error_msg[PATH_TOO_LONG]);
      exit(WRONG);
    }

  strcpy(helpfile,path_sep);
  strcat(helpfile,"mod");
  strcat(helpfile,path_sep);

  strcat(helpfile,model->ID);
  strcat(helpfile,path_sep);

  strcat(helpfile,model->ID);
  strcat(helpfile,"_doc.pdf");

  nvar = model->nvar;
  var = ((VAR*) pt);
  for (i=0; i<nvar; i++)
    {
      status+=ChkVar(pt_plan, &(var[i]));
      if (var[i].Vtype==PNLVECT && var[i].Val.V_PNLVECT->si
    ze != ((BSDISDIV1D*)pt)->Size.Val.V_PINT)
        status += 1;
    }
  return Valid(user,status,helpfile);
}
```

```
TYPEMOD BlackScholesDisDiv1dim;
MAKEMOD_FULL(BlackScholesDisDiv1dim);
```

References