Help

```
#ifndef  _OPTYPE_H
#define _OPTYPE_H


#include <stdio.h>
#include <stdlib.h>
#include <
href../common/math/cdo/cdo_math_h_src.pdfmath.h>
#include <string.h>
#include <stdarg.h>
#include <time.h>
#include <ctype.h>

#include "pnl/pnl_matrix.h"
#include "pnl/pnl_vector.h"
#include "
href../common/error_msg_h_src.pdferror_msg.h"

#ifdef _MSC_VER
#define MAXPATHLEN 256
#include "../configwin_specific.h"
#include <
href../common/math/equity_pricer/levy_process_h_src.pdfprocess.h> /*For calling
#else
#include <sys/param.h>
#define _spawnlp Spawnlp /* defined in var.c */
#define _P_WAIT 0
#endif


/*_____________________________________MACROS___________________________________*

#define TOSTR(X) #X
#define TOSTR_2(X) TOSTR(X)  /*if X is a macro, this forces evaluation*/
#define MERGE2_2(X,Y) MERGE2(X,Y)
#define MERGE2(X,Y) X##_##Y
#define MERGE3_2(X,Y,Z) MERGE3(X,Y,Z)
#define MERGE3(X,Y,Z) X##_##Y##_##Z
#define MERGE4_2(X,Y,Z,T) MERGE4(X,Y,Z,T)
```

```
#define MERGE4(X,Y,Z,T) X##_##Y##_##Z##_##T
#define MERGE5_2(X,Y,Z,T,U) MERGE5(X,Y,Z,T,U)
#define MERGE5(X,Y,Z,T,U) X##_##Y##_##Z##_##T##_##U


/*_____________________________________CONST&TYPES_______________________________



#define MAX_PATH_LEN MAXPATHLEN
#define MAX_CHAR 180
#define MAX_CHAR_X3 (3 * MAX_CHAR)
#define MAX_CHAR_X4 (4 * MAX_CHAR)
#define MAX_MET 40 /*!< maximum number of methods */
#define MAX_OPT 30 /*!< maximum number of options */
#define MAX_PAR 15 /*!< maximum number of parameters for  Pricing Methods */
#define MAX_PAR_DYNAMIC_TEST 30 /*!< maximum number of parameters for dynamic te
#define MAX_METHODS 40  /*!< = max number of Pricing methods */

typedef char       Label[MAX_CHAR];

#define NO_PAR -1
#define DONOTITERATE 16

#define TOSCREEN 0
#define TOFILE 1
#define TOSCREENANDFILE 2
#define NAMEONLYTOFILE 3
#define VALUEONLYTOFILE 4
#define TOVARARRAY 5
#define ZOOMTIME 1000

/* in pnl_mathtools.h
#define OK 0
#define WRONG 1
#define FAIL 1 */
#include "pnl/pnl_mathtools.h"
#define PREMIA_NONE -1
#define NONACTIVE -1 /* to identify non free objects */
```

```
/*_____________________________________VAR_____________________________________*/
typedef struct VAR_t VAR;

#define MAX_ITERATOR 3

typedef enum { SETABLE = 0, UNSETABLE = 1 } vsetable ;

typedef struct enumeration_t enumeration;
struct enumeration_t
{
  int            value;
  struct PremiaEnum_t *members;
};

struct VAR_t
{
  const char   *Vname;
  int Vtype;
  union
  {
    int V_INT;
    int V_INT2;
    int V_RGINT130;
    int V_RGINT13;
    int V_RGINT12;
    double V_DOUBLE;
    long V_LONG;
    double V_PDOUBLE;
    double V_SPDOUBLE;
    double V_SNDOUBLE;
    double V_SDOUBLE2;
    double V_RGDOUBLE051;
    double V_RGDOUBLE005;
    double V_DATE;
    double V_RGDOUBLE;
    double V_RGDOUBLE1;
    double V_RGDOUBLEM11;
    int V_PINT;
    double V_RGDOUBLE12;
    double V_RGDOUBLE02;
    int V_BOOL;
```

```
    int V_PADE;
    double V_RGDOUBLE14;
    char *V_FILENAME;
    struct NumFunc_1 *V_NUMFUNC_1;
    struct NumFunc_2 *V_NUMFUNC_2;
    struct NumFunc_nd *V_NUMFUNC_ND;
    struct PtVar *V_PTVAR;
    PnlVect *V_PNLVECT;
    PnlVectInt *V_PNLVECTINT;
    PnlMat *V_PNLMAT;
    enumeration V_ENUM;
  } Val;
  int   Viter;
  vsetable Vsetable; /* a flag telling if a variable is to
                      * be set or get  interactively */
  void (*setter)(void *); /* if not null, points to a setter function accepting
                           * a pointer to a TypeModel or a TypeOpt.  Should be
                           * called immidiately after the field has been
                           * changed */
};

/*
 * Vtype
 */
#define FIRSTLEVEL 29 /* first level types are stricly
                       * smaller than FIRSTLEVEL */

/*FirstClass*/
#define PREMIA_NULLTYPE 0
#define INT 1
#define DOUBLE 2
#define LONG 3
#define PDOUBLE 4
#define DATE 5
#define RGDOUBLE 6
#define BOOL 7
#define PADE 8
#define RGDOUBLE12 9
#define INT2 10
#define RGINT13 11
#define RGINT12 12
```

```
#define SPDOUBLE 13
#define RGDOUBLE051 14
#define RGDOUBLE005 15
#define RGDOUBLE14 18
#define RGDOUBLEM11 19
#define PINT 20
#define RGDOUBLE1 21
#define RGDOUBLE02 22
#define FILENAME 24
#define ENUM  25
#define RGINT130  26
#define SNDOUBLE 27
#define SDOUBLE2 28

/*SecondClass*/
#define NUMFUNC_1 29
#define NUMFUNC_2 30
#define NUMFUNC_ND 31
#define PTVAR 32
#define PNLVECT 33
#define PNLVECTINT 34
#define PNLMAT 35
/*This last type should be less than MAX_TYPE:*/

#define MAX_TYPE 40
/*Viter*/
#define IRRELEVANT -3
#define FORBID -2
#define ALLOW   -1
#define ALREADYITERATED 256
/*MAX_ITERATOR should be less than ALREADYITERATED*/

/*Useful Flags*/

#undef IN
#undef OUT
#define EURO 0
#define AMER 1
#define TOTAL 0
#define PARTIAL 1
#define CONT 0
```

```
#define DISC 1
#define OUT 0
#define IN 1
#define DOWN 0
#define UP 1
#define REBATE 0
#define NOREBATE 1
#define CONSTLIM 0
#define MOVLIM 1
#define TIMEAVERAGING 10

/*_____________________________________PLANNING_____________________________________

#define MAX_ITER 1000

typedef struct Iterator
{
  VAR    *Location;
  VAR     Min;
  VAR     Max;
  VAR     Default;
  int         StepNumber;
} Iterator;

typedef struct
{
  char       Action;
  int         NumberOfMethods;
  Iterator    Par[MAX_ITERATOR];
  int         VarNumber;
} Planning;

/*SecondLevelVars*/
/*Arrays of VAR*/

typedef struct PtVar
{
  VAR Par[MAX_PAR];
} PtVar;
```

```
/*NumericalFunctions*/

typedef struct NumFunc_1
{
  double (*Compute)(VAR *, double);
  VAR Par[MAX_PAR];
  int (*Check)(int user, Planning *, void *);
} NumFunc_1;

typedef struct NumFunc_2
{
  double (*Compute)(VAR *, double, double);
  VAR Par[MAX_PAR];
  int (*Check)(int user, Planning *, void *);
} NumFunc_2;

typedef struct NumFunc_nd
{
  double (*Compute)(VAR *, const PnlVect *);
  VAR Par[MAX_PAR];
  int (*Check)(int user, Planning *, void *);
} NumFunc_nd;


/*_____________________________________MODELS_____________________________________*

typedef struct  Model
{
  Label       ID;
  const char *Name;
  void       *TypeModel;
  int (*Get)(int user, Planning *, struct Model *);
  int (*FGet)(char **InputFile, int user, Planning *, struct Model *);
  int (*Show)(int user, Planning *, struct Model *);
  int (*Check)(int user, Planning *, struct Model *);
  int (*Init)(struct Model *);
  int         nvar; /* number of VARS in TypeModel */
  int         init; /* zero before initialization */
  /*  if HelpFilenameHint == NULL PDF file with documentation for the model can
  /*  otherwise the path to the documentation is doc/pdf_html/mod/%HelpFilenameH
```

```
  const char *HelpFilenameHint;
} Model;

#define MOD(X) MERGE2_2(TYPEMOD,X)
#define MAKEMOD(X)  MAKEMODEL(TOSTR_2(TYPEMOD), X)
#define MAKEMOD_FULL(X)  MAKEMODEL_FULL(TOSTR_2(TYPEMOD), X)
#define MAKEMODEL(Z,X) Model MOD(model)={Z ,#X,& X,Get_model_gen,FGet_model_gen,
Show_model_gen,chk_model_gen,MOD(Init), 0, 0, 0}
#define MAKEMODEL_FULL(Z,X) Model MOD(model)=\
    {Z ,#X,& X,MOD(Get),MOD(FGet), Show_model_gen,MOD(Check),MOD(Init), 0, 0, 0}

/*_____________________________________OPTIONS_____________________________________

typedef struct Option
{
  Label           ID;
  const char     *Name;
  void           *TypeOpt;
  int (*Get)(int user, Planning *, struct Option *, Model *);
  int (*FGet)(char **InputFile, int user, Planning *, struct Option *, Model *);
  int (*Show)(int user, Planning *, struct Option *, Model *);
  int (*Check)(int user, Planning *, struct Option *);
  int (*Init)(struct Option *, Model *);
  int             nvar; /* number of VARS  */
  int             init; /* zero before initialization */
  int             nvar_setable; /* number of VARS which are asked interactively
  /*  if HelpFilenameHint == NULL PDF file with documentation for the option can
  /*  otherwise the path to the documentation is doc/pdf_html/opt/%ID%/%HelpFile
  const char     *HelpFilenameHint;
} Option;

#define OPT(X) MERGE2_2(TYPEOPT,X)
#define MAKEOPT(X)  MAKEOPTION(TOSTR_2(TYPEOPT), X)
#define MAKEOPT_FULL(X)  MAKEOPTION_FULL(TOSTR_2(TYPEOPT), X)
/* #define MAKEOPTION(Z,X) Option OPT( X)={Z ,#X,& X,OPT(Get),OPT(FGet),OPT(Show
#define MAKEOPTION(Z,X) Option OPT( X)={Z ,#X,& X,Get_option_gen,FGet_option_gen
#define MAKEOPTION_FULL(Z,X) Option OPT( X)={Z ,#X,& X,OPT(Get),OPT(FGet),OPT(Sh

typedef Option *Family[MAX_OPT];

/*_____________________________________PRICINGS & DYNAMIC TESTS_________________
```

```
/*Pricing Methods*/
typedef struct PricingMethod
{
  const char                          *Name;
  VAR                                  Par[MAX_PAR];
  int (*Compute)(void *, void *, struct PricingMethod *);
  VAR                                  Res[MAX_PAR];
  int (*CheckOpt)(void *, void *);
  int (*Check)(int user, Planning *, void *);
  int (*Init)(struct PricingMethod *, Option *);
  int                                  init; /* zero before initialization */
  /* if HelpFilenameHint == NULL PDF file with documentation for the
   * pricing method can be found at
   * doc/pdf_html/mod/%Model%/%Model%/%Family%/%Name%_doc.pdf
   * otherwise the path to the documentation is
   * doc/pdf_html/mod/%Model%/%Model%/%Family%/%HelpFilenameHint%_doc.pdf
   * where Model is "corrected" model name for the pricing (see
   * Model::HelpFilenameHint)
   * Family is family name for the pricing
   */
  const char  *HelpFilenameHint;
} PricingMethod;

#define MET(X) MERGE3_2(TYPEMOD,TYPEOPT,X)
#define CALC(X) MERGE4_2(CALC,TYPEMOD,TYPEOPT,X)

/*Dynamic Tests*/

typedef struct  DynamicTest
{
  const char                           *Name;
  VAR Par[MAX_PAR_DYNAMIC_TEST];
  int (*Simul)(void *, void *, PricingMethod *Met, struct DynamicTest *);
  VAR Res[MAX_PAR_DYNAMIC_TEST];
  int (*CheckTest)(void *, void *, PricingMethod *Met);
  int (*Check)(int user, Planning *, void *);
  int (*Init)(struct DynamicTest *, Option *);
} DynamicTest ;
```

```
#define TEST(X) MERGE3_2(TYPEMOD,TYPEOPT,X)

typedef struct Pricing
{
  Label                 ID;
  PricingMethod    **Methods;
  DynamicTest **Test;
  int (*CheckMixing)(Option *, Model *);
} Pricing;



#define MOD_OPT(X) MERGE3_2(TYPEMOD,TYPEOPT,X)
#define CHK_OPT(X) MERGE4_2(CHK_OPT,TYPEMOD,TYPEOPT,X)
#define ID_MOD_OPT TOSTR_2(MERGE2_2(TYPEMOD,TYPEOPT))
#define CHK_TEST(X) MERGE5_2(CHK_TEST,TYPEMOD,TYPEOPT,MET,X)
/*Time Info*/

typedef struct TimeInfo
{
  Label                         Name;
  VAR Par[MAX_PAR];
  VAR Res[MAX_PAR];
  int (*Check)(int user, Planning *, struct TimeInfo *);
  int (*Init)(struct TimeInfo *);
} TimeInfo;

#endif
```