Help

```
#include "
href../common/optype_h_src.pdfoptype.h"
#include "
href../common/enums_h_src.pdfenums.h"
#include "pnl/pnl_random.h"
#include "pnl/pnl_basis.h"


static PremiaEnumMember NullEnumMembers[] =
{
  { NULL, NULLINT, 0}
};

static PremiaEnumMember BooleanMembers[] =
{
  { "No",  0, 0 },
  { "Yes", 1, 0 },
  { NULL, NULLINT, 0}
};

static PremiaEnumMember CirOrderMembers[] =
{
  { "Second Order for the CIR", 1 , 0},
  { "Third Order for the CIR", 2 , 0},
  { NULL, NULLINT , 0}
};

static PremiaEnumMember afd_members[] =
{
  { "Terminal Measure", 0 , 0},
  { "Spot Measure", 1 , 0},
  { NULL, NULLINT , 0}
};

static PremiaEnumMember averaging_members[] =
{
  { "Averaged Vol", 0 , 0},
  { "Time-Dep Vol", 1 , 0},
  { NULL, NULLINT , 0}
```

```
};

static PremiaEnumMember boundary_cond_members[] =
{
  {"Dirichlet", 0, 0},
  {"Andreasen", 1, 0},
  { NULL, NULLINT , 0}
};

static PremiaEnumMember DiscretizationScheme_members[] =
{
  { "Exact Scheme for Wishart and Weak Scheme for Stock", 1 , 0},
  { "Weak Scheme for Stock and Wishart", 2 , 0},
  { NULL, NULLINT , 0}
};

static PremiaEnumMember PrecondMembers[] =
{
  { "Diagonal", 1 , 0},
  { "ILU", 2 , 0},
  { NULL, NULLINT , 0}
};

static PremiaEnumMember schemetreenig_members[] =
{
  { "Improved Scheme", 1 , 0},
  { "MSS Scheme", 2 , 0},
  { NULL, NULLINT , 0}
};

static PremiaEnumMember exp_part_members[] =
{
  { "Decentered", 1 , 0},
  { "Centered", 2 , 0},
  { NULL, NULLINT , 0}
};

static PremiaEnumMember DeltaMethodMembers[] =
{
  { "Finite Difference", 1 , 0},
  { "Malliavin", 2 , 0},
```

```
  { "Malliavin Local", 3 , 0},
  { NULL, NULLINT , 0}
};

static PremiaEnumMember IntegralSchemeMembers[] =
{
  { "Riemann", 1 , 0},
  { "Trapezoidal", 2 , 0},
  { "Brownian Bridge", 3 , 0},
  { NULL, NULLINT , 0}
};

static PremiaEnumMember PnlBasisMembers [] =
{
  { "Canonical", PNL_BASIS_CANONICAL, 0},
  { "Hermite", PNL_BASIS_HERMITIAN, 0},
  { "Tchebychev", PNL_BASIS_TCHEBYCHEV, 0},
  { NULL, NULLINT, 0},
};

/*
 * Random Number Generator Array
 */
static PremiaEnumMember PnlRngMembers[] =
{
  {"KNUTH", PNL_RNG_KNUTH, 0},
  {"MRGK3", PNL_RNG_MRGK3, 0},
  {"MRGK5", PNL_RNG_MRGK5, 0},
  {"SHUFL", PNL_RNG_SHUFL, 0},
  {"L'ECUYER", PNL_RNG_LECUYER, 0},
  {"TAUSWORTHE", PNL_RNG_TAUSWORTHE, 0},
  {"MERSENNE", PNL_RNG_MERSENNE, 0},
  {"MERSENNE (Random Seed)", PNL_RNG_MERSENNE_RANDOM_SEED, 0},
  {"SQRT", PNL_RNG_SQRT, 0},
  {"HALTON", PNL_RNG_HALTON, 0},
  {"FAURE", PNL_RNG_FAURE, 0},
  {"SOBOL_I4", PNL_RNG_SOBOL_I4, 0},
  {"SOBOL_I8", PNL_RNG_SOBOL_I8, 0},
  {"NIEDERREITER", PNL_RNG_NIEDERREITER, 0},
  {NULL, NULLINT, 0}
};
```

```
/*
 * True MC generators do not take into account the parameter dimension in the
 * Compute function.
 */
static PremiaEnumMember PnlRngMCMembers[] =
{
  {"KNUTH", PNL_RNG_KNUTH, 0},
  {"MRGK3", PNL_RNG_MRGK3, 0},
  {"MRGK5", PNL_RNG_MRGK5, 0},
  {"SHUFL", PNL_RNG_SHUFL, 0},
  {"L'ECUYER", PNL_RNG_LECUYER, 0},
  {"TAUSWORTHE", PNL_RNG_TAUSWORTHE, 0},
  {"MERSENNE", PNL_RNG_MERSENNE, 0},
  {"MERSENNE (Random Seed)", PNL_RNG_MERSENNE_RANDOM_SEED, 0},
  {NULL, NULLINT, 0}
};

static PremiaEnumMember flat_members[] =
{
  {"Flat ZCB Prices", 0, 1},
  {"No_Flat ZCB Prices", 1, 1},
  { NULL, NULLINT, 0}
};

static PremiaEnumMember flat_members_copy[] =
{
  {"Flat ZCB Prices", 0, 1},
  {"No_Flat ZCB Prices", 1, 1},
  { NULL, NULLINT, 0}
};

static PremiaEnumMember flat_members2[] =
{
  {"Flat ZCB Prices", 0, 1},
  {"No_Flat ZCB Prices", 1, 2},
  { NULL, NULLINT, 0}
};

static PremiaEnumMember volatility_members[] =
{
```

```
  { "15/x", 0, 0 },
  { "0.01+0.1*exp(-x/100)+0.01*t", 1, 0 },
  { NULL, NULLINT, 0 }
};

static PremiaEnumMember is_type_members[] =
{
  { "Gaussian only", 1, 0 },
  { "Jump only", 2, 0 },
  { "Gaussian and Jump", 3, 0 },
  { NULL, NULLINT, 0 }
};

static PremiaEnumMember clusteringMethod_members[] =
{
  { "kmeans", 1 , 0},
  { "recursive bifurcation", 2 , 0},
  { "recursive bifurcation reduced state space", 3 , 0},
  { NULL, NULLINT , 0}
};

static PremiaEnumMember clusteringMethod1_members[] =
{
  { "kmeans", 1 , 0},
  { "recursive bifurcation", 2 , 0},
  { NULL, NULLINT , 0}
};

static PremiaEnumMember regression_type_members[] =
{
    {"Original method (polynomial of the assets)", 1, 0},
    {"Matching Projection Pursuit", 2, 0},
    {"Orthogonal Matching Pursuit", 3, 0},
    { NULL, NULLINT , 0}
};



DEFINE_ENUM(PremiaEnumNull, NullEnumMembers);
DEFINE_ENUM(PremiaEnumBool, BooleanMembers);
DEFINE_ENUM(PremiaEnumCirOrder, CirOrderMembers);
```

```
DEFINE_ENUM(PremiaEnumAfd, afd_members);
DEFINE_ENUM(PremiaEnumAveraging, averaging_members);
DEFINE_ENUM(PremiaEnumBoundaryCond, boundary_cond_members);
DEFINE_ENUM(PremiaEnumDiscretizationScheme, DiscretizationScheme_members);
DEFINE_ENUM(PremiaEnumPrecond, PrecondMembers);
DEFINE_ENUM(PremiaEnumSchemeTreeMSS, schemetreenig_members);
DEFINE_ENUM(PremiaEnumExpPart, exp_part_members);
DEFINE_ENUM(PremiaEnumDeltaMC, DeltaMethodMembers);
DEFINE_ENUM(PremiaEnumIntegralScheme, IntegralSchemeMembers)
DEFINE_ENUM(PremiaEnumRNGs, PnlRngMembers);
DEFINE_ENUM(PremiaEnumMCRNGs, PnlRngMCMembers);
DEFINE_ENUM(PremiaEnumBasis, PnlBasisMembers);
DEFINE_ENUM(PremiaEnumFlat, flat_members);
DEFINE_ENUM(PremiaEnumFlatCopy, flat_members_copy);
DEFINE_ENUM(PremiaEnumFlat2, flat_members2);
DEFINE_ENUM(PremiaEnumVolatility, volatility_members);
DEFINE_ENUM(PremiaEnumISType, is_type_members);
DEFINE_ENUM(PremiaEnumclusteringMethod, clusteringMethod_members);
DEFINE_ENUM(PremiaEnumclusteringMethod1, clusteringMethod1_members);
DEFINE_ENUM(PremiaEnumRegressionType, regression_type_members);
```