```
   Help
#if defined(PremiaCurrentVersion) && PremiaCurrentVersion <
     (2008+2) //The "#else" part of the code will be freely av
    ailable after the (year of creation of this file + 2)
#else
/*************************************************************
    ******************/
/*                                    highdim_matrix.h
                                  */
/*************************************************************
    ******************/
/*
                       */
/* type MATRIX
                       */
/*
                       */
/* Copyright (C) 1992-1995 Tomas Skalicky. All rights res
    erved.              */
/*
                       */
/*************************************************************
    ******************/
/*
                       */
/*        ANY USE OF THIS CODE CONSTITUTES ACCEPTANCE OF TH
    E TERMS            */
/*               OF THE COPYRIGHT NOTICE (SEE FILE copyrght.h
    )                 */
/*
                       */
/*************************************************************
    ******************/

#ifndef HIGHDIM_MATRIX_H
#define HIGHDIM_MATRIX_H

#include <stdlib.h>

#include "lastypes.h"
#include "elcmp.h"
```

```
#include "copyrght.h"

typedef struct {
    char *Name;
    size_t RowDim;
    size_t ClmDim;
    ElOrderType ElOrder;
    InstanceType Instance;
    int LockLevel;
    double Multipl;
    Boolean OwnData;
    size_t *Len;
    ElType **El;
    Boolean *ElSorted;
} Matrix;

void M_Constr(Matrix *M, char *Name, size_t RowDim, size_t
    ClmDim,
              ElOrderType ElOrder, InstanceType Instance,
    Boolean OwnData);
void M_Destr(Matrix *M);
void M_SetName(Matrix *M, char *Name);
char *M_GetName(Matrix *M);
size_t M_GetRowDim(Matrix *M);
size_t M_GetClmDim(Matrix *M);
ElOrderType M_GetElOrder(Matrix *M);
void M_SetLen(Matrix *M, size_t RoC, size_t Len);
size_t M_GetLen(Matrix *M, size_t RoC);
void M_SetEntry(Matrix *M, size_t RoC, size_t Entry, size_
    t Pos, double Val);
size_t M_GetPos(Matrix *M, size_t RoC, size_t Entry);
double M_GetVal(Matrix *M, size_t RoC, size_t Entry);
void M_AddVal(Matrix *M, size_t RoC, size_t Entry, double
    Val);

/* macros for fast access */
#define     M__GetLen(PtrM, RoC)              (PtrM)->Len[
    RoC]
#define     M__SetEntry(PtrM, RoC, Entry, Pos_, Val_) { {
                (PtrM)->El[RoC][Entry].Pos = (Pos_); {
                (PtrM)->El[RoC][Entry].Val = (Val_); {
```

```
            }
#define     M__GetPos(PtrM, RoC, Entry)        (PtrM)->El[
    RoC][Entry].Pos
#define     M__GetVal(PtrM, RoC, Entry)        (PtrM)->El[
    RoC][Entry].Val
#define     M__AddVal(PtrM, RoC, Entry, Val_) { {
                (PtrM)->El[RoC][Entry].Val += (Val_); {
            }

double M_GetEl(Matrix *M, size_t Row, size_t Clm);

void M_SortEl(Matrix *M);

void M_Lock(Matrix *M);
void M_Unlock(Matrix *M);

#endif /* HIGHDIM_MATRIX_H */

#endif //PremiaCurrentVersion
```

# References