```
   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
/*************************************************************
    ******************/
/*                                    qmatrix.h
                     */
/*************************************************************
    ******************/
/*
                     */
/* type QMATRIX
                     */
/*
                     */
/* 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 QMATRIX_H
#define QMATRIX_H

#include <stdlib.h>

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

```
#include "highdim_vector.h"
#include "copyrght.h"

typedef struct QMatrixType {
    char *Name;
    size_t Dim;
    Boolean Symmetry;
    ElOrderType ElOrder;
    InstanceType Instance;
    int LockLevel;
    double MultiplD;
    double MultiplU;
    double MultiplL;
    Boolean OwnData;
    size_t *Len;
    ElType **El;
    Boolean *ElSorted;
    Boolean *DiagElAlloc;
    ElType **DiagEl;
    Boolean *ZeroInDiag;
    double *InvDiagEl;
    Boolean UnitRightKer;
    double *RightKerCmp;
    Boolean UnitLeftKer;
    double *LeftKerCmp;
    void *EigenvalInfo;
    Boolean *ILUExists;
    struct QMatrixType *ILU;
} QMatrix;

void Q_Constr(QMatrix *Q, char *Name, size_t Dim, Boolean
    Symmetry,
              ElOrderType ElOrder, InstanceType Instance,
    Boolean OwnData);
void Q_Destr(QMatrix *Q);
void Q_SetName(QMatrix *Q, char *Name);
char *Q_GetName(QMatrix *Q);
size_t Q_GetDim(QMatrix *Q);
Boolean Q_GetSymmetry(QMatrix *Q);
ElOrderType Q_GetElOrder(QMatrix *Q);
void Q_SetLen(QMatrix *Q, size_t RoC, size_t Len);
```

```
size_t Q_GetLen(QMatrix *Q, size_t RoC);
void Q_SetEntry(QMatrix *Q, size_t RoC, size_t Entry, size_
    t Pos, double Val);
size_t Q_GetPos(QMatrix *Q, size_t RoC, size_t Entry);
double Q_GetVal(QMatrix *Q, size_t RoC, size_t Entry);
void Q_AddVal(QMatrix *Q, size_t RoC, size_t Entry, double
    Val);

/* macros for fast access */
#define     Q__GetLen(PtrQ, RoC)              (PtrQ)->Len[
    RoC]
#define     Q__SetEntry(PtrQ, RoC, Entry, Pos_, Val_) { {
                (PtrQ)->El[RoC][Entry].Pos = (Pos_); {
                (PtrQ)->El[RoC][Entry].Val = (Val_); {
            }
#define     Q__GetPos(PtrQ, RoC, Entry)       (PtrQ)->El[
    RoC][Entry].Pos
#define     Q__GetVal(PtrQ, RoC, Entry)       (PtrQ)->El[
    RoC][Entry].Val
#define     Q__AddVal(PtrQ, RoC, Entry, Val_) {
                (PtrQ)->El[RoC][Entry].Val += (Val_)

double Q_GetEl(QMatrix *Q, size_t Row, size_t Clm);

void Q_SortEl(QMatrix *Q);
void Q_AllocInvDiagEl(QMatrix *Q);

void Q_SetKer(QMatrix *Q, Vector *RightKer, Vector *LeftK
    er);
Boolean Q_KerDefined(QMatrix *Q);

void **Q_EigenvalInfo(QMatrix *Q);

void Q_Lock(QMatrix *Q);
void Q_Unlock(QMatrix *Q);

#endif /* QMATRIX_H */

#endif //PremiaCurrentVersion
```

References