```
   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

#ifndef _NR_UTILS_H_
#define _NR_UTILS_H_

/*static double sqrarg;
#define SQR(a) ((sqrarg=(a)) == 0.0 ? 0.0 : sqrarg*sqrarg)

static double dsqrarg;
#define DSQR(a) ((dsqrarg=(a)) == 0.0 ? 0.0 : dsqrarg*dsqr
    arg)

static double dmaxarg1,dmaxarg2;
#define DMAX(a,b) (dmaxarg1=(a),dmaxarg2=(b),(dmaxarg1) > (
    dmaxarg2) ?{
        (dmaxarg1) : (dmaxarg2))

static double dminarg1,dminarg2;
#define DMIN(a,b) (dminarg1=(a),dminarg2=(b),(dminarg1) < (
    dminarg2) ?{
        (dminarg1) : (dminarg2))


static double maxarg1,maxarg2;
#define FMAX(a,b) (maxarg1=(a),maxarg2=(b),(maxarg1) > (max
    arg2) ?{
        (maxarg1) : (maxarg2))

static double minarg1,minarg2;
#define FMIN(a,b) (minarg1=(a),minarg2=(b),(minarg1) < (mi
    narg2) ?{
        (minarg1) : (minarg2))

static long lmaxarg1,lmaxarg2;
#define LMAX(a,b) (lmaxarg1=(a),lmaxarg2=(b),(lmaxarg1) > (
    lmaxarg2) ?{
        (lmaxarg1) : (lmaxarg2))
```

```
static long lminarg1,lminarg2;
#define LMIN(a,b) (lminarg1=(a),lminarg2=(b),(lminarg1) < (
    lminarg2) ?{
        (lminarg1) : (lminarg2))

static int imaxarg1,imaxarg2;
#define IMAX(a,b) (imaxarg1=(a),imaxarg2=(b),(imaxarg1) > (
    imaxarg2) ?{
        (imaxarg1) : (imaxarg2))

static int iminarg1,iminarg2;
#define IMIN(a,b) (iminarg1=(a),iminarg2=(b),(iminarg1) < (
    iminarg2) ?{
 (iminarg1) : (iminarg2))*/

#define SIGN(a,b) ((b) >= 0.0 ? fabs(a) : -fabs(a))

void nrerror(char error_text[]);
double *vector(long nl, long nh);
int *ivector(long nl, long nh);
unsigned char *cvector(long nl, long nh);
unsigned long *lvector(long nl, long nh);
double *dvector(long nl, long nh);

double **matrix(long nrl, long nrh, long ncl, long nch);
double **dmatrix(long nrl, long nrh, long ncl, long nch);
int **imatrix(long nrl, long nrh, long ncl, long nch);
double **submatrix(double **a, long oldrl, long oldrh, lon
    g oldcl, long oldch,
  long newrl, long newcl);
double **convert_matrix(double *a, long nrl, long nrh, lon
    g ncl, long nch);
double ***f3tensor(long nrl, long nrh, long ncl, long nch,
    long ndl, long ndh);
void free_vector(double *v, long nl, long nh);
void free_ivector(int *v, long nl, long nh);

void free_cvector(unsigned char *v, long nl, long nh);
void free_lvector(unsigned long *v, long nl, long nh);
void free_dvector(double *v, long nl, long nh);
```

```
void free_matrix(double **m, long nrl, long nrh, long ncl,
    long nch);
void free_dmatrix(double **m, long nrl, long nrh, long ncl,
     long nch);
void free_imatrix(int **m, long nrl, long nrh, long ncl,
    long nch);
void free_submatrix(double **b, long nrl, long nrh, long nc
    l, long nch);
void free_convert_matrix(double **b, long nrl, long nrh,
    long ncl, long nch);
void free_f3tensor(double ***t, long nrl, long nrh, long nc
    l, long nch,
  long ndl, long ndh);

#endif /* _NR_UTILS_H_ */

#endif //PremiaCurrentVersion
```

References