

```
#if defined(PremiaCurrentVersion) && PremiaCurrentVersion < (2007+2) //The "#els
#else
```

```
#include<stdio.h>
#include<stdlib.h>
#include <math.h>
#include "cirpp1d_stdi.h"
```

[illegible]

static double bond(double T);	Return the ZC bond value $P(0,T)$
static double Shift(double s);	Return the Shift of the model (σ)

```

////////////////////////////////////
//////////////////////////////////// Structure PDE and TREE for CIR++
////////////////////////////////////

```

```

struct Tree
{
    double *t;          /*Time step grid, from t[0] to T[Ngrid].*/
    double Tf;          /*Final time time of the tree, dt=Tf/Ngrid*/
    int Ngrid;          /*Numter of time step in the Tree, R[0][] is the first and
    double **Payoffunc; /*Vector Payoff for the tree (see the functions initPayoff

    double P_T;         /*The value of the Z-C bond P(0,T)*/
    double **pLRij;     /*The value of the short rates in the tree*/
    double **pLQij;     /*The value of the Options or other things (depend on Payo
    double **pLPDo;     /*Transition proba. in the trinomial tree for the down poi
    double **pLPMi;     /*Transition proba. in the trinomial tree for the midle po
    double **pLPUp;     /*Transition proba. in the trinomial tree for the uper poi
    int **pLRef;        /*Reference index for the midle point of the next time ste
    int *TSize;         /*Size of the scale rate at given time step.*/
};

struct EDP
{
    double *t;          /*Time step grid, from t[0] to T[Ngrid].*/
    double Tf;          /*Final time time of the tree, dt=Tf/Ngrid*/
    int Ngrid;          /*Numter of time step in the Tree, R[0] is the first and R[

    double dx;          /*Pas d'espace (rate axis)*/
    double dt;          /*Time step*/
    int nx;             /*nombre de point d'espace*/
    double Rm;

    double **M1;        /*Matrice de l'EDP a inverser*/
    double **M2;        /*Matrice du second membre de d'EDP*/
    double **Payoffunc; /*Matrix Payoff in space and time or Matrix solution*/
};

int SetTimegrid_EDP(struct EDP *Meth, int n, double T);
double OPTIONr_EDP(struct EDP *Meth, double r, double s, double T0);
void initPayoff1_EDP(struct EDP *Meth, double T0);

double OPTIONr_tr(struct Tree *Meth, double r, double s);
void initPayoff1_tr(struct Tree *Meth, double T0);
void freePayoff1_tr(struct Tree *Meth, double T0);

```

```
#endif  
#endif //PremiaCurrentVersion
```