```
   Help
#if defined(PremiaCurrentVersion) && PremiaCurrentVersion <
     (2007+2) //The "#else" part of the code will be freely av
    ailable after the (year of creation of this file + 2)
#else

/// {file cdsmkt.h
/// {brief CDS_NoCorr_MarketData class
/// {author  M. Ciuca (MathFi, ENPC)
/// {note (C) Copyright Premia 8 - 2006, under Premia 8 Sof
    tware license
//
//  Use, modification and distribution are subject to the
//  Premia 8 Software license

#ifndef _CDSMKT_H
#define _CDSMKT_H

#include "cirpp.h"
#include "numint.h"
#include "base.h"


// forward declaration
struct DateCreal;

class PConstShortRate
{
public:
  PConstShortRate(string inputFileName, string outputFil
    eName="");
  PConstShortRate(vector<double>& RatesMat, vector<double>
    & Rates);
  double ComputeShortRate(double t) const;
  double ComputeZC(double t) const;
  double f0_t(double t) const;
private:
  vector<DateCreal> _curveZC;
  vector<DateCreal> _pConstShortRate;
  string _inputFileName;
  int _dim;
```

```
  void ReadData(string fileName);
  void ReadData(vector<double>& pMat, vector<double>& pRa
    tes);
};


struct DateCreal
{
  //public:
  double date;
  double r;
  DateCreal(double _date=0, double _r=1):
  date(_date), r(_r)
  {
    if((date < 0) || (r <= 0))
    {
      throw logic_error("DateCreal Constructor: Incor
    rect input data!");
    }
  }
};




class CDS_NoCorr_MarketData
{
public:
  CDS_NoCorr_MarketData(double Z, vector<double>& timesT,
    string inputIntensity,
    string inputZC);

  CDS_NoCorr_MarketData(vector<double>& intensityMat, vec
    tor<double>& intensityRates,
    vector<double>& RatesMat, vector<double>& Rates,
    double maturity, double period,
    double recovery);


  double CdsRate()
  { return CdsRate(_timesT[_timesT.size() - 1], _periodN);
```

```
    }
  double CdsRate(double& paymentLeg, double& defaultLeg)
  { return CdsRate(_timesT[_timesT.size() - 1], _periodN,
    paymentLeg, defaultLeg); }
  double CdsRate(double T, int noTi);
  double CdsRate(double T, int noTi, double& paymentLeg,
    double& defaultLeg);
  double CdsRate(double T, int noTi, double& I1, double&
    I2, double& S);

  double MarketZC(double t) const {
    return _pConstShortRate.ComputeZC(t);
  }
  double CDS(double T, int noTi, double Rf);

  double DP(double t) const
  { return 1-exp( -IntegralPLin(t) ); }
  double GetMaturity() { return _timesT[_timesT.size() - 1
    ]; }
private:
  double _Z;
  vector<DateRate> _pLinIntensity;
  vector<DateRate> _curveZC;
  int _periodN;
  vector<double> _timesT;
  double I1;
  double I2;
  double S;
  PConstShortRate _pConstShortRate;
  void ReadData(vector<DateRate>& data, string fileName);
  void ReadData(vector<DateRate>& data,
    vector<double>& pMat, vector<double>& pRates);
  void Write(vector<DateRate>& data, string outputFileNa
    me);
  double IntegralPLin(double t) const;
  double ComputeIntensity(double t) const;

  double f1(double u);
  double f2(double u);
  double f_Sum(int n0, int n) const;
```

```
  NumInt<CDS_NoCorr_MarketData> numInt;
};


#endif

#endif //PremiaCurrentVersion
```

References