Help

```
#include <iostream>
#include <cmath>
#include <cstdlib>
#include <cstring>
#include <algorithm>

#include "
href../../../../common/math/mcam/src/performance_h_src.pdfperformance.hpp"
#include "
href../../../../common/math/jlparser/include/jlparser/parser_h_src.pdfjlparser/p
#include "pnl/pnl_vector.h"
#include "pnl/pnl_matrix.h"

//
// Performance options
//

double mcam::PerformanceOption::payoff(const PnlMat *S, int t)
{
    PnlVect S0 = pnl_vect_wrap_mat_row(S, 0);
    double panier_prev = pnl_vect_scalar_prod(lambda, &S0);
    double sum = 0.;
    for (int i = 1 ; i <= t ; i++)
    {
        PnlVect St = pnl_vect_wrap_mat_row(S, i);
        double panier = pnl_vect_scalar_prod(lambda, &St);
        sum += fmax(panier / panier_prev - 1., 0.);
        panier_prev = panier;
    }
    return(1. + sum);
}

mcam::PerformanceOption::PerformanceOption ()
{
    lambda = NULL;
}

mcam::PerformanceOption::PerformanceOption (const Param &P)
    : Option(P)
```

```
{
    label = "performance";
    P.extract("payoff coefficients", lambda, size);
}

void mcam::PerformanceOption::print() const
{
    std::cout << std::endl;
    std::cout << "**************************************" << std::endl;
    std::cout << "**** Performance Option Characteristics ****" << std::endl;
    mcam::Option::print();
    std::cout << " payoff coefficients : ";
    pnl_vect_print_asrow(lambda);
    std::cout << " strike : " << K << std::endl;
    std::cout << "**************************************" << std::endl << std::
}

mcam::PerformanceOption::~PerformanceOption ()
{
    if (lambda)
        pnl_vect_free(&lambda);
}
```