

## Help

```
/* Nothing is printed to files if the number of samples
 * exceeds 100000. Max File size is exceeded otherwise. */
#include <iostream>
#include <fstream>
#include <cmath>
#include <cstdlib>
#include <string>

#include "math/ImportanceSampling_jl/src/Option.hpp"
#include "math/ImportanceSampling_jl/src/AsianOption.hpp"
#include "math/ImportanceSampling_jl/src/BarrierOption.hpp"
#include "math/ImportanceSampling_jl/src/BasketOption.hpp"
#include "math/ImportanceSampling_jl/src/MountainOption.hpp"
#include "math/ImportanceSampling_jl/src/MaxOption.hpp"
#include "math/ImportanceSampling_jl/src/PremiaOption.hpp"

using namespace std;

BaseOption::BaseOption() { }

/**
 * Creates the correct option according to name
 */
BaseOption *instantiate_option(const Param &P)
{
    BaseOption *opt = NULL;
    string optionType;
    P.extract("option type", optionType);

    if (optionType == "exchange" || optionType == "basket")
        opt = new BasketOption(P);
    else if (optionType == "geometriccall")
        opt = new GeometricBasketOption(P, true);
    else if (optionType == "geometricput")
        opt = new GeometricBasketOption(P, false);
    else if (optionType == "atlas")
        opt = new AtlasOption(P);
    else if (optionType == "altiplano")
        opt = new AltiplanoOption(P);
```

```

    else if (optionType == "everest")
        opt = new EverestOption(P);
    else if (optionType == "digital")
        opt = new DigitalOption(P);
    else if (optionType == "digitalfinal")
        opt = new DigitalFinalOption(P);
    else if (optionType == "digitalbasket")
        opt = new DigitalBasketOption(P);
    else if (optionType == "barrier")
        opt = new BarrierOption(P);
    else if (optionType == "asian")
        opt = new AsianCallOption(P);
    else if (optionType == "maximum")
        opt = new MaxOption(P);
    else if (optionType == "bestof")
        opt = new BestOfOption(P);
    else if (optionType == "worstof")
        opt = new WorstOfOption(P);
    else if (optionType == "performance")
        opt = new PerformanceOption(P);
    else if (optionType == "PremiaOption")
        opt = new PremiaOption(P);
    else
    {
        cout << "BaseOption " << optionType << " unknow. Abort." << endl;
        abort();
    }

    return opt;
}

```

```

BaseOption::BaseOption(const Param &P)
{
    P.extract("option size", size);
    P.extract("timestep number", nTimeSteps);
    P.extract("maturity", maturity);
}

```

```

void BaseOption::print() const
{

```

```
cout << " option size : " << size << endl;  
cout << " maturity time : " << maturity << endl;  
}
```