**Finance Object Reference Library**

**Bond**

Stores general attributes and functionality of bond types. Provides user with methods for calculating a variety of bond characteristics such as price (multiple ways) , YTM, convexity, duration etc. Attributes are currently set to public and be accessed directly however mutator and accessor methods have been provided if protection and input verification is desired.

**Constructor**

Bond()

No params

**Attributes**

p; //price

r; //yield to maturity

cvx; //convexity

dur; //duration

dur\_mac; //duration macauly

dur\_mod; //duration modified

cashflow\_times; //number of cashflow discount periods

cashflow\_amounts; //cashflow amounts by period

coupon\_times; //number of coupon discount periods

coupon\_amounts; //coupon amounts by period

principal\_amounts; //principal amounts by period

principal\_times; //number of principal discount periods

**Mutators**

.setFV(a);

.setR(b);

.setPrice(c);

.setCVX(d);

.setDur(e);

.setNumPayments(f);

.setCoupon(g);

.setPrincipalTimes(x);

.setPrincipalAmounts(x);

.setCouponTimes(x);

.setCouponAmounts(x);

.setCashflows(x);

.setCashflowAmounts(x);

.setCashflowTimes(x);

**Accessors**

.getPrice();

.getDur();

.getCvx();

.getCashflowAmounts();

.getCashflowTimes();

**Methods**

.price()

calculate bond price when term structure is flat

Must have values set for cashflow\_times, cashflow\_amounts, r.

Returns price (p).

.priceBoth()

* calculate bond price when term structure is flat, given both coupon and principals.
* Must have values set for coupon\_times, coupon\_amounts, principle\_times, principle\_amounts, r.
* No return

.duration()

* calculate the duration of a bond, simple case where the term

structure is flat.

* Must have values set for cashflow\_amounts, cash\_times, r.

.convexity()

* calculate the convexity of a bond, simple case where the term

structure is flat.

* Must have values set for cashflow\_times, cashflow\_amounts, r.
* No return

.ytm()

* calculate the convexity of a bond, simple case where the term

structure is flat, interest rate r.

* calls .price()…see required values
  + can replace .price() with .priceBoth() or .priceDiscrete()
* Must have values set for p.
* No return

.priceDiscrete()

* calculate and set bond price when term structure is flat.
* must have values set for cashflows, r, cashflow\_times.
* No return

.toString()

* displays contents of the object in string format
* return type string

**------> Option <-----**

**| |**

**Option\_AmericanBinom Option\_EuropeanBlack**

**Option** (Base Class)

Stores general attributes and functionality of all option pricing methods. Attributes are currently set to public and be accessed directly however mutator and accessor methods have been provided if protection and input verification is desired.

**Constructor**

Option() -> Base Class

No params

**Attributes**

p; //Price of call option

S; //Current stock price

X; //Strike price

r; //Interest rate

std; //Standard deviation of stock

t; //Time to maturity

type; //Set to Call or Put

**Mutators**

**.**setPrice(price);

.setStockPrice(stockPrice);

.setStrike(strikePrice);

.setR(interestRate);

.setSTD(volatility);

.setTime(timeToMaturity);

**Accessors**

.getPrice();

.getStockPrice();

.getStrike();

.getR();

.getSTD();

.getTime();

**Methods**

.toString() -> Abstract method

* requires child classes to display contents of the instantiated object in string format
* return type string

**Option\_AmericanBinom** (Extends Option)

Binomial option pricing method for American Put or Call options. Attributes are currently set to public and be accessed directly however mutator and accessor methods have been provided if protection and input verification is desired.

**Constructor**

Option\_AmericanBinom() -> Extends Option()

Calls parent class constructor

**Attributes**

n; //Number of steps

**Mutators**

**.**setN();

* sets maximum number of step iterations for pricing methods

**Accessors**

**.**getN();

**Methods**

.callPrice()

* calculates price of American Call Option using the binomial pricing method
* sets object price (p)
* No return

.putPrice()

* calculates price of American Put Option using the binomial pricing method
* sets object price (p)
* No return

.toString()

* displays contents of the object in string format
* return type string

**Option\_EuropeanBlack** (Extends Option)

Black-Scholes option pricing method for European Put or Call options. Attributes are currently set to public and be accessed directly however mutator and accessor methods have been provided if protection and input verification is desired.

**Constructor**

Option\_EuropeanBlack() -> Extends Option()

Calls parent class constructor

**Attributes**

none

**Mutators**

none

**Accessors**

none

**Methods**

.callPrice()

* calculates price of European Call Option using the Black-Scholes pricing method
* sets object price (p)
* No return

.putPrice()

* calculates price of European Put Option using the Black-Scholes pricing method
* sets object price (p)
* No return

.toString()

* displays contents of the object in string format
* return type string