Appendix A - Financial Formula[1]

Note there are many formulae available for these calculations are you are free to use any you wish.

Compound Interest – Simple savings with lump sum (no payments)

$$A = P\left(1 + \frac{r}{n}\right)^{nt}$$

Simple Saving. View The future value (n=12)

Where:

A = the future value of the investment/loan, including interest
P = the principal investment amount (the initial deposit or loan amount – present value)
r = the annual interest rate (e.g. 3.2% is 0.032)
n = the number of times that interest is compounded per unit time (this is always monthly for the purpose of this coursework, i.e. **12** times per year)
t = the time the money is invested or borrowed in **years**

Interest Rate

$$r = n\left[\left(\frac{A}{P}\right)^{\frac{1}{nt}} - 1\right]$$

Simple Saving. View The interest rate (n=12)

Principle Amount

$$P = \frac{A}{\left(1 + \frac{r}{n}\right)^{nt}}$$

Simple Saving. View The amount invested (n=12)

Formula for time

$$t = \frac{ln\left(\frac{A}{P}\right)}{n\left[ln\left(1 + \frac{r}{n}\right)\right]}$$

Simple Saving. View Time of the investment in years (n=12)

Note: that ln is the natural logarithm (this is **log()** in Swift)

Compound interest savings formula (with regular payment or contributions)

These equations assume frequency of contribution and compounding is the same (for example every month)

[1] Create and test all formula by using Swift Playgrounds before implementing these into the iOS UIKit application.

Compound interest formula (with regular contributions) for deposits made at the <u>end</u> of the period - (important note: any money given so going away from lender/saver e.g. a principle amount or payments is a negative number)

Note that the Total A = **[Compound interest for principal] + [Future value of a series]**

Compound interest for a principal amount

These two formulae need to be added together to give the total value at the end (n=12)

$$\boldsymbol{P\left(1+\frac{r}{n}\right)^{nt}}$$

Simple Saving. View With Contribution Compounded Interest (n=12)

Future value of series

$$A = PMT \times \left\{\frac{\left[\left(1+\frac{r}{n}\right)^{nt}-1\right]}{\frac{r}{n}}\right\}$$

Savings with a Contribution View The future value of the series (n=12)

Payment

$$PMT = \frac{A}{\left\{\frac{\left[\left(1+\frac{r}{n}\right)^{nt}-1\right]}{\frac{r}{n}}\right\}}$$

Savings with a Contribution View The payment (n=12)

Time to achieve a certain future value A – these assume annual compounds

The following formula will calculate the **time taken in years** to **reach an investment goal A loan.** Ln is the natural log here which log(…) in Swift.

$$t = \frac{\ln(1+\frac{rA}{PMT})}{\ln(1+r)}$$

Savings with a Contribution View The total time of the investment

Loans and Mortgages

Number of payments

The following formula will calculate the **number of monthly payments** to completely pay a loan**.**

The log in the formula is log_{10} which is log10(…) in Swift.

$$firstPart = \log\left(1-\left(\frac{P}{PMT}\times\frac{r}{12}\right)\right)$$

P is the loan amount

$$secondPart = -\log\left(\frac{r}{12} + 1\right)$$

$$number\ of\ payments = \frac{firstPart}{secondPart}$$

Loans View
The total number of payments

To get this in years simply divide the result by 12.

Mortgage Payments (or any loan) – Note that the future value is assumed to be equal to zero for mortgage and loan calculations (note that t is years and P is the amount loaned)

$$PMT = \frac{P\frac{r}{12}\left(1 + \frac{r}{12}\right)^{12t}}{(1 + \frac{r}{12})^{12t} - 1}$$

Loans View
The payment amount

Note that you do not need to calculate interest rate for loans.

Amount that can be borrowed – Present Value

$$P = \frac{PMT\left(\left(\frac{r}{12} + 1\right)^{n} - 1\right)\left(\frac{r}{12} + 1\right)^{-n}}{\frac{r}{12}}$$

Loans View
The amount that can be borrowed (present value)

Note in this case *n* is the number of payments in total. E.g. 36 payments -> 3 years.

Note: <u>always check maths for correctness</u> and test algorithms in isolation. Playgrounds in XCode is highly recommended for developing and testing the maths before deploying to an iOS app.