For this exercise, we arranged the stochastic process functionality we have been developing over the course of the semester in an object oriented manner. We have two flavors of the stochastic processes:

1. Discrete time processes e.g. Markov, random walk, poisson
2. Continuous time processes e.g. Geometric Brownian motion, Vasicek and CIR Interest rate models and Monte carlo simulations.

This categorization is based on the nature of the index variable that drive the process. For continuous time processes, we consider time as a continuous set of values whereas for discrete time process, we consider time as distinct values. Beyond this, the two categories describe the same category of processes and therefore both can inherit a common class *StochasticProcesses* which will contain functionality that is common across the two categories but specific to stochastic processes(SP). Any other reusable code that is not specific to stochastic processes e.g. mean, variance, max, build histogram et.c. will be located in Calculations class.

Any code that is specific to either of the processes will be implemented specifically in the class itself. For instance, the process to generate the vasicek set only makes sense in the continuous time process space and therefore can be implemented there only. Since the user of the program only cares about certain functionality in the code such as simulate(), buildHistogram(), we have encapsulated most of the code in these public methods that make sense to the users. Only a few methods have public access and can be accessed from outside the class.

Inheritance and polymorphism is used to a great extent here and we can see its power in simplifying code by making reuse possible and allowing for a clear separation of tasks and object definition. It also makes it easier to understand and interact with the code.