Documentation for VBASim.Python

Modified from the code provided by
http://users.iems.northwestern.edu/~nelsonb/IEMS435/code.htm

I. Classes (in Basic_Classes.py)

1. class Entity			
	Definition	a customer, a bus, a car, a bike, …	
	Attributes		
		CreationTime	the creation time of the entity
		(If necessary, more attributes can be added during the simulation)	
	Methods		
		Entity(clock)	Initializer of the object clock: the creation time of the entity
2. class FIFOQueue			
	Definition	A First-In First-Out (FIFO) queue	
	Attributes		
		ThisQueue	The list of entities in the queue.
	Methods		
		FIFOQueue()	Initializer of the object
		Len()	Return the length of the queue
		Add(X, clock)	X: the entity to add clock: the time of addition
		Remove(clock)	clock: the time of removal Return the first entity in the queue.
		Mean(clock)	Return the mean of the queue length (a continuous-time statistic). clock: the current time
3. class Resource			
	Definition	A resource that can be requested by entities	
	Attributes		
		NumberOfUnits	Total number of units
		Busy	The number of units being seized
	Methods		
		Resource()	Initializer of the object
		SetUnits(Units)	Units: the units of resource that can be requested by entities.

		Seize(Units, clock)	Return True or False Units: the units being seized (could be integer or fractional). clock: the time of seizing
		Free(Units, clock)	Return True or False Units: the units of resources being freed (could be integer or fractional) clock: the time of freeing
		Mean(clock)	Return the mean of the resource utilization (a continuous-time statistic) clock: the current time
4. class CTStat			
	Definition	A continuous-time statistic (for example, the average length of a queue)	
	Methods		
		CTStat()	Initializer of the object
		Record(X,clock)	Update the CTStat. X: the new value after update clock: the time of update (This method is only necessary for user-created CTStat. No need to call Record() for queues and resources.)
		Mean(clock)	Return the value of the continuous-time statistic, that is, the area divided by time clock: the current time
		Clear(clock)	Clear the statistic for a new simulation clock: the current time
		ClearWarmUp(clock)	Clear the statistic when the warm-up period ends clock: the current time
5. class DTStat			
	Definition	A discrete-time statistic (for example, the average waiting time for all customers)	
	Attributes		
		History	The list of all records.

	Method		
		DTStat()	Initializer of the object
		Record(X)	X: the value to record.
		Mean()	Return the value of the continuous-time statistic, that is, the mean of the observations or sum divided by number of observations
		StdDev()	Return the standard deviation of the observations
		Len()	Return the number of observations
		Clear()	Clear the statistic

II. Global functions (in VBASim.py)

1. def VBASimInit(calendar,queues,ctstats,dtstats,resources,clock):

   To initialize all the classes before each repetition.

   calendar: the event calendar

   queues: a list of queues that used in the simulation

   ctstats: a list of user-created CTStat objects.

   dtstats: a list of user-created DTStat objects.

   resources: a list of resources that are used in the simulation

   clock: the start time of the simulation

2. def ScheduleEvent(calendar,EventType, EventTime, clock):

   To schedule an event

   calendar: the event calendar

   EventType: the event type

   EventTime: the time between the event and the current time

   Clock: current time

3. def getNextEvent():

   To get the next event in the event calendar.

4. def Clearstats(queues, ctstats,dtstats,resources, clock):

To clear the statistics (after the warm-up period)

queues: a list of queues that used in the simulation

ctstats: a list of user-created CTStat objects.

dtstats: a list of user-created DTStat objects.

resources: a list of resources that are used in the simulation

clock: the time of the clearance (to make the calculation of continuous-time statistics correct.)

III. Example code (MM1_simple.py)

IV. Python implementation

Because the code is written in pure Python, you can use PyPy to speed up the simulation.