```
#!/usr/bin/env python

# Mach1.py

# Introductory SimPy example:  Two machines, which sometimes break down.
# Up time is exponentially distributed with mean 1.0, and repair time is
# exponentially distributed with mean 0.5.  There are two repairpersons,
# so the two machines can be repaired simultaneously if they are down
# at the same time.

# Output is long-run proportion of up time.  Should get value of about
# 0.66.

# SimPy constructs used:

#   Process class:  for setting up processes
#   yield hold:  yield is a Python generators construct, applied here by
#                SimPy to simulate the passage of time
#   initialize():  initializes the SimPy system
#   activate():  gets a process started
#   simulate():  gets the simulation started, and runs until the
#                specified time limit

# Overview of operation:

# Main program (see "'main' program starts here" near end of this file)
# sets up two objects of class Machine and calls activate() to start
# them.  Then main program calls simulate() to run the simulation, and
# then prints out the results.

# Objects of the Machine class represent machines.  The function
# Machine.Run() simulates the action of a machine.  It does this by
# calling "yield hold" to simulate an up time, then calling "yield hold"
# to simulate a repair time, repeating that cycle indefinitely.  It also
# of course does bookkeeping, e.g. to add to the total up time.

# required imports
from __future__ import generators  # delete if use Python >= 2.3
from SimPy.Simulation import *
from random import Random,expovariate,uniform

class Machine(Process):
   TotalUpTime = 0.0  # total up time for all machines
   NextID = 0  # next available ID number for Machine objects
   def __init__(self):
      Process.__init__(self)  # required in any Process subclass
      self.UpTime = 0.0  # amount of work this machine has done
      self.StartUpTime = 0.0  # time the current up period started
      self.ID = Machine.NextID   # ID for this Machine object
      Machine.NextID += 1
   def Run(self):
      print "starting machine", self.ID
      while 1:
         # record current time, now(), so can see how long machine is up
         self.StartUpTime = now()
         # hold for exponentially distributed up time
         yield hold,self,Rnd.expovariate(UpRate)
         # update up time total
         Machine.TotalUpTime += now() - self.StartUpTime
         # hold for exponentially distributed repair time
         yield hold,self,Rnd.expovariate(RepairRate)

# "main" program starts here

# set rates for this model
UpRate = 1/1.0
RepairRate = 1/0.5

# create an object of type Random, so can generate random numbers
Rnd = Random(12345)

initialize()  # required SimPy statement

# set up the two machine processes
for I in range(2):
   # create a Machine object
   M = Machine()
   # get the process M started, executing M's Run() method,
   # right away (i.e. no delay)
   activate(M,M.Run(),delay=0.0)

# run until simulated time 10000
MaxSimtime = 10000.0
simulate(until=MaxSimtime)

# print results
print "the percentage of up time was", Machine.TotalUpTime/(2*MaxSimtime)

```