```
import simpy
import random
import numpy as np

# note:output below
check_ls = []
scan_ls = []
waitingTime = []

for a in range(25, 36):
    for b in range(25, 36):
        print("Checkers: %i" % a)
        print("Scanners: %i" % b)
        check_ls.append(a)
        scan_ls.append(b)

        # resources
        numCheckers = a
        numScanners = b

        # rates
        arrRate = 50
        checkRate = 0.75
        scanRate = random.uniform(0.5, 1)
        runTime = 60*3 # 3 hours simulation
        replications = 50

        avgCheckTime = []
        avgScanTime = []
        avgWaitTime = []
        avgSystemTime = []


        # system simulation
        class System(object):
            def __init__(self,env):
                self.env = env
                self.checker = simpy.Resource(env,numCheckers) # 1 queue for
all checkers
                self.scanner = []
                for i in range(numScanners):
                    # individual queue for each scanner
                    self.scanner.append(simpy.Resource(env,1))

            # check time
            def check(self,passenger):
                yield self.env.timeout(random.expovariate(1.0/checkRate))

            # scan time
            def scan(self,passenger):
                yield self.env.timeout(scanRate)


        # passenger simulation
```

```
def passenger(env,name,s):
    global checkWait
    global scanWait
    global sysTime
    global totThrough

    # arrival time, now when function called
    timeArrive = env.now

    # check queue calling system class
    with s.checker.request() as request:
        yield request
        tIn = env.now # check start
        yield env.process(s.check(name))
        tOut = env.now # check end
        checkTime.append(tOut - tIn) # checking time

    # finding shortest scanner queue
    minq = 0
    for i in range(1,numScanners):
        # compare scanner queue number to get lowest
        if (len(s.scanner[i].queue) < len(s.scanner[minq].queue)):
            minq = i

    # scan queue calling system class
    with s.scanner[minq].request() as request:
          yield request
          tIn = env.now # scan start
          yield env.process(s.scan(name))
          tOut = env.now # scan end
          scanTime.append(tOut - tIn) # scanning time

    # end of passenger cycle
    timeLeave = env.now
    sysTime.append(timeLeave - timeArrive) # total time in system
    totThrough += 1 # number of passengers passed through


# setting up simulation, passenger going into system class
def setup(env):
    i = 0
    s = System(env)
    while True:
        yield env.timeout(random.expovariate(arrRate))
        i += 1 # passenger counter

        # calling passenger function with system class s
        env.process(passenger(env,'Passenger %d' % i,s))


# running simulation model [replication] number of times
for i in range(replications):
    random.seed(i)
```

```
            # instantiate environment
            env = simpy.Environment()

            totThrough = 0
            checkTime = []
            scanTime = []
            sysTime = []

            env.process(setup(env))
            env.run(until=runTime)

            # average times for one replication
            avgSystemTime.append(sum(sysTime[1:totThrough]) / totThrough)
            avgCheckTime.append(sum(checkTime[1:totThrough]) / totThrough)
            avgScanTime.append(sum(scanTime[1:totThrough]) / totThrough)
            avgWaitTime.append(avgSystemTime[i] - avgCheckTime[i] -
avgScanTime[i])

        # overall averages across all replications
        print('Average system time = %.2f' % (sum(avgSystemTime)/
replications))
        print('Average check time = %.2f' % (sum(avgCheckTime)/replications))
        print('Average scan time = %.2f' % (sum(avgScanTime)/replications))
        print('Average wait time = %.2f' % (sum(avgWaitTime)/replications))

        # record waiting time
        waitingTime.append((sum(avgWaitTime)/replications))

np.column_stack((check_ls, scan_ls, waitingTime))

# 33 checkers & 31 scanners should the optimal resources required to keep the
# average waiting time <15 minutes.
#
#      Checkers     Scanners       Waiting Time
# array([[25.        , 25.        , 29.49206945],
#        [25.        , 26.        , 40.39752291],
#        [25.        , 27.        , 38.57644881],
#        [25.        , 28.        , 36.76854219],
#        [25.        , 29.        , 34.98098431],
#        [25.        , 30.        , 33.22284384],
#        [25.        , 31.        , 31.57282498],
#        [25.        , 32.        , 30.38466079],
#        [25.        , 33.        , 29.93095365],
#        [25.        , 34.        , 29.79467801],
#        [25.        , 35.        , 29.73866407],
#        [26.        , 25.        , 42.26344665],
#        [26.        , 26.        , 36.11473412],
#        [26.        , 27.        , 34.13906763],
#        [26.        , 28.        , 32.18180298],
#        [26.        , 29.        , 30.27999825],
#        [26.        , 30.        , 28.68890819],
#        [26.        , 31.        , 27.81608241],
#        [26.        , 32.        , 27.59439218],
#        [26.        , 33.        , 27.52133246],
```

```
#        [26.        , 34.        , 27.49356992],
#        [26.        , 35.        , 27.48101109],
#        [27.        , 25.        , 38.10891765],
#        [27.        , 26.        , 25.10694229],
#        [27.        , 27.        , 25.10344603],
#        [27.        , 28.        , 25.10225819],
#        [27.        , 29.        , 25.10175958],
#        [27.        , 30.        , 25.10156754],
#        [27.        , 31.        , 25.10148691],
#        [27.        , 32.        , 25.10145045],
#        [27.        , 33.        , 25.10144303],
#        [27.        , 34.        , 25.10144088],
#        [27.        , 35.        , 25.10144088],
#        [28.        , 25.        , 22.92285889],
#        [28.        , 26.        , 22.84956183],
#        [28.        , 27.        , 22.84919532],
#        [28.        , 28.        , 22.84909393],
#        [28.        , 29.        , 22.84906523],
#        [28.        , 30.        , 22.84906102],
#        [28.        , 31.        , 22.8490591 ],
#        [28.        , 32.        , 22.8490591 ],
#        [28.        , 33.        , 22.8490591 ],
#        [28.        , 34.        , 22.8490591 ],
#        [28.        , 35.        , 22.8490591 ],
#        [29.        , 25.        , 20.39154218],
#        [29.        , 26.        , 34.8686601 ],
#        [29.        , 27.        , 32.82844877],
#        [29.        , 28.        , 30.79192666],
#        [29.        , 29.        , 28.76437839],
#        [29.        , 30.        , 26.75287408],
#        [29.        , 31.        , 24.7673026 ],
#        [29.        , 32.        , 22.85845798],
#        [29.        , 33.        , 21.43395386],
#        [29.        , 34.        , 20.87767048],
#        [29.        , 35.        , 20.71487478],
#        [30.        , 25.        , 36.93933423],
#        [30.        , 26.        , 27.00651094],
#        [30.        , 27.        , 24.69175866],
#        [30.        , 28.        , 22.41644264],
#        [30.        , 29.        , 20.29986265],
#        [30.        , 30.        , 18.88411047],
#        [30.        , 31.        , 18.47917968],
#        [30.        , 32.        , 18.37202957],
#        [30.        , 33.        , 18.33482253],
#        [30.        , 34.        , 18.31952382],
#        [30.        , 35.        , 18.31177573],
#        [31.        , 25.        , 29.42455903],
#        [31.        , 26.        , 16.01386883],
#        [31.        , 27.        , 16.00568876],
#        [31.        , 28.        , 16.00214626],
#        [31.        , 29.        , 16.00050278],
#        [31.        , 30.        , 15.99988593],
#        [31.        , 31.        , 15.99956831],
#        [31.        , 32.        , 15.99948131],
```

```
#        [31.        , 33.        , 15.99943511],
#        [31.        , 34.        , 15.9994218 ],
#        [31.        , 35.        , 15.99941904],
#        [32.        , 25.        , 13.05696005],
#        [32.        , 26.        , 40.22603682],
#        [32.        , 27.        , 38.37730591],
#        [32.        , 28.        , 36.53235013],
#        [32.        , 29.        , 34.68960732],
#        [32.        , 30.        , 32.84774508],
#        [32.        , 31.        , 31.00987769],
#        [32.        , 32.        , 29.17529005],
#        [32.        , 33.        , 27.34627531],
#        [32.        , 34.        , 25.5215491 ],
#        [32.        , 35.        , 23.70138716],
#        [33.        , 25.        , 42.20816287],
#        [33.        , 26.        , 24.67177767],
#        [33.        , 27.        , 22.2413113 ],
#        [33.        , 28.        , 19.82024794],
#        [33.        , 29.        , 17.42190465],
#        [33.        , 30.        , 15.07857036],
#        [33.        , 31.        , 12.94210041],
#        [33.        , 32.        , 11.58538098],
#        [33.        , 33.        , 11.24286978],
#        [33.        , 34.        , 11.14220191],
#        [33.        , 35.        , 11.10665829],
#        [34.        , 25.        , 26.95658827],
#        [34.        , 26.        , 41.15734476],
#        [34.        , 27.        , 39.34465163],
#        [34.        , 28.        , 37.53302541],
#        [34.        , 29.        , 35.72593791],
#        [34.        , 30.        , 33.92078918],
#        [34.        , 31.        , 32.11891129],
#        [34.        , 32.        , 30.3206681 ],
#        [34.        , 33.        , 28.5257462 ],
#        [34.        , 34.        , 26.7331717 ],
#        [34.        , 35.        , 24.94272434],
#        [35.        , 25.        , 43.08294731],
#        [35.        , 26.        , 11.29901838],
#        [35.        , 27.        ,  8.5962348 ],
#        [35.        , 28.        ,  6.88195204],
#        [35.        , 29.        ,  6.50755947],
#        [35.        , 30.        ,  6.42181273],
#        [35.        , 31.        ,  6.39332316],
#        [35.        , 32.        ,  6.38180194],
#        [35.        , 33.        ,  6.37654215],
#        [35.        , 34.        ,  6.37387659],
#        [35.        , 35.        ,  6.37267205]])
```