**Programming Language: Python**

**Libraries/Framework: XLSXWRITER (Excel File)**

**import** xlsxwriter  
**import** random  
  
  
**class** Packet(object):  
 **def** \_\_init\_\_(self, ts):  
 self.time\_slot = ts  
 self.next = None  
  
  
**class** LinkedList(object):  
 head = None  
 tail = None  
 size = 0  
  
 **def** add\_to\_last(self, new\_packet):  
 **if** self.head **is** None:  
 self.head = new\_packet  
 self.tail = new\_packet  
 self.size += 1  
 **else**:  
 new\_packet.next = self.head  
 self.head = new\_packet  
 self.size += 1  
  
 **def** remove\_first(self):  
 **if** self.tail **is** None:  
 **return  
 elif** self.head == self.tail:  
 self.head = None  
 self.tail = None  
 self.size -= 1  
 **else**:  
 tmp\_node = self.head  
 **while** tmp\_node.next != self.tail:  
 tmp\_node = tmp\_node.next  
 tmp\_node.next = None  
 self.tail = tmp\_node  
 self.size -= 1  
  
 **def** peek(self):  
 **return** self.tail  
  
 **def** increase\_delay(self):  
 tmp\_node = self.head  
 **while** tmp\_node **is not** None:  
 tmp\_node.time\_slot += 1  
 tmp\_node = tmp\_node.next  
  
  
**class** QueueSystem(LinkedList):  
 **def** enqueue(self, new\_packet):  
 super(QueueSystem, self).add\_to\_last(new\_packet)  
  
 **def** dequeue(self):  
 super(QueueSystem, self).remove\_first()  
  
  
**class** PacketSimulation(object):  
 **def** \_\_init\_\_(self):  
 self.prob\_list = []  
 p = 0.02  
 **while** p <= 1.0:  
 self.prob\_list.append(p)  
 p += 0.02  
 self.throughput\_list = []  
 self.delay\_list = []  
  
 **def** simulate(self):  
 queueSystem = QueueSystem()  
 s = 0.8  
 p = 0.02  
 **while** p <= 1.0:  
 num\_service\_packets = 0  
 total\_delay = 0  
 **while** queueSystem.size > 0:  
 queueSystem.dequeue()  
 **for** time\_slot **in** range(0, 1000000, 1):  
 **if** queueSystem.size > 0:  
 **if** random.random() < s:  
 init\_time = queueSystem.peek().time\_slot  
 total\_delay += time\_slot - init\_time  
 queueSystem.dequeue()  
 *# queueSystem* **if** queueSystem.size < 5:  
 **if** random.random() < p:  
 queueSystem.enqueue(Packet(time\_slot))  
 num\_service\_packets += 1  
 avg\_throughput = num\_service\_packets / float(time\_slot)  
 avg\_delay = total\_delay / float(num\_service\_packets)  
 *# print(avg\_delay)* self.throughput\_list.append(avg\_throughput)  
 self.delay\_list.append(avg\_delay)  
 p += 0.02  
  
  
 **def** create\_excel(self):  
 workbook = xlsxwriter.Workbook(**'QueueSimulation.xlsx'**)  
 worksheet = workbook.add\_worksheet()  
 bold = workbook.add\_format({**'bold'**:1})  
  
 *# grapgh data* headings = [**'Probability'**, **'Avg Throughput'**, **'Avg Delay'**]  
 data = []  
 data.append(self.prob\_list)  
 data.append(self.throughput\_list)  
 data.append(self.delay\_list)  
  
 *# add data to excel* worksheet.write\_row(**'A1'**, headings, bold)  
 worksheet.write\_column(**'A2'**, data[0])  
 worksheet.write\_column(**'B2'**, data[1])  
 worksheet.write\_column(**'C2'**, data[2])  
  
 *# graph 1 (busy vs prob)* chart1 = workbook.add\_chart({**'type'**: **'line'**})  
 chart1.add\_series({  
 **'name'**: **'=Sheet1!$B$1'**,  
 **'categories'**: **'=Sheet1!$A$2:$A$51'**,  
 **'values'**: **'=Sheet1!$B$2:$B$51'** })  
  
 chart1.set\_title({**'name'**: **'Avg Throughput vs Probability'**})  
 chart1.set\_x\_axis({**'name'**: **'Probability'**})  
 chart1.set\_y\_axis({**'name'**: **'Avg Throughput'**})  
 chart1.set\_style(10)  
  
 *# graph 2 (busy vs prob)* chart2 = workbook.add\_chart({**'type'**: **'line'**})  
 chart2.add\_series({  
 **'name'**: **'=Sheet1!$C$1'**,  
 **'categories'**: **'=Sheet1!$A$2:$A$51'**,  
 **'values'**: **'=Sheet1!$C$2:$C$51'** })  
  
 chart2.set\_title({**'name'**: **'Avg Delay vs Probability'**})  
 chart2.set\_x\_axis({**'name'**: **'Probability'**})  
 chart2.set\_y\_axis({**'name'**: **'Avg Delay'**})  
 chart2.set\_style(10)  
  
 worksheet.insert\_chart(**'D2'**, chart1, {**'x\_offset'**: 30, **'y\_offset'**: 100})  
 worksheet.insert\_chart(**'D2'**, chart2, {**'x\_offset'**: 30, **'y\_offset'**: 500})  
 workbook.close()  
  
p = PacketSimulation()  
p.simulate()  
p.create\_excel()  
**print** (**"done"**)