Chapter 1:

1- There are multiple simulation languages and software in the market, provide a list of three languages and three software packages and explain the pros and cons.

Three languages:

	pros	cons
Python	It is a good language for people who don't have programming skills before, easy to learn	Speed is slow and memory consumption is huge
R	R has a good amount of statistical analysis packages	It's hard to learn and it's complicated for doing algorithm
Matlab	It provides very accurate outputs when running simulation results	Matlab needs to improve its help function. It would take a long time to debug.

Three software packages:

	pros	cons
math	Easy to understand and widespread	There is limitation for built-in function
keras	It's a powerful python package for evaluating deep learning model	Hard to learn for beginner
stats	It's Includes a bunch of functions for statistical calculations in R	Compared to Python, R is not so popular for data analysis

2- Use the "Managing Workshop Machinery" example in Page 17 and answer the following questions.

A. Using the Classifying simulation models section in Page 9, in what classification (static/dynamic, deterministic/stochastic, continuous/discrete) this problem falls into?
The problem falls into discrete models. Because the working time of some machines are overlapping.

B. Using the Approaching a simulation-based problem section in Page 10, explain the seven steps you would take to solve this problem using simulation.

1. Problem analysis
   The aim of building a model is to decide the minimum time necessary for all work to be completed.

2. Data collection
   There are two machines called A1 and A2. We collect the minutes of each machine working in A1 and A2, separately.

3. Setting up the simulation model
   We should know the variables of interest, the methodology of model designed, and which part of the result we should focus on.

4. Simulation software selection
   Suggest choose python for the simulation software. Because Python has many packages to support running simulation models.

5. Verification of the software solution
   The most important thing is the model should be running successfully. In order to avoid debugging large-scale simulation code, suggest other people to peer-review to minimize potential errors.

6. Validation of the simulation model
   We should check the simulation model measurements produced are approximated by the real system's output measurements.

7. Simulation and analysis of results
   The outputs could immediately point out the measurements. The simulation model will give maximum insight and help with decision-making for users, which can save many time and costs.

3- Similar to Q3 above, use the “Predator-prey model” example in Page 20 and answer the following questions.

c) Using the Classifying simulation models section in Page 9, in what classification (static/dynamic, deterministic/stochastic, continuous/discrete) this problem falls into?

This problem belongs with deterministic and continuous simulation classification.

1. The default two parameters setting is determined by the initial statement.
2. The size of each animal’s population varies over time.

d) Using the Approaching a simulation-based problem section in Page 10, explain the seven steps you would take to solve this problem using simulation.

1. Problem analysis
   Starting from a known initial situation, we want to research the pattern of the size of both populations (prey and predators) over time.

2. Data collection
   Collect mortality rate, birth rate, the supplement chain of their food of prey and predators.

3. Setting the simulation model
   The prerequisite is to understand the probability distributions of the variables of interest. According to the textbook, we can use the equations to achieve our research goal:

$$\begin{cases} \frac{dx}{dt} = \alpha * x(t) - \beta * x(t) * y(t) \\ \frac{dy}{dt} = \gamma * y(t) - \delta * x(t) * y(t) \end{cases}$$

4. Simulation software selection
   Python is the perfect programming language for this question.

5. Verification of the software solution
   Double check the codes to see whether they are syntax errors

6. Validation of the simulation model
   Discuss the simulated output with peers, whether it makes sense or not.

7. Simulation and analysis of results
   After running the simulation model, the output shows the sizes of population changes over time.

Chapter 2:

4- Random Number Generators using Python codes

Code is in the other file.

A. In **Linear congruential generator** (in Page 34) explain the formula and elaborate the sensitivity of the random variable generator with respect to the parameters. In other words, do you have a better set of random variables for into higher a, c, and m? Set the parameters to (a =3, c= 4, m=5, and x=3) and generate the first 100 pseudorandom values in Python. Explain the role of each Python function you use.

   Ans: a, c, m must be non-negative integers. In order to be effectively using Linear congruential generators, we need to set m values as a large number. Because Xn+1 is determined by Xn, the sequence will get into a cycle as soon as the previous number in the sequence was repeated. Thus, we want the output to look like random numbers, so m should be a large number.

B. Explain the **Learmonth-Lewis generator** method and its parameters (similar to Problem 5 above). Set the parameters equal to a = 60, c = 0, and m = 2^(31) -1, and generate the first uniform random variables in the range of [0, 1] in Python. Explain the commands and the role of each function you use.

   Ans: we need to set uniform variables = x/m. (the parameters should be divided to the corresponding percentage between 0 to 1) In this case,The output of uniform distribution would be within the range of [0, 1].

C. Explain about **Lagged Fibonacci generato**r method and provide a list of benefits over other methods. Explain what m is in this method and how it helps to improve the generator. Implement a simple example of additive LFG in Python using the following parameters: x0 = x1 = 1 and m = 2^(32).

   Ans: m is the mod value, which sets the limit for the highest number the sequence can take. m is usually a power of 2 (m = 2^m). For example, a modval of 10 means the values in the sequence will be within the range of 10-1 = 9 that we can limit the calculation to a finite space.

5- Using Python function for random number generators do the following.

a) Generate 20 Pseudorandom numbers between 0 and 1.
0.7793 0.4987 0.4970 0.5484 0.5945 0.2897 0.3568 0.8099 0.6127 0.7301
0.7752 0.7317 0.2130 0.8085 0.8029 0.1062 0.5125 0.7403 0.4100 0.8685

b) Do part (a) using random.seed() function to generate the same list of random Variables.
0.6842 0.0958 0.1669 0.6051 0.3280 0.9146 0.7881 0.8647 0.4145 0.8288
0.5670 0.8306 0.3212 0.5482 0.4403 0.4122 0.3921 0.4143 0.1472 0.1424

c) Make a random number generator using a “for-loop” that generates integer random numbers 'n' number of times
For example, n = 10
0.31006641098111054
0.6936778047641966
0.660880825947305
0.7550396466876804
0.7252641506742268
0.8164943919155454
0.10326330437811937
0.3872833169208726
0.6218675131273177
0.06532881604247609

d) Use a random generator to select three “same” sets of random countries from the following list.
{Canada, USA, Italy, China, India, South Africa, Spain, Brazil, Mexico}
['Italy', 'USA', 'India'] ['Italy', 'USA', 'India'] ['Italy', 'USA', 'India']
Note: code is in the another file

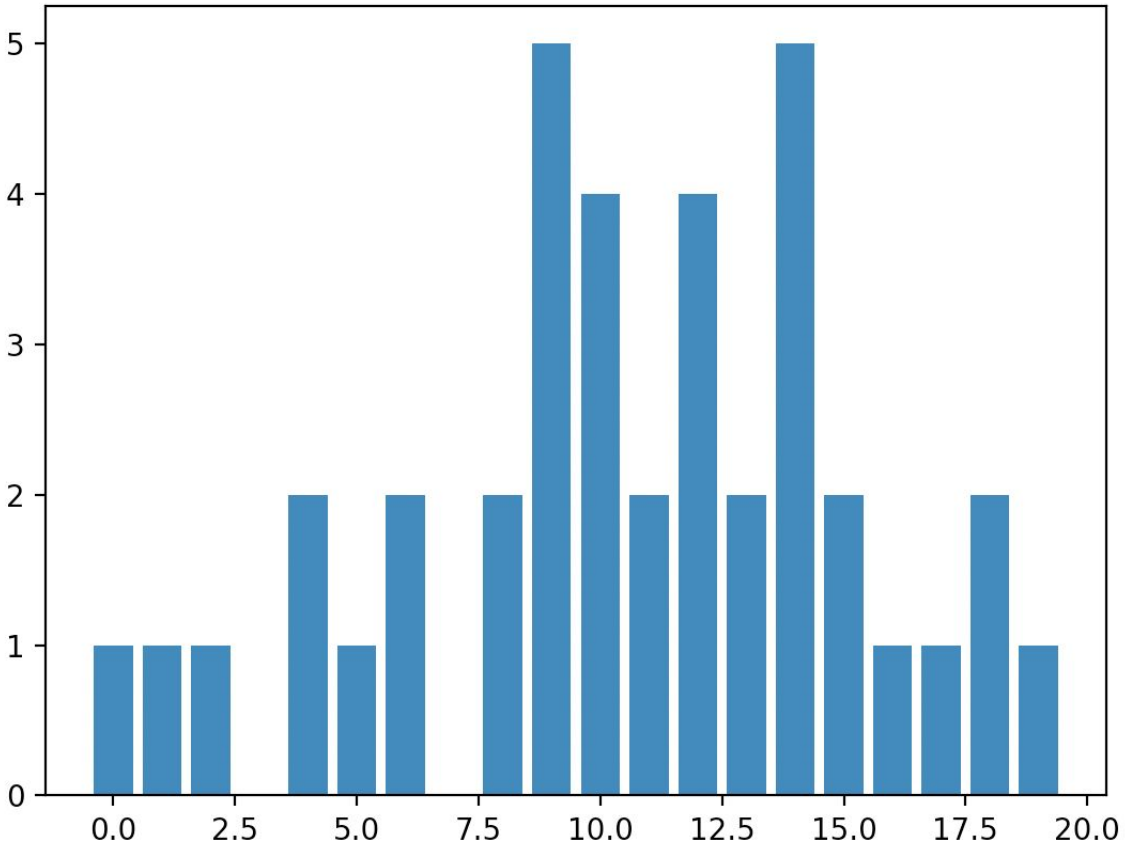
6- Testing Uniformity of Final Exam Grades. A random sample of final examination grades for a college course follows.

55 85 72 99 48 71 88 70 59 98 80 74 93 85 74 82 90 71 83 60 95 77 84 73 63 72 95 79 51 85 76 81 78 65 75 87 86 70 80 64

In the book we had test of Uniformity. Use the above dataset and methods in page 45-47 to determine if these values are uniformly distributed. Setup your Null and Alternative hypothesis carefully.

Ho: The sequence is uniformly distributed v.s. Ha: The sequence is not uniformly distributed.

The result is below:
R = [1 1 1 0 2 1 2 0 2 5 4 2 4 2 5 2 1 1 2 1]
V = 20.5
degrees of freedom = (2-1)*(20-1) = 19



According to the chi-squared distribution table, the v-value is 20.5, which is smaller than 30.14, meaning that we fail to reject the null hypothesis. Thus, the random sample of final examination grades is uniformly distributed with 95 % confidence interval.

Chapter 3:

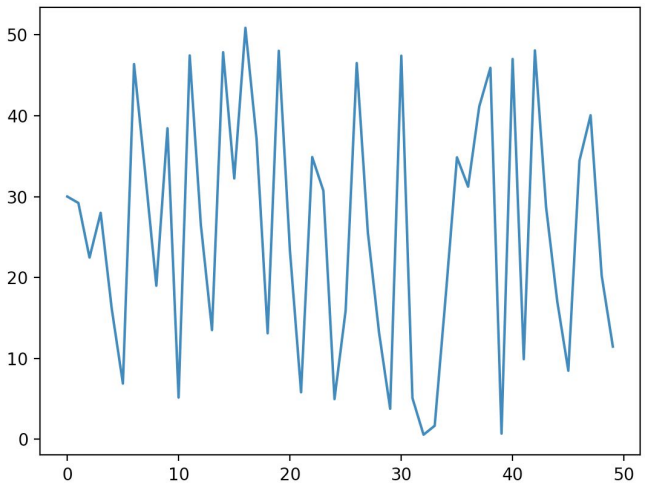
7- According to the above definition, solve the following two problems.

a) Toss two fair dice. Let A be the event that the sum of the two dice is 9. Let B be the event that the first die is a 4. Are A and B independent?
Ans: Yes, A and B are independent because knowing the outcome of A (3:6, 4:5, 5:4, 6:3) would not affect the outcome of B (4: ?).

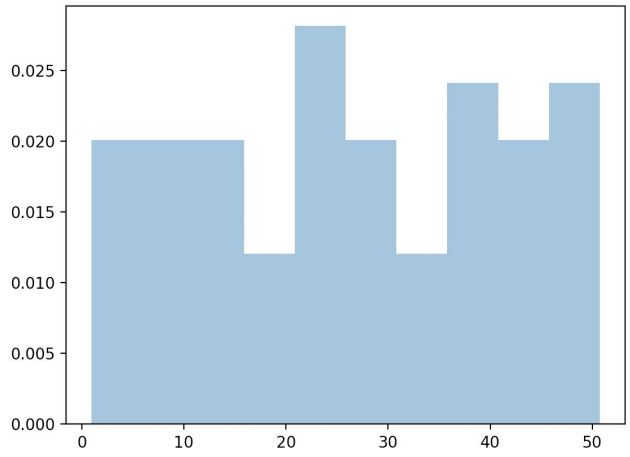
b) Toss two fair dice. Suppose A is the event that the sum of the two dice equals 7, and B is the event that the first die shows a 4. Are A and B independent?
Ans: Yes, A and B are independent because knowing the outcome of A (1:6, 2:5, 3:4, 4:3, 5:2, 6:1) would not affect the outcome of B (4: ?). In other words, as outcome of A gives no information about the outcome of B so A and B are still independent.

8- Using Python functions, generate a uniform distribution of random numbers contained in intervals [0, 50] and do the following.

• Draw a diagram in which we report the values of the 50 random numbers that we have generated.



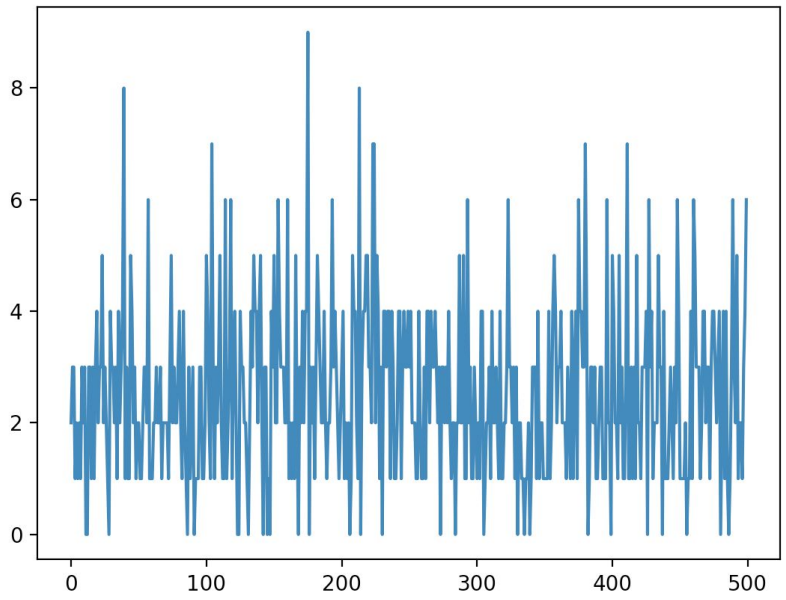
• Draw a graph of the probability density function.



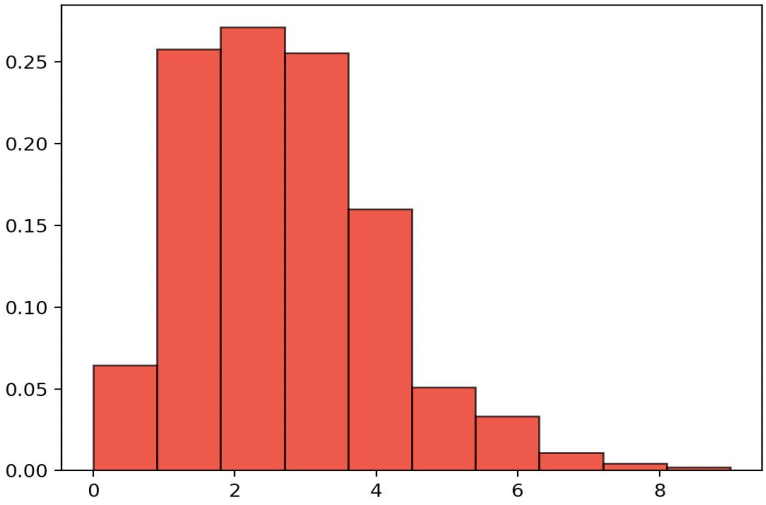
• Repeat the same graph by replicating 500 samples.
Please see the code and run it, the filename is called problem8.py

9- Using Python functions and libraries, evaluate the probability density function of a binomial distribution X as the probability of success with N = 500, n = 25, and p = 0.1.

• Generate the probability distribution
• Plot the data generated.



• Plot the histogram of the return values.

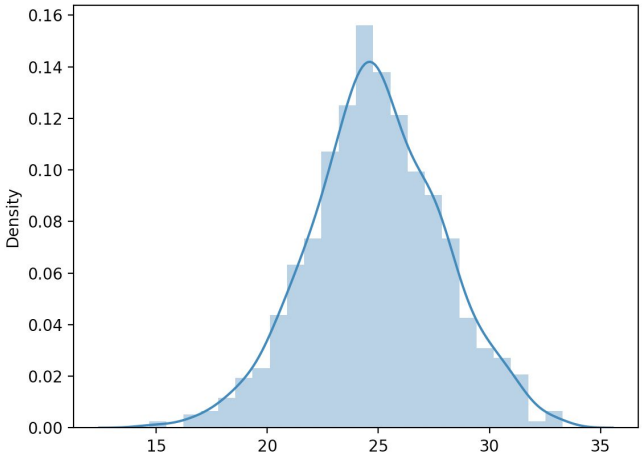


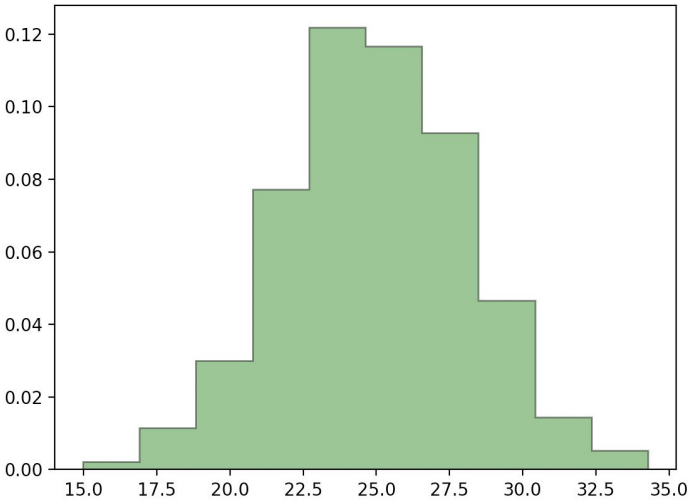
• Find probability of having 5 successes (that is p (X = 5))
Ans: 0.072 (because we extract random samples so everytime the probability is different).

10- Using Python functions and libraries, generate a normal probability distribution Y with mu= 25, sigma = 3.

• Generate the probability distribution
• Plot the data generated. Generate the same plot by changing Sigma = 4 and 5.
• Plot the histogram of the return values.
• FindP(Y>20)
• Find P (Y = 22.5)

generate a normal probability distribution Y with mu= 25, sigma = 3

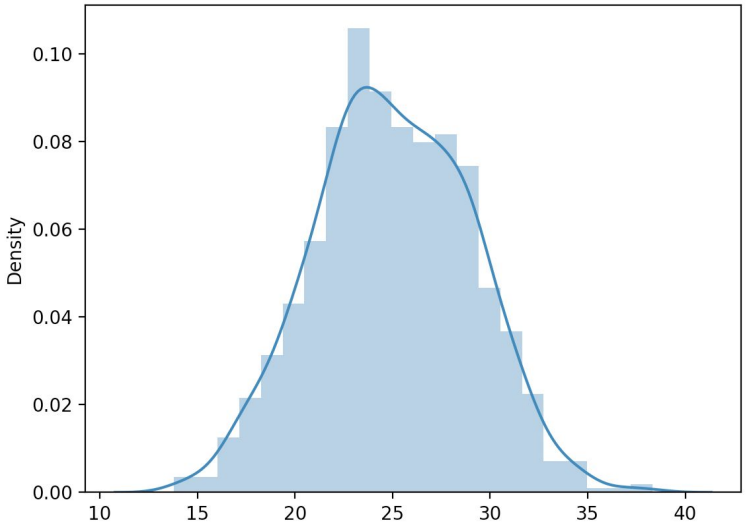


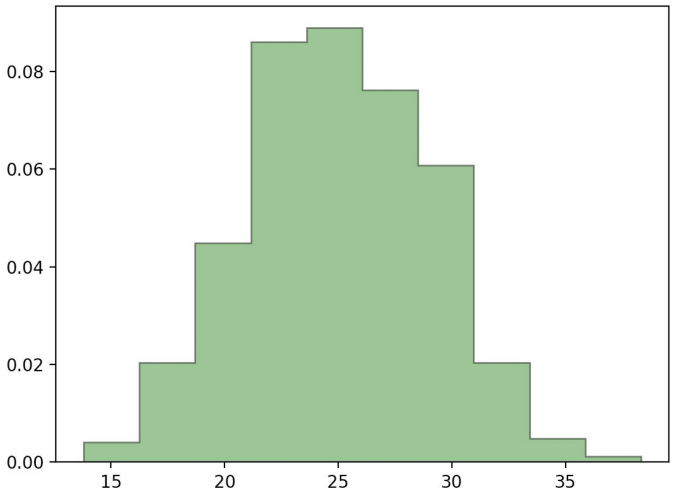


• FindP(Y>20): 0.944

• Find P (Y = 22.5): 0

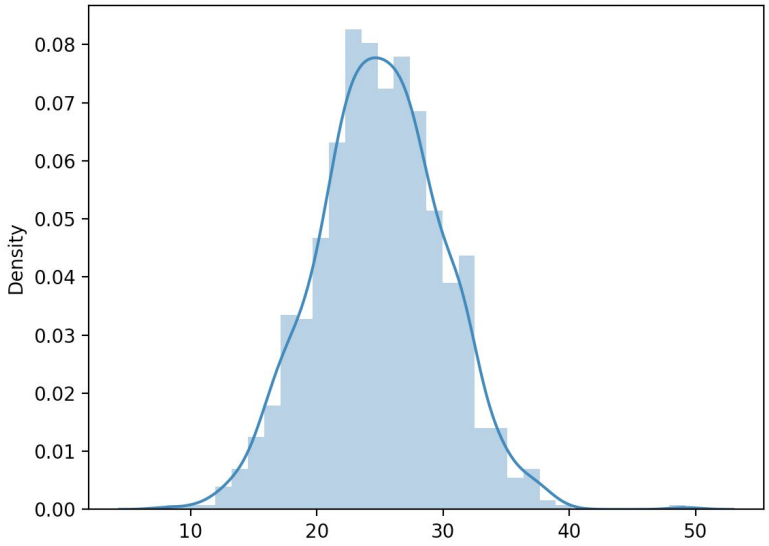
generate a normal probability distribution Y with mu= 25, sigma = 4

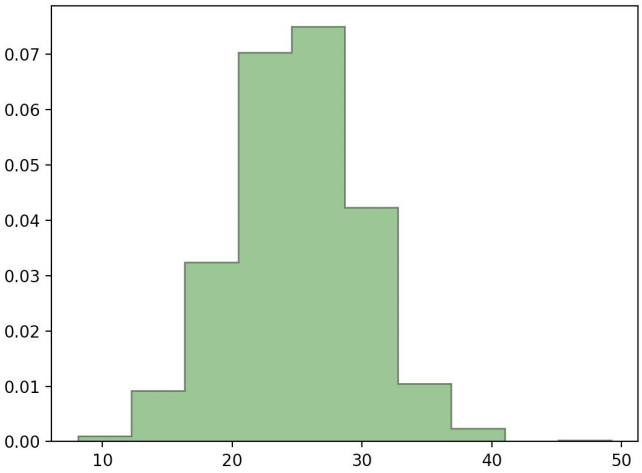




- FindP(Y>20): 0.896
- Find P (Y = 22.5): 0.896

generate a normal probability distribution Y with mu= 25, sigma = 5





- FindP(Y>20): 0.847

- Find P (Y = 22.5): 0