I though the due was tomorrow. I couldn’t finish.

1. Write an algorithm to use a fair six-sided die to generate coin flips.

import random

x=random.randint(1,6)

if (x<=3):

print ("It is Tail")

else:

print ("it is head")

1. The section “Getting Fairness from Biased Sources” explains how you can use a biased coin to get fair coin fl ips by fl ipping the coin twice. But sometimes doing two fl ips produces no result, so you need to repeat the process. Suppose the coin produces heads three-fourths of the time and tails one-fourth of the time. In that case, what is the probability that you’ll get no result after two fl ips and have to try again?

¾ \* ¾ = 9/16

¼ \* ¼ = 1/16

+\_\_\_\_\_\_\_\_\_\_\_

10/16

1. Again consider the coin described in Exercise 2. This time, suppose you were wrong, and the coin is actually fair but you’re still using the algorithm to get fair fl ips from a biased coin. In that case, what is the probability that you’ll get no result after two fl ips and have to try again?

If it is fair than no need to calculate there are %50 chance.

2. Write an algorithm to pick M random values from an array containing N items (where M ≤ N). What is its run time? How does this apply to the example described in the text where you want to give books to fi ve people selected from 100 entries? What if you got 10,000 entries? import random

for i in range(m):

x=random.randint(0,n)

newarray[i]=arr[x]

Run time is O(m)

3. What happens to Euclid’s algorithm if A < B initially?

It will not work because it is working with for example if a=100 and b=120 then 100%120 will be 0.

1. The least common multiple (LCM) of integers A and B is the smallest integer that A and B both divide into evenly. How can you use the GCD to calculate the LCM?

After finding GCD then if we found the smallest divisor of GDC it will be LDC. For example, GCD of 24 and 56 is 8. Smallest divisor of 8 is 2.