Name:

Please implement the following pseudo codes in your own programming language. Paste your code below each question.

1. Getting Fairness from Biased Sources:

Loop for 10 times

Flip the **biased coin** twice.

If the result is {Heads, Tails}, return Heads.

If the result is {Tails, Heads}, return Tails.

If the result is something else, start over.  
  
Hint: Biased coin can be implemented as follows in Python.

def BiasedCoin():

number = random.randint(1,4)

if number == 3:

return "tails"

else:

return "heads"

**import random**

**def BiasedCoin():**

**number = random.randint(1,4)**

**if number == 3:**

**return "tails"**

**else:**

**return "heads"**

**def fair():**

**toss1=BiasedCoin()**

**toss2=BiasedCoin()**

**print (toss1)**

**print (toss2)**

**if (toss1!=toss2):**

**if(toss1=="heads"):**

**print (toss1)**

**else:**

**print (toss1)**

**else:**

**fair()**

**fair()**

1. Randomize an Array:

RandomizeArray(String: array[])

Integer: max\_i = <Upper bound of array>

For i = 0 To max\_i - 1

// Pick the item for position i in the array.

Integer: j = <pseudorandom number between i and max\_i inclusive>

<Swap the values of array[i] and array[j]>

Next i

End RandomizeArray

**import random**

**def randomizeArray(arr):**

**max=len(arr)-1**

**for i in range(max-1):**

**j=random.randint(i,max)**

**a=arr[i]**

**arr[i]=arr[j]**

**arr[j]=a**

**a=["1", "2", "3", "4", "5", "6"]**

**randomizeArray(a)**

1. Calculate A to the Power P:

// Calculate A to the power P.

Float: RaiseToPower(Float: A, Integer: P)

<Use the first fact to quickly calculate A, A2, A4, A8, and so on until you get to a value AN where N + 1 > P>

<Use those powers of A and the second fact to calculate AP>

Return AP

End RaiseToPower

1. Finding Prime factors:

List Of Integer: FindFactors(Integer: number)

List Of Integer: factors

// Pull out factors of 2.

While (number Mod 2 == 0)

factors.Add(2)

number = number / 2

End While

// Look for odd factors.

Integer: i = 3

Integer: max\_factor = Sqrt(number)

While (i <= max\_factor)

// Pull out factors of i.

While (number Mod i == 0)

// i is a factor. Add it to the list.

factors.Add(i)

// Divide the number by i.

number = number / i

// Set a new upper bound.

max\_factor = Sqrt(number)

End While

// Check the next possible odd factor.

i = i + 2

End While

// If there's anything left of the number, it is a factor, too.

If (number > 1) Then factors.Add(number)

Return factors

End FindFactors

**def findFactors(n):**

**factors=[]**

**i=2**

**while(i<n):**

**while (n%i==0):**

**factors.append(i)**

**n=n/i**

**i+=1**

**if (n>1):**

**factors.append(int(n))**

**return factors**

**print (findFactors(30))**

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?

GCD Pseudocode:

Integer: GCD(Integer: A, Integer: B)

While (B != 0)

Integer: remainder = A Mod B

// GCD(A, B) = GCD(B, remainder)

A = B

B = remainder

End While

Return A

End GCD

**def GCD(a, b):**

**while(b!=0):**

**remainder=a%b**

**a=b**

**b=remainder**

**return a**

**x= GCD(32, 20)**

**print (x)**

Please paste LCD implementation here:

**def GCD(a, b):**

**while(b!=0):**

**remainder=a%b**

**a=b**

**b=remainder**

**return a**

**def LCD(a,b):**

**return a\*b/GCD(a,b)**

**x= LCD(32, 20)**

**print (x)**