Question 2

```
import numpy as np
import pandas as pd
from numpy import log10
import math
```

```
# Importing the dataset
df = pd.read_excel("dataset_metrics.xlsx")
```

```
#Defining evaluated and simulated value
e = list(df.actual_y)
s = list(df.predicted_y)
```

Defining Mean, Variance and Standard Deviation functions below, which will be called in question parts below.

```
#Calculating mean
def mean(arr):
  Sum=0
  for i in arr:
    Sum+=i
  mean=Sum/len(arr)
  return mean
```

```
#Defining variance function
def var(arr):
  mu=mean(arr)
  Sum=0
  for i in arr:
    Sum=Sum+((i-mu)**2)
  var=Sum/(len(arr)-1)
  return var
```

```
#Defining standard deviation function
def std(arr):
  var1=var(arr)
  std=math.sqrt(var1)
  return std
```

Part 1: Bias

```
#Defining Bias function
```

```
def bias(arr1,arr2):
  sum = 0
  for i in range(len(arr1)):
      sum += arr1[i] - arr2[i]
  bias = sum/len(arr1)
  return bias


#Finding Bias
print('Bias is : ',bias(e,s))
```

```
     Bias is :  -0.5
```

Part 2: pBias

```
#Defining pBias function
def pbias(arr1,arr2):
  sum1 = 0
  sum2 = 0
  for i in range(len(arr1)):
      sum1 += arr1[i] - arr2[i]
      sum2 += arr1[i]
  pbias = 100*sum1/sum2
  return pbias


#Finding pBias
print('p-bias is : ',pbias(e,s))
```

```
     p-bias is :  -3.125
```

Part 3: Nash-Sutcliffe (NSE)

```
#Defining NSE function
def nse(arr1,arr2):
  sum1=0
  sum2=0

  for i in range(len(arr1)):
    sum1+=(arr1[i]-arr2[i])**2

  emean=mean(arr1)

  for i in range(len(arr1)):
    sum2+=(arr1[i]-emean)**2

  nse = 1-(sum1/sum2)
  return nse


#Finding NSE
print('NSE is : ',nse(e,s))
```

```
     NSE is :  0.8417085427135679
```

Part 4: log Nash-Sutcli□e

```
#Defining logNSE function
def lognse(arr1,arr2):
  sum1=0
  sum2=0

  for i in range(len(arr1)):
    sum1+=(log10(arr1[i])-log10(arr2[i]))**2

  emean=mean(arr1)

  for i in range(len(arr1)):
    sum2+=(log10(arr1[i])-log10(emean))**2

  lognse = 1-(sum1/sum2)
  return lognse


#Finding logNSE
print('logNSE is : ',lognse(e,s))
```

```
     logNSE is :  0.7121054364600314
```

Part 5: Correlation Coefficient

```
#Defining correlation function
def corr(arr1,arr2):
  sum1=0
  sum2=0
  sum3=0
  mu1=mean(arr1)
  mu2=mean(arr2)
  for i, j in zip(arr1, arr2):
    sum1+=((i-mu1)*(j-mu2))
  for i in arr1:
    sum2+=((i-mu1)**2)
  for j in arr2:
    sum3+=((j-mu2)**2)
  corr=sum1/(math.sqrt(sum2*sum3))
  return corr


#Finding Correlation Coefficient
print('Correlation Coefficient is : ',corr(e,s))
```

```
     Correlation Coefficient is :  0.92088934100307
```

Part 6: rsquared

```
#Defining rsquared
def rsquared(arr1,arr2):
  sum1=0
  sum2=0
  sum3=0
  mu1=mean(arr1)
  mu2=mean(arr2)
  for i, j in zip(arr1, arr2):
    sum1+=((i-mu1)*(j-mu2))
  for i in arr1:
    sum2+=((i-mu1)**2)
  for j in arr2:
    sum3+=((j-mu2)**2)
  corr=sum1/(math.sqrt(sum2*sum3))
  rsquared=corr**2
  return rsquared



#Finding rsquared
print('rsquared is : ',rsquared(e,s))
```

```
     rsquared is :  0.8480371783730685
```

Part 7: MSE

```
#Defining Mean of Squared Error Function mse
def mse(arr1,arr2):
  arr3=[]
  for i, j in zip(arr1, arr2):
    arr3.append(i-j)
  sum=0
  for i in arr3:
    sqr=i**2
    sum+=sqr
  mse=sum/(len(arr2))
  return mse


#Finding Mean of Squared Error mse
print('Mean of Squared Error mse is : ',mse(e,s))
```

```
     Mean of Squared Error mse is :  6.3
```

Part 8: Root MSE

```
#Defining Root Mean of Squared Error Function rmse
def rmse(arr1,arr2):
  arr3=[]
  for i, j in zip(arr1, arr2):
    arr3.append(i-j)
  sum=0
  for i in arr3:
```

```
  for i in arr3:
    sqr=i**2
    sum+=sqr
  mse=sum/(len(arr2))
  rmse=mse**0.5
  return rmse


#Finding Root Mean of Squared Error rmse
print('Root Mean of Squared Error rmse is : ',rmse(e,s))
```

```
Root Mean of Squared Error rmse is :  2.5099800796022267
```

Part 9: Mean Absolute Error

```
#Defining Mean Absolute Error mae
def mae(arr1,arr2):
  sum=0
  arr3=[]
  for i, j in zip(arr1, arr2):
    arr3.append(i-j)
  sum=0
  for i in arr3:
    sqr=abs(i)
    sum+=sqr
  mae=sum/(len(arr2))
  return mae


#Finding Mean Absolute Error mae
print('Mean Absolute Error mae is : ',mae(e,s))
```

```
Mean Absolute Error mae is :  1.9
```

Part 10: RRMSE

```
#Defining RRMSE
def rrmse(arr1,arr2):
  arr3=[]
  for i, j in zip(arr1, arr2):
    arr3.append(i-j)
  sum=0
  for i in arr3:
    sqr=i**2
    sum+=sqr
  mse=sum/(len(arr2))
  rmse=mse**0.5
  rrmse=rmse/mean(e)
  return rrmse


#Finding RRMSE
print('RRMSE is : ',rrmse(e,s))
```

```
     RRMSE is :  0.15687375497513917
```

Part 11: Agreementindex

```
#Defining Agreementindex
def aggrementindex(arr1,arr2):
  sum1 = 0
  sum2 = 0
  for i in range(len(arr1)):
    sum1 += (arr1[i] - arr2[i])**2

  for j in range(len(arr1)):
    sum2 += (abs(arr2[j] - mean(arr1)) + abs(arr1[j] - mean(arr1)))**2

  d = 1 - (sum1/sum2)
  return d


#Finding Agreementindex
print('Agreementindex is : ',aggrementindex(e,s))
```

```
     Agreementindex is :  0.955096222380613
```

Part 12: Covariance

```
#Defining Covariance
def covariance(arr1,arr2):
  sum=0
  emean=mean(arr1)
  smean=mean(arr2)
  for i, j in zip(arr1, arr2):
    sum += ((i-emean)*(j-smean))
  covariance= sum/len(arr1)
  return covariance


#Finding Covariance
print('Covariance is : ',covariance(e,s))
```

```
     Covariance is :  33.5
```

Part 13: Kling-Gupta Efficiency (kge)

```
#Defining Kling-Gupta Efficiency (kge)
def kge(arr1,arr2):
  emean=mean(arr1)
  smean=mean(arr2)

  estd=std(arr1)
  sstd=std(arr2)
```

```
  cc=corr(e,s)

  kge=1-((cc-1)**2+(emean/smean)**2+(estd/sstd)**2)**0.5

  return kge


#Finding Kling-Gupta Efficiency (kge)
print('Kling-Gupta Efficiency (kge) is : ',kge(e,s))
```

```
     Kling-Gupta Efficiency (kge) is :  -0.46409125077961644
```

Part 14: RSR

```
#Defining RSR
def rsr(arr1,arr2):
  arr3=[]
  for i, j in zip(arr1, arr2):
    arr3.append(i-j)
  sum=0
  for i in arr3:
    sqr=i**2
    sum+=sqr
  mse=sum/(len(arr2))
  rmse=mse**0.5

  estd=std(arr1)
  rsr=rmse/estd

  return rsr


#Finding RSR
print('RSR is : ',rsr(e,s))
```

```
     RSR is :  0.37744179890122
```

Part 15: Volume Error

```
#Defining Volume Error
def vol_error(arr1,arr2):
  sum1 = 0
  for i in range(len(arr1)):
    sum1 += (arr2[i] - arr1[i])
  sum2 = 0
  for j in arr1:
    sum2 += j
  vol_error = sum1/sum2
  return vol_error


#Finding Volume Error
print('Volume Error is : ',vol_error(e,s))
```

```
print('Volume Error is : ',vol_error(e,3))
```

```
Volume Error is :  0.03125
```