Missing Values

```
import numpy as np
import pandas as pd
data = { 'A': [1, 2,
np.nan, 4, 5],
'B': [np.nan, 6, 7, np.nan, 9],
'C': [10, 11, 12, np.nan, 14],
'D': [np.nan, np.nan, np.nan, np.nan, np.nan]
}
df = pd.DataFrame(data) df_dropped
= df.dropna() df_filled = df.fillna(0)
df_mean_filled = df.fillna(df.mean())
df_median_filled = df.fillna(df.median())
df_ffilled = df.ffill() df_bfilled = df.bfill()
print("Original DataFrame:")
print(df)
print("\nDataFrame after removing rows with missing values:")
print(df_dropped)
print("\nDataFrame after filling missing values with 0:")
print(df_filled)
print("\nDataFrame after filling missing values with column means:")
print(df_mean_filled)
print("\nDataFrame after filling missing values with column medians:")
print(df_median_filled)
print("\nDataFrame after forward filling missing values:") print(df_ffilled)
print("\nDataFrame after backward filling missing values:")
print(df_bfilled)
```

K Means Clustering

```
import numpy as np
import pandas as pd
import math
import matplotlib.pyplot as plt

np.random.seed(42)

def PointsInCircum(r, n=100):
   return [(math.cos(2 * math.pi / n * x) * r + np.random.normal(0, 30),
         math.sin(2 * math.pi / n * x) * r + np.random.normal(0, 30)) for x
in range(1, n + 1)]

df = pd.DataFrame(PointsInCircum(500, 1000))
df = pd.concat([df, pd.DataFrame(PointsInCircum(300, 700))],
ignore_index=True)
df = pd.concat([df, pd.DataFrame(PointsInCircum(100, 300))],
ignore_index=True)
df = pd.concat([df, pd.DataFrame([(np.random.randint(-600, 600),
np.random.randint(-600, 600)) for i in range(300)])], ignore_index=True)

print(df.head())
```

Combinatorics

```
def factorial(n):
        if(n==1 or n==0):
                return n
 else:
        return (n*factorial(n-1))
def permutations(n,r):
        return factorial(n)//factorial(n-r)
def combinations(n,r):
        return permutations(n,r)//factorial(r)
n=int(input("Enter the number of items :"))
r=int(input("Enter the number of items to be choosed :"))
print("The Combinations is : ",combinations(n,r))
```

Add matrix

```
A= []
print("Enter 9 Elements for First Matrix: ")
for i in range(3):
        A. append([])
        for j in range(3):
                num = int(input())
                A[i].append(num)
B = []
print("Enter 9 Elements for Second Matrix: ")
for i in range(3):
        B. append([])
        for j in range(3):
                num = int(input())
                B[i].append(num)
Result = []
for i in range(3):
        Result.append([])
        for j in range(3):
                Result[i].append(A[i][j]+B[i][j])
print("\nAddition Result of Two Given Matrix is:")
for i in range(3):
        for j in range(3):
                print(Result[i][j], end=" ")
 print()
```

Permutation of given number

```
def factorial(n):
        if(n==1 or n==0):
                return n
        else:
                return (n*factorial(n-1))
def permutations(n,r):
        return factorial(n)//factorial(n-r)
n=int(input("Enter the number of items :"))
r=int(input("Enter the number of items to be choosed :"))
print("The permutations ",permutations(n,r))
```

Factorial

```
def factorial(n):
        if n == 0:
                return 1
         else:
                return n * factorial(n - 1)
num = int(input("Enter a number: "))
print("Factorial:", factorial(num))
factorial(n)
```