Assignment 4

Name: Mahaprasad Mohanty

Roll No: 24MDT0061

In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
```

1.1 Generate the matrix A

In [2]:
```
A = np.array([[15+j + 10*i for i in range(6)] for j in range(6) if j !=2])
A
```

Out[2]:
```
array([[15, 25, 35, 45, 55, 65],
       [16, 26, 36, 46, 56, 66],
       [18, 28, 38, 48, 58, 68],
       [19, 29, 39, 49, 59, 69],
       [20, 30, 40, 50, 60, 70]])
```

1.2 Generate the matrix B and C

In [3]:
```
B = A[:4, :3]
B
```

Out[3]:
```
array([[15, 25, 35],
       [16, 26, 36],
       [18, 28, 38],
       [19, 29, 39]])
```

In [4]:
```
C = A[:4, :5]
C
```

Out[4]:
```
array([[15, 25, 35, 45, 55],
       [16, 26, 36, 46, 56],
       [18, 28, 38, 48, 58],
       [19, 29, 39, 49, 59]])
```

1.3 Generate D

In [5]:
```
D = A[[4, 0]][:, [0, 2, 4]]
D
```

Out[5]:
```
array([[20, 40, 60],
       [15, 35, 55]])
```

1.4 Generate E, F and G

In [6]:
```
E = np.linspace(10,60,9).reshape(3,3).astype(int)
E
```

Out[6]:
```
array([[10, 16, 22],
       [28, 35, 41],
       [47, 53, 60]])
```

In [7]:
```
F = np.linspace(10,60,15).reshape(5,3).astype(int)
F
```

Out[7]:
```
array([[10, 13, 17],
       [20, 24, 27],
       [31, 35, 38],
       [42, 45, 49],
       [52, 56, 60]])
```

In [8]:
```
G = np.arange(1,10,2)
G
```

Out[8]:
```
array([1, 3, 5, 7, 9])
```

1.5 Compute $H = BF^TC + G$

In [46]:
```
H = (B@F.T@C) + G
H
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[46], line 1
----> 1 H = (B@F.T@C) + G
      2 H

ValueError: matmul: Input operand 1 has a mismatch in its core dimension 0, with gufunc signature (n?,k),(k,m?)->(n?,
m?) (size 4 is different from 5)
```

This matrix multiplication is not possible since the dimensions of (B@F.T) and C do not match. Neither is broadcasting possible.

1.6 Rounding the entires of H

This is not possible to compute either since it depends on the previous problem.

1.7 Finding inverse, eigenvalues and eigenvectors

In [48]:
```
E_inv = np.linalg.inv(E)
print('Inverse of E is : \n', E_inv)
```

```
Inverse of E is :
 [[ 0.228125 -0.64375   0.35625 ]
 [-0.771875  1.35625  -0.64375 ]
 [ 0.503125 -0.69375   0.30625 ]]
```

In [14]:
```
eig_val_E = np.linalg.eig(E).eigenvalues
eig_vec_E = np.linalg.eig(E).eigenvectors

print(f'The eigenvalues of E are : {eig_val_E}')
print(f'The eigenvectors of E are \n: {eig_vec_E}')
```

```
The eigenvalues of E are : [110.45129675  -5.93911436   0.48781761]
The eigenvectors of E are
: [[-0.26117003 -0.76196737  0.37441534]
 [-0.53391558 -0.11680098 -0.81940245]
 [-0.80419175  0.63699549  0.43404237]]
```

1.8 To find out $(ED^T)^T F^T$

In [49]:
```
result = ((E@D.T).T)@F.T

#row-wise sum
row_wise = np.sum(result, axis = 1)

for i,j in enumerate(row_wise,1):
    print(f"Sum of row {i} is {j}")

# #column-wise sum
col_wise = np.sum(result, axis = 0)

for i,j in enumerate(col_wise,1):
    print(f"Sum of column {i} is {j}")
```

```
Sum of row 1 is 2371520
Sum of row 2 is 2091560
Sum of column 1 is 361800
Sum of column 2 is 619320
Sum of column 3 is 893440
Sum of column 4 is 1159240
Sum of column 5 is 1429280
```

1.9 Is E@F possible? by broadcasting?

In [12]:
```
print(f"Shape of E is: {E.shape}")
print(f"Shape of F is: {F.shape}")
```

```
Shape of E is: (3, 3)
Shape of F is: (5, 3)
```

In [50]:
```
E@F
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[50], line 1
----> 1 E@F

ValueError: matmul: Input operand 1 has a mismatch in its core dimension 0, with gufunc signature (n?,k),(k,m?)->(n?,
m?) (size 5 is different from 3)
```

No they are not broadcastable because the dimensions don't match and it is not possible to replicate any of the columns/ rows.

1.10 Generate a matrix of 4x3x5x3 by reshaping C,E,F

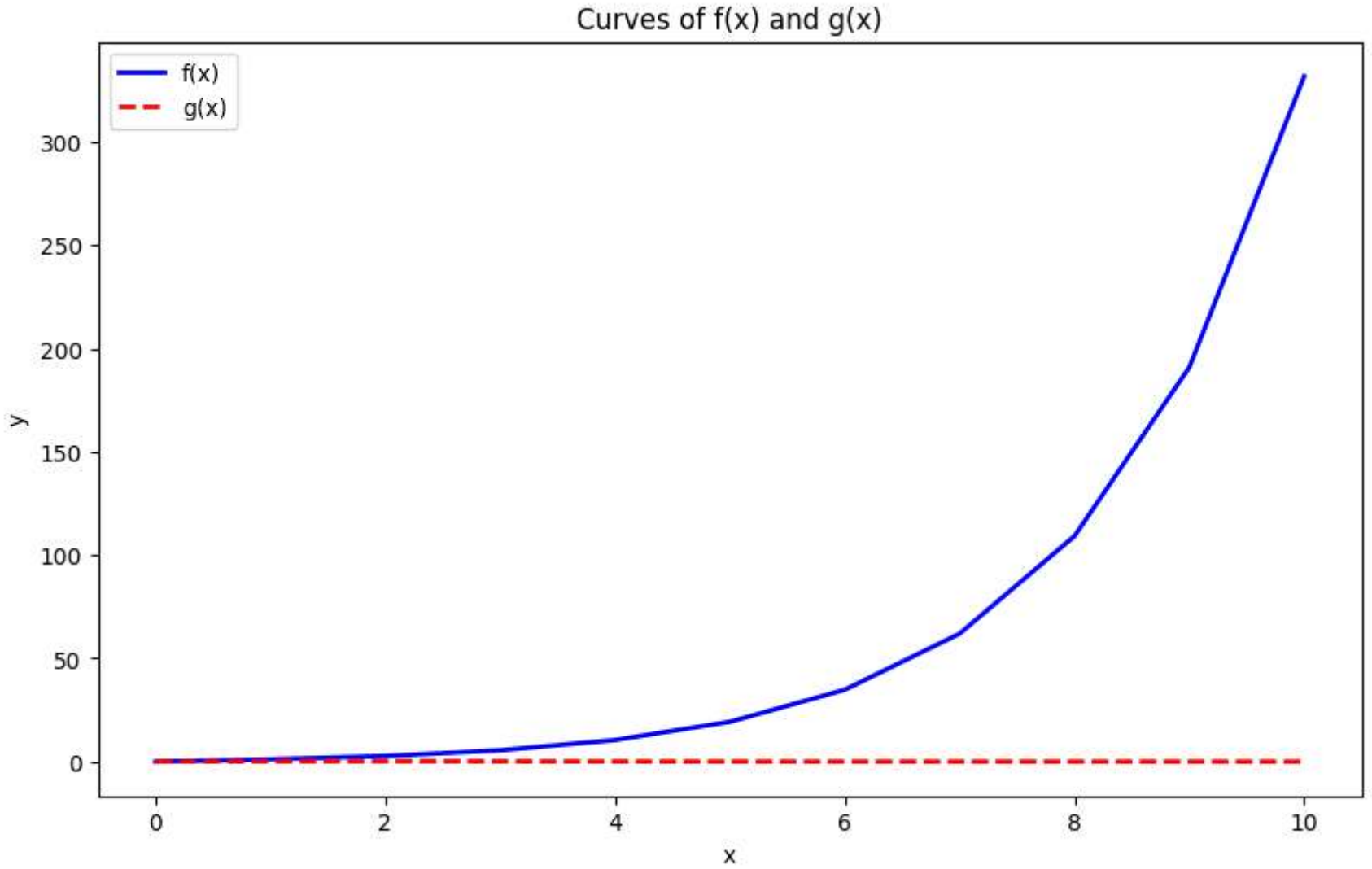
In [14]:
```
C_reshaped = C.reshape(4,1,5,1)
E_reshaped = E.reshape(1,3,1,3)
F_reshaped = F.reshape(1,1,5,3)

result = C_reshaped + E_reshaped + F_reshaped
print("Shape of the result: ", result.shape)
```

```
Shape of the result:  (4, 3, 5, 3)
```

Exercise 2

In [15]:
```
x = np.arange(11)
f = np.sqrt((np.exp(x) + np.sinh(x))*(x/3))
g = (0.5*x)/(2 + np.cosh(x))
```

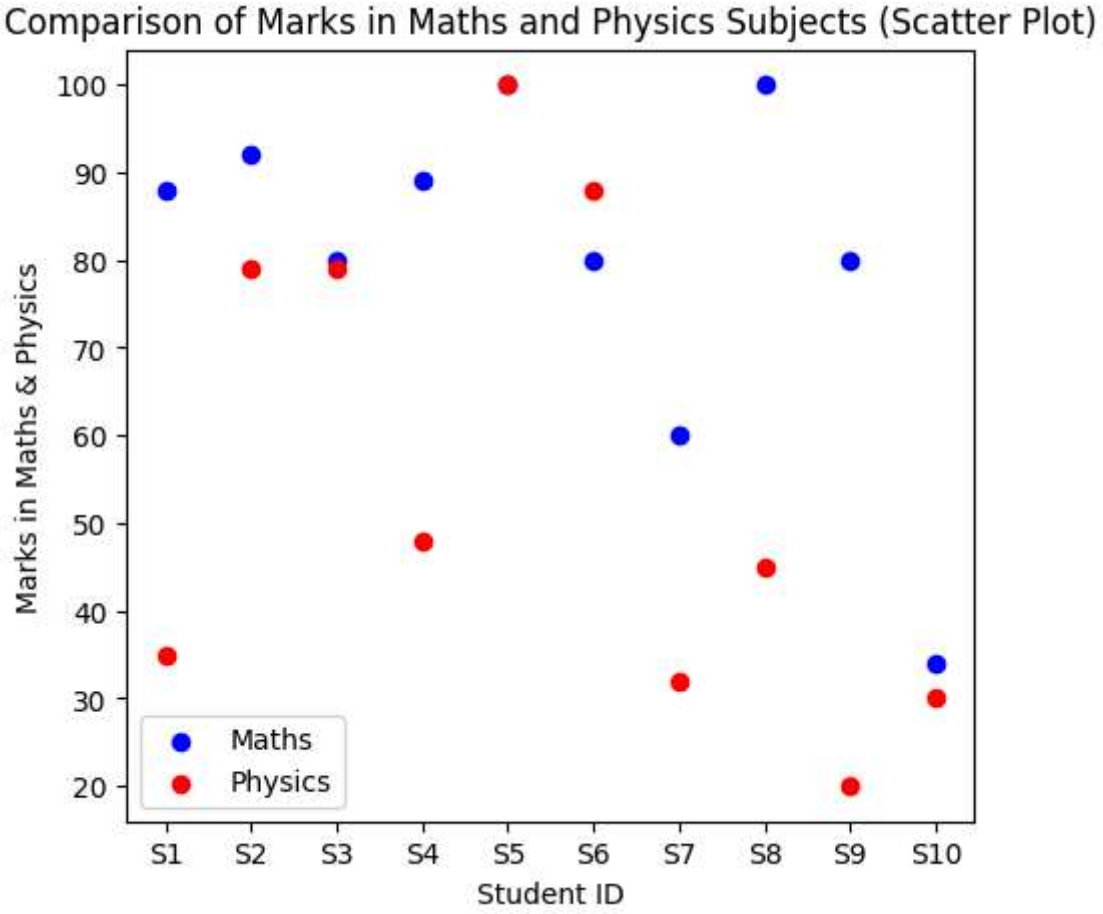
In [22]:
```
plt.figure(figsize=(10, 6))
plt.plot(x, f, color='blue', linestyle='solid', linewidth=2, label='f(x)')
plt.plot(x, g, color='red', linestyle='dashed', linewidth=2, label='g(x)')

plt.xlabel('x')
plt.ylabel('y')
plt.title('Curves of f(x) and g(x)')

plt.legend()
plt.show()
```



Exercise 3

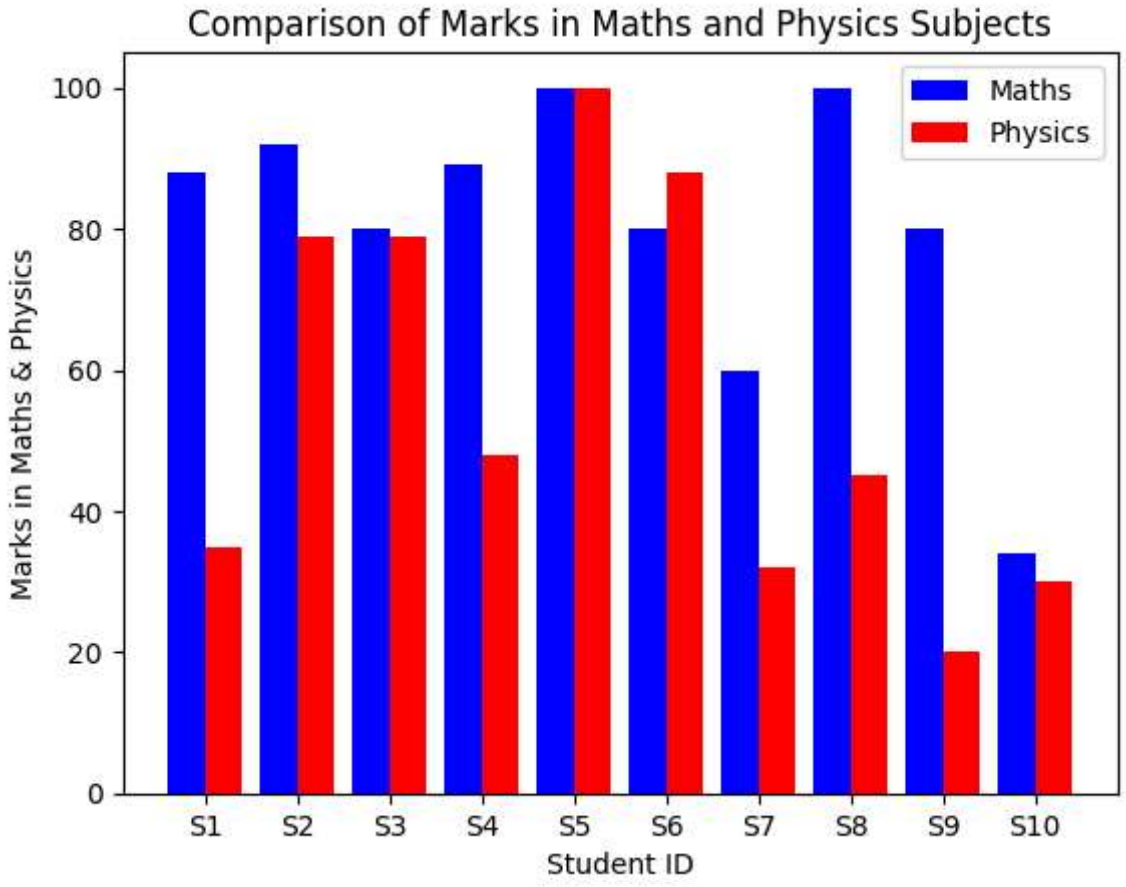
In [24]:
```
student_ids = ['S1', 'S2', 'S3', 'S4', 'S5', 'S6', 'S7', 'S8', 'S9', 'S10']
marks_maths = [88, 92, 80, 89, 100, 80, 60, 100, 80, 34]
marks_physics = [35, 79, 79, 48, 100, 88, 32, 45, 20, 30]
```

In [26]:
```
plt.figure(figsize=(12, 5))
plt.subplot(1, 2, 1)
plt.scatter(student_ids, marks_maths, label='Maths', color='blue')
plt.scatter(student_ids, marks_physics, label='Physics', color='red')
plt.xlabel('Student ID')
plt.ylabel('Marks in Maths & Physics')
plt.title('Comparison of Marks in Maths and Physics Subjects (Scatter Plot)')
plt.legend()
plt.show()
```



In [43]:
```
x = np.arange(len(student_ids))
plt.bar(x - 0.2, marks_maths,0.4,label='Maths', color='blue')
plt.bar(x + 0.2, marks_physics,0.4,label='Physics', color='red')
```

```
plt.xlabel('Student ID')
plt.ylabel('Marks in Maths & Physics')
plt.title('Comparison of Marks in Maths and Physics Subjects')
plt.xticks(x, student_ids)
plt.legend()
plt.show()
```



In [ ]: