

1. a) Arun is working in an office which is N blocks away from his house. He wants to minimize the time it takes him to go from his house to the office. He can either take the office cab or he can walk to the office. Arun's velocity is V_1 m/s when he is walking. The cab moves with velocity V_2 m/s but whenever he calls for the cab, it always starts from the office, covers N blocks, collects Arun and goes back to the office.

The cab crosses a total distance of N meters when going from office to Arun's house and vice versa, whereas Arun covers a distance of $(\sqrt{2})(2)*N$ while walking. Write a program to help Arun to find whether he should walk or take a cab to minimize the time.

CODE:

```
N = int(input('Enter the number of blocks :'))
V1 = int(input('Enter the Walking velocity :'))
V2 = int(input('Enter the cab velocity:'))

walk= 2*(2**0.5)*N/V1
cab = 2*N/V2
print('cab') if(walk>cab) else print('walk')
```

OUTPUT:

```
● PS D:\Code\kmit> & C:/Users/akuld/AppData/Local/Microsoft/WindowsApps/python3.9.exe d:/Code/kmit/python/syl/1a.py
Enter the number of blocks :2
Enter the Walking velocity :10
Enter the cab velocity:14
cab
```

- b) There is a robot which wants to go the charging point to charge itself. The robot moves in a 2-D plane from the original point (0,0). The robot can move toward UP, DOWN, LEFT and RIGHT with given steps. The trace of robot movement is shown as the following: UP 5 DOWN 3 LEFT 3 RIGHT 2 Then, the output of the program should be: 2 The numbers after the direction are steps. Write a program to compute the distance between the current position after a sequence of movement and original point. If the distance is a float, then just print the nearest integer (use round() function for that and then convert it into an integer).

CODE:

```
x = 0
y = 0
for i in range(4):
    direction, steps = input().split()
    if direction == 'UP':
        y += int(steps)
    elif direction == 'DOWN':
        y -= int(steps)
    elif direction == 'LEFT':
        x -= int(steps)
    elif direction == 'RIGHT':
        x += int(steps)

print(round((x**2 + y**2)**0.5))
```

OUTPUT:

```
● PS D:\Code\kmit> & C:/Users/akuld/AppData/Local/Microsoft/WindowsApps/python3.9.exe d:/Code/kmit/python/syl/1b.py
UP 5
DOWN 3
LEFT 3
RIGHT 2
2
```

2. a) Write a program that prints the numbers from 1 to 100. But for multiples of three print “Fizz” instead of the number and for the multiples of five print “Buzz”. For numbers which are multiples of both three and five print.

CODE:

```
#2a
n = int(input('enter number:'))
for i in range(1,n+1):
    if(i%3==0 and i%5==0):
        print('FizzBuzz')
    elif (i%3==0 ):
        print('fizz')
    elif(i%5==0 ):
        print('Buzz')
    else:
        print(i)
```

OUTPUT:

```
● PS D:\Code\kmit> & C:/Users/akuld/AppData/Local/Microsoft/WindowsApps/python3.9.exe d:/Code/kmit/python/syl/2a.py
enter number:15
1
2
fizz
4
Buzz
fizz
7
8
fizz
Buzz
11
fizz
13
14
FizzBuzz
```

- b) Write a program to check whether a given positive number is a Harshad Number or not. A Harshad number is an integer that is divisible by the sum of its digits. For example, 171 is a Harshad Number because the sum of digits is 9(1+7+1) and 171 is divisible by 9.

CODE:

```
#2b
num = int(input());
rem = sum = 0;

n = num;

while(num > 0):
    rem = num%10;
    sum = sum + rem;
    num = num//10;
```

```

if(n%sum == 0):
    print(str(n) + " is a harshad number");
else:
    print(str(n) + " is not a harshad number");

```

OUTPUT:

```

PS D:\Code\kmit> & C:/Users/akuld/AppData/Local/Microsoft/WindowsApps/python3.9.exe d:/Code/kmit/python/syl/2b.py
408
408 is a harshad number
● PS D:\Code\kmit> & C:/Users/akuld/AppData/Local/Microsoft/WindowsApps/python3.9.exe d:/Code/kmit/python/syl/2b.py
123
123 is not a harshad number

```

c) Given a number n, write an efficient function to print all unique prime factors of n. For example, if the input number is 315, then output should be "3 5 7".

CODE:

```

#2c
import math

def primefac(n):
    l = []
    while n % 2 == 0:

        l.append(2)
        n = n / 2
    for i in range(3, int(math.sqrt(n)) + 1, 2):
        while n % i == 0:
            l.append(i)
            n = n / i
    if n > 2:
        l.append(n)

    l = list(set(l))
    for i in l:
        i = int(i)
        print(i, end=' ')

```

```

n = int(input('enter number:'))
primefac(n)

```

OUTPUT:

```

● PS D:\Code\kmit> & C:/Users/akuld/AppData/Local/Microsoft/WindowsApps/python3.9.exe d:/Code/kmit/python/syl/2c.py
enter number:75
○ 3 5

```

d) A semiprime number is an integer which can be expressed as a product of two distinct primes. For example, $15 = 3 \times 5$ is a semiprime number but $9 = 3 \times 3$ is not. Given an integer number N, find whether it can be expressed as a sum of two semi-primes or not (not necessarily distinct).

CODE:

```

def prime(n):
    if n < 2:
        return False
    for i in range(2, int(n**0.5)+1):
        if n % i == 0:
            return False
    return True

def semiprime(n):

    for i in range(2, int(n**0.5)+1):
        if n % i == 0:
            j = n // i
            if prime(i) and prime(j) and i != j:
                return True
    return False

def sum(n):
    for i in range(2, n):
        if semiprime(i) and semiprime(n-i):
            return True
    return False

x = sum(int(input('enter the number:'))))
print('sum of semiprimes') if x else print('not sum of semiprimes')

```

OUTPUT:

```

● PS D:\Code\kmit> & C:/Users/akuld/AppData/Local/Microsoft/WindowsApps/python3.9.exe d:/Code/kmit/python/syl/2d.py
enter the number:35
sum of semiprimes
● PS D:\Code\kmit> & C:/Users/akuld/AppData/Local/Microsoft/WindowsApps/python3.9.exe d:/Code/kmit/python/syl/2d.py
enter the number:18
not sum of semiprimes

```

3. a) You are given a string which is a password, write a program to check if the password entered by the user is a valid or not. If it is a valid password then print “valid” else print “Invalid”

Rules of the password:

- i) Minimum length is 6 characters and maximum length is 15 characters.
- ii) It should have a minimum of one Capital Letter, One digit and one special symbol
- iii) Special symbols allowed are only (*, @, #)

CODE:

```

#3a
def check(password):
    if len(password) < 8 or len(password) > 15:
        return "Invalid"
    cap = dig = special = low = False
    count = 0
    for char in password:
        if char.isupper():
            cap = True
            count += 1
        elif char.islower():

```

```

        low = True
        count += 1
    elif char.isdigit():
        dig = True
        count += 1
    elif char in ['*', '@', '$']:
        special = True
        count += 1
    if cap and dig and special and (count == len(password)):
        return "Valid"
    else:
        return "Invalid"

```

```

password = input("Enter the password: ")
print(check(password))

```

OUTPUT:

```

● PS D:\Code\kmit> & C:/Users/akuld/AppData/Local/Microsoft/WindowsApps/python3.9.exe d:/Code/kmit/python/syl/3a.py
    Enter the password: Kmit123@
    Valid
● PS D:\Code\kmit> & C:/Users/akuld/AppData/Local/Microsoft/WindowsApps/python3.9.exe d:/Code/kmit/python/syl/3a.py
    Enter the password: kmit
    Invalid

```

- b) Given a string S having characters from English alphabets [‘a’ - ‘z’] and ‘.’ as the special character (without quotes). Write a program to construct the lexicographically smallest palindrome by filling each of the faded character (‘.’) with a lower-case alphabet.

The smallest lexicographical order is an order relation where string s is smaller than t, given the first character of s (s1) is smaller than the first character of t (t1), or in case they are equivalent, the second character, etc.

CODE:

```

#3b
s = input("Enter a string: ")
alpha = []
for i in range(97,123):
    alpha.append(chr(i))
for c in alpha:
    new = s.replace('. ',c)
if new==new[::-1]:
    print(new)
    break
else:
    print("Not possible to make the string as palindrome")

```

OUTPUT:

```

● PS D:\Code\kmit> & C:/Users/akuld/AppData/Local/Microsoft/WindowsApps/python3.9.exe d:/Code/kmit/python/syl/3b.py
    Enter a string: pe.p
    peep
● PS D:\Code\kmit> & C:/Users/akuld/AppData/Local/Microsoft/WindowsApps/python3.9.exe d:/Code/kmit/python/syl/3b.py
    Enter a string: r.t
    Not possible to make the string as palindrome

```

- c) Given an alphanumeric string S, write a program to extract maximum numeric value from that string. All the alphabets are in lower case. Take the maximum consecutive digits as a single number.

#3c

```

def extmax(string):
    num, res = 0, 0

    for i in range(len(string)):

        if string[i] >= "0" and string[i] <= "9":
            num = num * 10 + int(int(string[i]) - 0)
        else:
            res = max(res, num)
            num = 0

    return max(res, num)

string = input('enter the string :')

print(extmax(string))

```

OUTPUT:

```

● PS D:\Code\kmit> & C:/Users/akuld/AppData/Local/Microsoft/WindowsApps/python3.9.exe d:/Code/kmit/python/syl/3c.py
enter the string :23dsa43dsa98
98

```

4. a) Given a list A of elements of length N, ranging from 0 to N-1. All elements may not be present in the array. If the element is not present then there will be -1 present in the array. Rearrange the array such that A[i] = i and if i is not present then insert -1 at that place.

CODE:

```

#4a
def new_array(arr, n):
    s = set()
    for i in range(n):
        s.add(arr[i])
    for i in range(n):
        if i in s:
            arr[i] = i
        else:
            arr[i] = -1
    return arr

arr = [int(x) for x in input("Enter the elements seperated by space").split()]
n = len(arr)
print(new_array(arr, n))

```

OUTPUT:

```

● PS D:\Code\kmit> & C:/Users/akuld/AppData/Local/Microsoft/WindowsApps/python3.9.exe d:/Code/kmit/python/syl/4a.py
Enter the elements seperated by space : 1 -1 3 4 -1 5
[-1, 1, -1, 3, 4, 5]

```

- b) Given an array A of N numbers, you have to write a program which prints the sum of the elements of array A with the corresponding elements of the reverse of array A. If array A has elements [1,2,3], then reverse of the array A will be [3,2,1] and the resultant array should be [4,4,4].

CODE:

```
#4b
def rev_sum(A):
    n = len(A)
    result = []

    for i in range(n):
        result.append(A[i] + A[n - 1 - i])

    return result

A = [int(x) for x in input("Enter the elements seperated by space: ").split()]

print(rev_sum(A))
```

OUTPUT:

```
PS D:\Code\kmit> & C:/Users/akuld/AppData/Local/Microsoft/WindowsApps/python3.9.exe d:/Code/kmit/python/syl/4b.py
Enter the elements seperated by space : 4 6 8 2
[6, 14, 14, 6]
```

- c) Given a list A of N distinct integers, write a program to sort the list by moving an element to the end of the list. Find the minimum number of moves required to sort the list using this method in ascending order.

CODE:

```
#4c
def min_moves(A):
    l = []
    for i in A:
        l.append(i)
    l.sort()
    moves = 0

    for num in A:
        if num == l[0]:
            l.pop(0)
        else:
            moves += 1

    return moves

A = [int(x) for x in input("Enter the numbers seperated by spaces :").split()]
print("Minimum moves ->", min_moves(A))
```

OUTPUT:

```
PS D:\Code\kmit> & C:/Users/akuld/AppData/Local/Microsoft/WindowsApps/python3.9.exe d:/Code/kmit/python/syl/4c.py
Enter the numbers seperated by spaces :1 3 2 4 5
Minimum moves -> 3
```

5. a) A lower triangular matrix is a square matrix (where the number of rows and columns are equal) where all the elements above the diagonal are zero. Write a program to convert a given square matrix into a lower triangular matrix.

CODE:

```
def lowt_mat(matrix):
    n = len(matrix)
    for i in range(n):
        for j in range(n):
            if i < j:
                matrix[i][j] = 0
    return matrix
n = int(input("Enter number of rows: "))
matrix =[list(map(int, input().split())) for _ in range(n)]
x = lowt_mat(matrix)
print('Lower triangular matrix is:')
print("\n".join([" ".join(map(str, row)) for row in x]))
```

OUTPUT:

```
PS D:\Code\kmit> & C:/Users/akuld/AppData/Local/Microsoft/WindowsApps/python3.9.exe d:/Code/kmit/python/syl/5a.py
Enter number of rows: 3
7 8 9
4 6 5
3 1 2
Lower triangular matrix is:
7 0 0
4 6 0
3 1 2
PS D:\Code\kmit>
```

- b) Given a square matrix, write a program to print it in a counter-clockwise spiral form.

CODE:

```
a = int(input("Enter number of rows: "))
mat=[]
print("Enter elements:")
for i in range(0,a):
    l = list(map(int, input().split()))
    mat.append(l)
m,n = a,a
k,l,count = 0,0,0
total = a*a
print("Counter-clockwise spiral form:")
while (k < m and l < n) :
    if (count == total) :
        break
    for i in range(k, m) :
        print(mat[i][l], end=" ")
        count += 1
    l += 1
    if (count == total) :
        break
    for i in range (l, n) :
        print(mat[m - 1][i], end = " ")
        count += 1
```

```

m -= 1
if (count == total) :
    break
if (k < m) :
    for i in range(m - 1, k - 1, -1) :
        print(mat[i][n - 1], end = " ")
        count += 1
n -= 1
if (count == total) :
    break
if (l < n) :
    for i in range(n - 1, l - 1, -1) :
        print( mat[k][i], end = " ")
        count += 1
    k += 1

```

OUTPUT:

```

● PS D:\Code\kmit > & C:/Users/akuld/AppData/Local/Microsoft/WindowsApps/python3.9.exe d:/Code/kmit/python/syl/5b.py
Enter number of rows: 4
Enter elements:
1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16
Counter-clockwise spiral form:
1 5 9 13 14 15 16 12 8 4 3 2 6 10 11 7

```

6. a) Given a number n, define a function named printDict() which can print a dictionary where the keys are numbers between 1 and n (both included) and the values are square of keys. The function printDict() doesn't take any argument.

CODE:

```

def printDict(n):

    dict = {}
    for i in range(1, n + 1):
        dict[i] = i ** 2
    print(dict)

num = int(input('Enter the number :'))
print('the resultant dictionary is:')
printDict(num)

```

OUTPUT:

```

● PS D:\Code\kmit > & C:/Users/akuld/AppData/Local/Microsoft/WindowsApps/python3.9.exe d:/Code/kmit/python/syl/6a.py
Enter the number :5
the resultant dictionary is:
{1: 1, 2: 4, 3: 9, 4: 16, 5: 25}

```

b) Write a program to find the most common scores in the list of scores given in sorted order based on occurrence from largest to smallest. If any of scores are having same occurrence then consider the largest score first.

Input format: First line contains the list of scores and next line contains a number (k) which represent the top most scores to display.

Output format: display the k top most scores.

CODE:

```
def common(scores, k):
    count = {}
    for i in scores:
        count[i] = count.get(i, 0) + 1

    result = sorted(count.keys(), key=lambda x: (-count[x], -x))[:k]

    return result

scores = list(map(int, input().split()))
k = int(input('enter the number of top common scores required:'))

print("Top", k, "most common scores:", common(scores, k))
```

OUTPUT:

```
1 2 3 7 8 12 10 15 17 13 5 3 6 7 11 16
● PS D:\Code\kmit> & C:/Users/akuld/AppData/Local/Microsoft/WindowsApps/python3.9.exe d:/Code/kmit/python/syl/6b.py
1 2 2 2 2 3 4 7 8 8
enter the number of top common scores required:4
Top 4 most common scores: [2, 8, 7, 4]
```

7. a) Write a program that loads roll numbers and names from the given CSV file into dictionary where data is organized as one row per record. It takes a roll number or name as input and prints the corresponding other value from dictionary.

```
#download file from : http://github.com/akuldeepj/data/blob/main/students.csv
import csv
with open('students.csv','r') as f:
    8. Import the iris dataset from https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data
    a) Find the mean, median, standard deviation of iris's 'sepal length' (1st column).
        reader = csv.reader(f)
        d = {}
        print(reader)
        for row in reader:
            d[row[0]] = row[1]
        roll = input("Enter roll number: ")
        #capitalize letters in roll
        roll = roll.upper()
        if roll in d:
            print("Roll number of the student is: ",d[roll])
        else:
            print("Roll number not found")
```

```
Enter roll number: 22bd1a0501
Roll number of the student is: AKULDEEP JAKKULA
```

- b) Write a program to find the frequency of distinct words in the given text file and store the words along with the frequency in a CSV file.

```
import csv

def word_count(str):
    counts = dict()
    words = str.split()

    for word in words:
        if word in counts:
            counts[word] +=1
        else:
            counts[word] = 1
    return counts

with open('text.txt', 'r') as file:
    data = file.read().replace('\n', '')

word_count_dict = word_count(data)

with open('word_count.csv', 'w') as csv_file:
    writer = csv.writer(csv_file)
    for key, value in word_count_dict.items():
```

	writer.writerow([key, value])
welcome	1
to	2
KMITwelcome	1
python	1
programming	1
lab	1

7a.py 7b.py text.txt X

text.txt

```

1 welcome to KMIT
2 welcome to python programming lab

```

```

import pandas as pd
iris = pd.read_csv('https://archive.ics.uci.edu/ml/machine-learning-
databases/iris/iris.data', header=None)
iris.columns =
['sepal_length', 'sepal_width', 'petal_length', 'petal_width', 'class']
print('mean :', iris['sepal_length'].mean())
print('median :', iris['sepal_length'].median())
print("standard deviation :", iris['sepal_length'].std())

mean : 5.843333333333334
median : 5.8
standard deviation : 0.828066127977863

```

- b) Filter the rows of iris_2d that has petallength (3rd column) > 1.5 and sepallength (1st column) < 5.0

```

import pandas as pd

iris = pd.read_csv('https://archive.ics.uci.edu/ml/machine-learning-
databases/iris/iris.data', header=None)
iris_2d = iris.iloc[:, [0,1,2,3,4]]
iris_2d.columns = ['sepal_length', 'sepal_width', 'petal_length',
'petal_width', 'class']

print(iris_2d[(iris_2d['petal_length'] > 1.5) & (iris_2d['sepal_length'] <
5.0)])

```

	sepal_length	sepal_width	petal_length	petal_width	class
11	4.8	3.4	1.6	0.2	Iris-setosa
24	4.8	3.4	1.9	0.2	Iris-setosa
29	4.7	3.2	1.6	0.2	Iris-setosa
30	4.8	3.1	1.6	0.2	Iris-setosa
57	4.9	2.4	3.3	1.0	Iris-versicolor
106	4.9	2.5	4.5	1.7	Iris-virginica

9. Import the Cars93 dataset into a data frame from https://raw.githubusercontent.com/selva86/datasets/master/Cars93_miss.csv

a) Check if dataframe has any missing values.

```

import pandas as pd

#why .sum().sum()?
#df.isnull().sum() gives the sum of null values in each column
#df.isnull().sum().sum() gives the total sum of null values in the dataframe

df =
pd.read_csv('https://raw.githubusercontent.com/selva86/datasets/master/Cars93_
miss.csv')
print(df.isnull().sum().sum())

```

9a.py
null values : 170

b) Count the number of missing values in each column of dataframe. Which column has the maximum number of missing values?

```

import pandas as pd

df =
pd.read_csv('https://raw.githubusercontent.com/selva86/datasets/master/Cars93_
miss.csv')
print(df.isnull().sum().sort_values(ascending=False))
print('max value:', df.isnull().sum().idxmax())

```

```

      AirBags          38
Luggage.room         19
MPG.city              9
Fuel.tank.capacity     8
DriveTrain             7
Horsepower              7
Min.Price              7
Weight                  7
Rev.per.mile            6
Width                   6
Cylinders                5
Turn.circle               5
Max.Price                5
Man.trans.avail            5
Origin                  5
Rear.seat.room            4
Length                   4
Manufacturer              4
RPM                      3
Type                      3
Make                      3
Passengers                2
EngineSize                 2
MPG.highway                2
Price                      2
Wheelbase                  1
Model                      1
dtype: int64
max value: AirBags

```

c) Replace missing values in Min.Price and Max.Price columns with their respective mean.

```

import pandas as pd

df =
pd.read_csv('https://raw.githubusercontent.com/selva86/datasets/master/Cars93_
miss.csv')
print('min :',df['Min.Price'].fillna(df['Min.Price'].mean()))
print('max :',df['Max.Price'].fillna(df['Max.Price'].mean()))

```

```
min : 0      12.900000
1      29.200000
2      25.900000
3      17.118605
4      17.118605
...
88     16.600000
89     17.600000
90     22.900000
91     21.800000
92     24.800000
Name: Min.Price, Length: 93, dtype: float64
max : 0      18.800000
1      38.700000
2      32.300000
3      44.600000
4      21.459091
...
88     22.700000
89     22.400000
90     23.700000
91     23.500000
92     28.500000
Name: Max.Price, Length: 93, dtype: float64
```

10. a) A number raised to the third power is a *cube*. Plot the first five cubic numbers, and then plot the first 5000 cubic numbers.

```
import matplotlib.pyplot as plt

x_values = list(range(1, 6))
y_values = [x**3 for x in x_values]

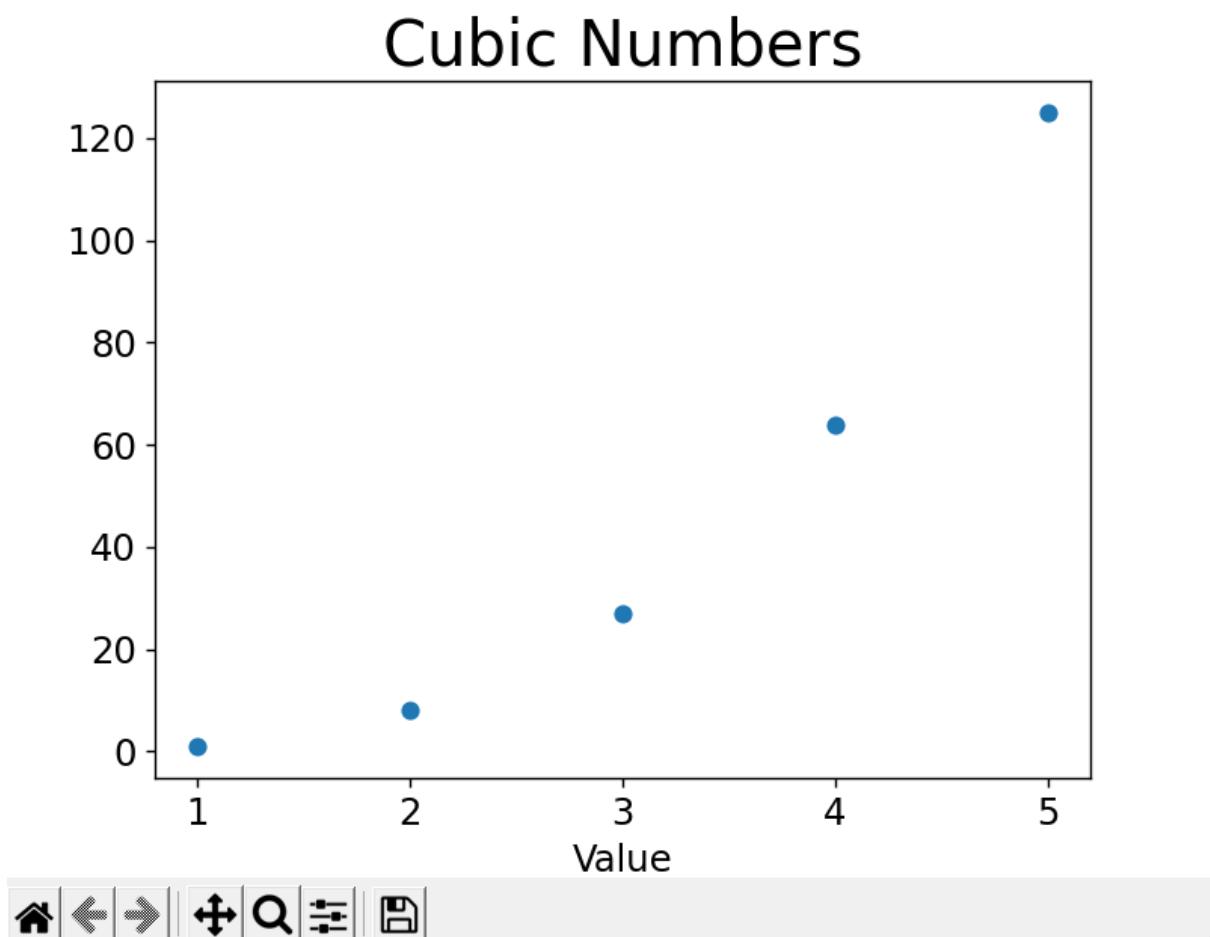
plt.scatter(x_values, y_values)

plt.title("Cubic Numbers", fontsize=24)
plt.xlabel("Value", fontsize=14)

plt.tick_params(axis='both', which='major', labelsize=14)

plt.show()
```

Figure 1



b) Colored Cubes: Apply a colormap to your cubes plot.

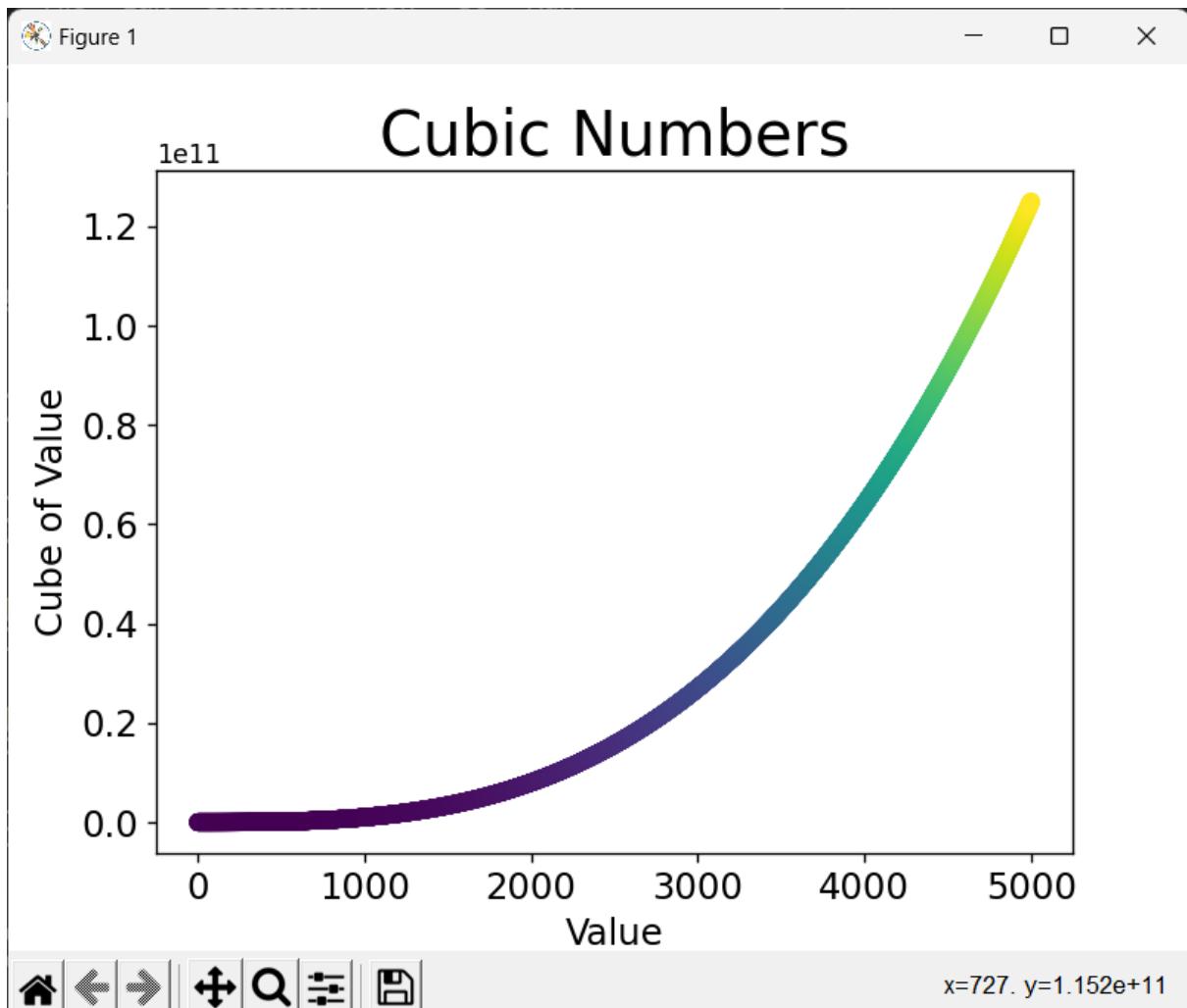
```
import matplotlib.pyplot as plt
x_values = list(range(1, 5001))
y_values = [x**3 for x in x_values]

plt.scatter(x_values, y_values, c=y_values)

plt.title("Cubic Numbers", fontsize=24)
plt.xlabel("Value", fontsize=14)
plt.ylabel("Cube of Value", fontsize=14)

plt.tick_params(axis='both', which='major', labelsize=14)

plt.show()
```



11. Sitka is in a temperate rainforest, so it gets a fair amount of rainfall. In the data file `sitka_weather_2018_simple.csv` is a header called PRCP, which represents daily rainfall amounts. Make a visualization focusing on the data in this column.

```

import matplotlib.pyplot as plt
import pandas as pd

url =
'https://raw.githubusercontent.com/ehmatthes/pcc_2e/master/chapter_16/the_csv_
file_format/data/sitka_weather_2018_simple.csv'
df = pd.read_csv(url)

# Histogram
df['PRCP'].plot(kind='hist', bins=50, figsize=(6, 4))

# Scatter plot
df.plot(x='DATE', y='PRCP', kind='scatter', figsize=(6, 5), title='Rainfall in
2018')

# Box plot
df.boxplot(column='PRCP', by='TMAX', figsize=(10, 6))

```

```

# Average Rainfall by Date
df[['DATE', 'PRCP']].groupby('DATE').mean().sort_values(by='PRCP',
ascending=False).plot.bar(figsize=(10, 6), rot=10, title='Avg Rainfall by
Date')

# Average Rainfall by TMAX
df[['TMAX', 'PRCP']].groupby('TMAX').mean().sort_values(by='PRCP',
ascending=False).plot.bar(figsize=(10, 6), rot=10, title='Avg Rainfall by
TMAX')

plt.show()

```

