1a) Write a python program to find the best of two test average marks out of three test's marks accepted from the user.

```
# Version 1: using conditional statement
mark1 = int(input('Enter the Mark of Test 1: '))
mark2 = int(input('Enter the Mark of Test 2: '))
mark3 = int(input('Enter the Mark of Test 3: '))

if (mark1 < mark2) and (mark1 < mark3):
    total_marks = mark2 + mark3
elif (mark2 < mark1) and (mark2 < mark3):
    total_marks = mark1 + mark3
else:
    total_marks = mark1 + mark2

average_marks = total_marks / 2;
print('Total of 2 Best Scores: ', total_marks)
print('Average of 2 Best Scores: ', average_marks)

# Version 2: without using conditional statement
mark1 = int(input('Enter the Mark of Test 1: '))
mark2 = int(input('Enter the Mark of Test 2: '))
mark3 = int(input('Enter the Mark of Test 3: '))

total_marks = mark1 + mark2 + mark3
```

```
total_marks = total_marks - min(mark1, mark2, mark3)
average_marks = total_marks / 2;
print('Total of 2 Best Scores: ', total_marks)
print('Average of 2 Best Scores: ', average_marks)

# Version 3: using list
marks = []
for i in range(3):
    mark = int(input('Enter the marks of Test %d : '%
(i+1)))
    marks.append(mark)
total_marks = sum(marks) - min(marks)

average_marks = total_marks / 2;

print('Total of 2 Best Scores: ', total_marks)
print('Average of 2 Best Scores: ', average_marks)
```

1b) Develop a Python program to check whether a given number is palindrome or not and also count the number of occurrences of each digit in the input number.

```
# Version 1: Using Numbers and List
num = int(input('Enter an integer: '))
temp = num
rev = 0;
digits_count = [0]* 10
while num > 0:
    rem = num % 10
    rev = (rev * 10) + rem
    num = num // 10
    digits_count[rem] = digits_count[rem]+1
if temp == rev:
    print(f'{temp} is a palindrome number')
else:
    print(f'{temp} is NOT a palindrome number')

for i in range(len(digits_count)):
    print(f'digit {i} occurs {digits_count[i]} times')

# Version 2: Using Strings and Dictionary
num = int(input('Enter an integer: '))
num = str(num)
rev = num[::-1]
```

```
if num == rev:
    print(f'{num} is a palindrome number')
else:
     print(f'{num} is NOT a palindrome number')
digit_count = {}
for ch in num:
     digit_count[ch] = digit_count.get(ch, 0) + 1
for (key, value) in sorted(digit_count.items()):
     print(f'{key} occurs {value} times')
```

2a) Defined as a function F as F(n) = F(n-1) + F(n-2). Write a Python program which accepts a value for N (where N >0) as input and pass this value to the function. Display suitable error message if the condition for input value is not followed.

```
def fib1(n):
    a, b = 0, 1
    for _ in range(n):
        print(a, end='  ')
        a, b = b, a + b
    print()


def fib2(n):
    if n <= 1:
        return n
    else:
        return fib2(n-2) + fib2(n-1)
n = int(input('How many terms would you like? '))
if n <= 0:
    print('Please Enter a value greater than Zero')
else:
    print('Fibonacci Sequence using Approach 1: ')
    fib1(n)
    print('Fibonacci Sequence using Approach 2: ')
    for i in range(n):
        print(fib2(i), end='  ')
    print()
```

2b) Develop a python program to convert binary to decimal, octal to hexadecimal using functions.

```
def bin2dec(n):
    bin = power = 0
    while n > 0:
        rem = n % 10
        bin = bin + rem * (2**power)
        power += 1
        n = n // 10
    return bin


def oct2hex(n):
    dec = power = 0
    while n > 0:
        rem = n % 10
        dec = dec + rem * (8 ** power)
        power += 1
        n = n // 10
    conversion_table = {10: 'A', 11: 'B', 12:'C',
13:'D', 14:'E', 15:'F'}
    hex = ""
    while dec > 0:
        rem = dec % 16
        if rem >= 10:
            hex = conversion_table[rem] + hex
```

```
        else:
            hex = str(rem) + hex
        dec = dec // 16
    return hex

bin_num = int(input('Enter a Binary Number: '))
dec_num = bin2dec(bin_num)
print(f'The Decimal Equivalent of binary number
{bin_num} is: {dec_num}')

print('\n')

oct_num = int(input('Enter a Octal Number: '))
hex_num = oct2hex(oct_num)
print(f'The Hexadecimal Equivalent of octal number
{oct_num} is: {hex_num}')
```

3a) Write a Python program that accepts a sentence and find the number of words, digits, uppercase letters and lowercase letters.

```
sentence = input('Enter a sentence:\n')
(word_count, digit_count, upper_count, lower_count) =
(0,0,0,0)
word_count = len(sentence.split())


for ch in sentence:
    if ch.isdigit():
        digit_count += 1
    elif ch.isupper():
        upper_count += 1
    elif ch.islower():
        lower_count += 1


print(f'No. of Words: {word_count}')
print(f'No. of Digits: {digit_count}')
print(f'No. of Uppercase Letters: {upper_count}')
print(f'No. of Lowercase Letters: {lower_count}')
```

3b) Write a Python program to find the string similarity between two given strings

```
str1 = input('Enter First string: \n')
str2 = input('Enter Second String: \n')

len1 = len(str1)
len2 = len(str2)

maxlen = max(len1, len2)
common_char_count = 0

for i in range(maxlen):
    if (i < len1) and (i< len2) and (str1[i] ==
str2[i] ):
        common_char_count += 1

similarity = common_char_count / maxlen

print(f'String1: {str1}')
print(f'String2: {str2}')

print(f'Similarity: {similarity:.2f}')
```

4a) Write a Python program to Demonstrate how to Draw a Bar Plot using Matplotlib.

```
import matplotlib.pyplot as plt


categories = ['Category A', 'Category B', 'Category C']
values = [10, 17, 25]


plt.bar(categories, values, width=0.3, align='edge')


plt.xlabel('Categories')
plt.ylabel('Values')
plt.title('Bar Plot')


plt.show()
```

4b) Write a Python program to Demonstrate how to Draw a Scatter Plot using Matplotlib.

```
import matplotlib.pyplot as plt


x = [1, 2, 3, 4, 5]
y = [10, 15, 27, 8, 19]


plt.scatter(x, y, c='red', marker='*')


plt.xlabel('X values')
plt.ylabel('Y values')
plt.title('Scatter Plot')


#plt.ylim((-5, 30))
#plt.xlim((0, 10))


plt.show()
```

5a) Write a Python program to Demonstrate how to Draw a Histogram Plot using Matplotlib.

```
import matplotlib.pyplot as plt
import numpy as np


np.random.seed(0)
data = np.random.randn(1000)


plt.hist(data, bins=20, color='skyblue',
edgecolor='black')


plt.xlabel('Value')
plt.ylabel('frequency')
plt.title('Histogram Plot')


plt.show()
```

5b) Write a Python program to Demonstrate how to Draw a Pie Chart using Matplotlib.

```
import matplotlib.pyplot as plt
labels = ['Excellent', 'Good', 'Average', 'Poor']
count = [17, 38, 20, 23]
colors = ['yellow', 'blue', 'green', 'red']

plt.pie(count, labels=labels, colors=colors,
autopct='%0.2f%%', startangle=120)
plt.title('Pie Chart')

plt.show()
```

6a) Write a Python program to illustrate Linear Plotting using Matplotlib.

```
import matplotlib.pyplot as plt


x = [1, 2, 3, 4, 5, 6]
y = [2, 5, 9, 8, 7, 3]


plt.plot(x, y,label='Linear plot')
plt.xlabel('X-axis')
plt.ylabel('Y-axis')


plt.title('Linear Plot')
plt.legend()
plt.grid(True)
plt.show()
```

6b) Write a Python program to illustrate liner plotting with line formatting using Matplotlib.

```
import matplotlib.pyplot as plt
x = [1, 2, 3, 4, 5, 6]
y = [2, 5, 9, 8, 7, 3]

plt.plot(x, y,label='Linear plot', color='red',
marker='o', linestyle='--', linewidth=2, markersize=8)

plt.xlabel('X-axis')
plt.ylabel('Y-axis')
plt.title('Linear Plot')

plt.legend()
plt.grid(True)
plt.show()
```

7) Write a Python program which explains uses of customizing seaborn plots with Aesthetic functions.

```
import matplotlib.pyplot as plt
import seaborn as sns

iris = sns.load_dataset('iris')

# print(iris.head(10))
# print(iris['species'].unique())
# print(iris['species'].value_counts())

sns.scatterplot(x='sepal_length',
y='sepal_width',data=iris,hue='species',
style='species', s=20,alpha=0.7, markers=['o','s','D'] )

plt.xlabel('Sepal Length')
plt.ylabel('Sepal Width')
plt.title('Customized Seaborn Scatter Plot')
plt.legend(title='Species')

plt.show()
```

8a) Write a Python program to explain working with bokeh line graph using Annotations and Legends.

```
from bokeh.plotting import figure, show, output_file
from bokeh.models import Label, Legend

x = [1, 2, 3, 4, 5]
y1 = [2, 4, 6, 8, 10]
y2 = [10, 8, 6, 4, 2]

p=figure(title='Line Graph with Annotations and
Legends', x_axis_label='X-axis', y_axis_label='Y-axis')

p.line(x, y1, legend_label='Line1', line_color='blue',
line_width=2)

p.line(x, y2, legend_label='Line2', line_color='red',
line_width=2, line_dash='dashed')

annotation1 = Label(x=3, y=7, text='Annotation1',
text_font_size='12pt', text_color='green')

annotation2 = Label(x=2, y=4, text='Annotation2',
text_font_size='12pt', text_color='purple')

p.add_layout(annotation1)
p.add_layout(annotation2)
```

```
legend = Legend(items=[('Line1',[p.line(x,y1)]),
('Line2',[p.line(x,y2)])])


p.add_layout(legend)
output_file('Lab8A.html')
show(p)
```

8b) Write a Python program for plotting different types of plots using Bokeh.

```
from bokeh.plotting import figure, show, output_file
from bokeh.layouts import row

output_file('Lab8B.html')
wh = 300
ht = 300
x_values = [1, 2, 3, 4, 5]
y_values = [2, 4, 6, 8, 10]

line_plot = figure(title="Line Plot", width=wh,
height=ht, x_axis_label="X-axis", y_axis_label="Y-axis")

line_plot.line(x_values, y_values, line_color="blue",
line_width=2)

scatter_plot = figure(title="Scatter Plot", width=wh,
height=ht, x_axis_label="X-axis", y_axis_label="Y-axis")

scatter_plot.scatter(x_values, y_values, size=5,
color="red")

bar_plot = figure(title="Bar Plot", width=wh, height=ht,
            x_axis_label="Rating", y_axis_label="Count")
bar_plot.vbar(x=x_values, top=y_values, width=0.5,
color="green")
```

```
x, y, radius = 0, 0, 1
start_angle = [0, 1.4, 2.5, 3.9, 5.6]
end_angle = [1.4, 2.5, 3.9, 5.6, 0]
colors = ['green', 'violet', 'red', 'yellow', 'blue']

pie_chart = figure(title="Pie Chart", width=wh,
height=ht)
pie_chart.wedge(x, y, radius, start_angle, end_angle,
color=colors)

show(row(line_plot, scatter_plot, bar_plot, pie_chart))
```

9) Write a Python program to draw 3D Plots using Plotly Libraries.

```
import plotly.express as px
import pandas as pd

data = pd.DataFrame({  'X': [1, 2, 3, 4, 5],
                       'Y': [5, 4, 3, 2, 1],
                       'Z': [1, 2, 3, 4, 5]
                    }
                    )
fig = px.scatter_3d(data, x='X', y='Y', z='Z')

fig.update_layout(scene=dict(xaxis_title='X-axis',
                             yaxis_title='Y-axis',
                             zaxis_title='Z-axis' ) )
fig.show()
```

10a) Write a Python program to draw Time Series using Plotly Library.

```
import plotly.express as px
import pandas as pd

data = pd.DataFrame({
'Date': pd.date_range(start='2023-01-01', periods=10,
freq='D'), 'Value': [10, 12, 14, 18, 22, 28, 30, 32, 29,
26]  })

fig = px.line(data, x='Date', y='Value', title='Time
Series Plot')

fig.update_xaxes(title_text='Date')
fig.update_yaxes(title_text='Value')

fig.show()
```

10b) Write a Python program for creating Maps using Plotly Library.

```
import plotly.express as px
import pandas as pd

data = pd.DataFrame({
'City': ['Bangalore', 'Mumbai', 'Delhi', 'Kolkata',
'Bhopal'],
'Lat': [12.9629, 18.9582, 28.7041, 22.5744, 23.2599],
'Lon': [77.5775, 72.8321, 77.1025, 88.3659, 77.4126],
'Population': [8398748, 3990456, 8716000, 2320255,
1680992]
})

fig = px.scatter_geo(data, lat='Lat', lon='Lon',
text='City', size='Population', projection="natural
earth", title='Sample City Population Map',scope='asia')


fig.update_geos(showcoastlines=True,
coastlinecolor="RebeccaPurple", showland=True,
landcolor="LightGreen",)
fig.show()
```