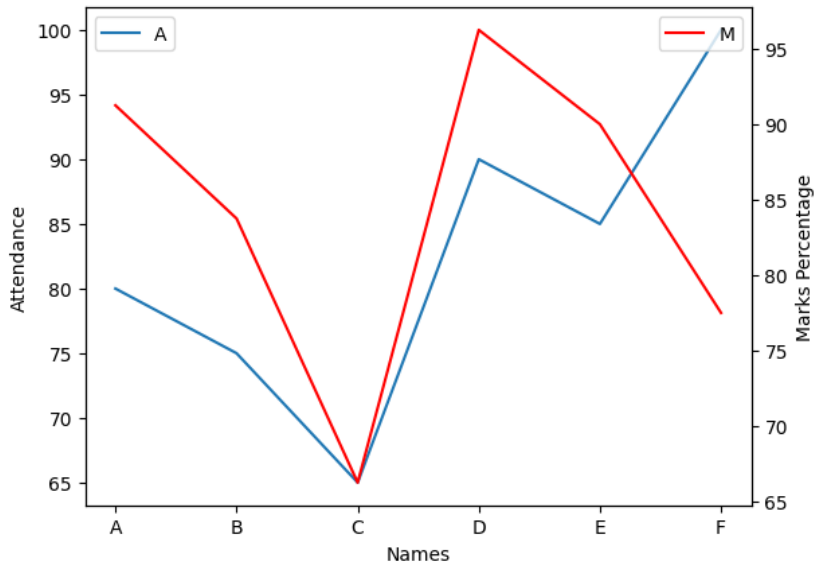
Q. Plot secondary axis in matplotlib with legends

A.

```
import matplotlib.pyplot as plt

name = df['Name']
marks = df['Marks']
percentage = df['Percentage']
attendance= df['Attendance']

fig1 = plt.figure()
fig1, ax= plt.subplots(1)

ax1 = ax.twinx()

def plot1(xcoord,ycoord,ax=None):
    ax=ax
    ax.plot(xcoord,ycoord)

def plot2(xcoord,ycoord,ax=None):
    ax=ax
    ax.plot(xcoord,ycoord,'red')

plot1(name,attendance,ax)
plot2(name,percentage,ax1)

ax.set_xlabel("Names")

ax.set_ylabel("Attendance")
ax1.set_ylabel("Marks Percentage")

ax.legend('Attendance')
ax1.legend('Marks Percentage')
```



Q. Use lambda operations on the dataframes

A.

```
df = df.assign(Days_Attended=lambda x: x['Attendance']*2.75)
print(df)
```

	Name	Marks	Attendance	Percentage	Days_Attended
0	A	365	80	91.25	220.00
1	B	335	75	83.75	206.25
2	C	265	65	66.25	178.75
3	D	385	90	96.25	247.50
4	E	360	85	90.00	233.75
5	F	310	100	77.50	275.00

Q. Code to check anagrams

A.

```
str1 = "bored"
str2 = "robed"
str3 = "faced"

newStr1 = "".join(sorted(str1))
newStr2 = "".join(sorted(str2))
newStr3 = "".join(sorted(str3))
```

```
if (newStr1 == newStr3):
    print("Anagram!")
else:
    print("Not an anagram!")
```

Q. Regex example in python

```
import re

regex = re.compile(r'([A-Za-z0-9]+[.-_])*[A-Za-z0-9]+@[A-Za-z0-9-]+(\.[A-Z|a-z]{2,})+')
emails = ["john.doe@example.com", "invalid_email", "user@domain-name.co.uk"]

for email in emails:
  if re.match(regex, email):
    print(f"{email} is a valid email address")
  else:
    print(f"{email} is not a valid email address")
```

Q. Explain three types of merges in git

A.

- Standard Merge: It creates a new merge commit that combines the changes from the branch you're merging into and the branch you're merging from. (Easy but can get cluttered)
- Squash Merge: This type of merge combines the commits from the branch you're merging from into a single, new commit before merging it into the branch you're merging into. (Cleaner but rewrites merge history)
- Rebase and Merge: This type of merge involves rewriting the history of the branch you're merging from on top of the branch you're merging into.