20_December_EDA

December 20, 2024

1 20 December

1.1 Write a program to find the maximum of 3 numbers using `if-else`

[9]:

```
a = int(input("Enter the first number:"))
b = int(input("Enter the second number:"))
c = int(input("Enter the third number:"))
if a>=b and a>=c:
    print("Largest is:",a)
elif b>=a and b>=c:
    print("Largest is:",b)
else:
    print("Largest is:",c)
```

```
Enter the first number: 4
Enter the second number: 4
Enter the third number: 2
```

```
Largest is: 4
```

1.2 Print all the prime numbers in 1 to 50

[22]:

```
def is_prime(n):
    if n<2:
        return False
    for i in range(2,int(n**.5)+1):
        if n%i == 0:
            return False
    return True
for i in range(1,51):
    if is_prime(i):
        print(i," is a prime")
```

```
2  is a prime
3  is a prime
5  is a prime
7  is a prime
11  is a prime
```

```
13  is a prime
17  is a prime
19  is a prime
23  is a prime
29  is a prime
31  is a prime
37  is a prime
41  is a prime
43  is a prime
47  is a prime
```

[ ]:

```
## Write a function to calculate the area of a circle given the radius
```

[30]:

```
import math
def area_cir(rad):
    return math.pi*rad**2
r = int(input("Enter the radius of the circle:"))
print(f"Area of the circle is:{area_cir(r)} sq unit")
```

```
Enter the radius of the circle: 5
```

```
Area of the circle is:78.53981633974483 sq unit
```

1.3 Use lambda function to filterout even number from the list

[28]:

```
og_list = [ x for x in range(1,51)]
k = lambda x: x%2==0
even_list = [x for x in og_list if k(x)]
print("Original list:\n",og_list)
print("Filtered list:\n",even_list)
```

```
Original list:
 [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22,
23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42,
43, 44, 45, 46, 47, 48, 49, 50]
Filtered list:
 [2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40,
42, 44, 46, 48, 50]
```

1.4 Alternate way

[37]:

```
og_list = [ x for x in range(1,51)]
even_list = list(filter(lambda x:x%2==0,og_list))
print("Original list:\n",og_list)
print("Filtered list:\n",even_list)
```

```
Original list:
 [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22,
23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42,
43, 44, 45, 46, 47, 48, 49, 50]
```

```
Filtered list:
 [2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40,
42, 44, 46, 48, 50]
```

[58]:

```
import pandas as pd
import numpy as np

## making the data
data = {"Name":
  ↪['S1','S2','S3','S4','S5','S6','S7','S8','S9','S10','S11','S12','S13','S14','S15','S16','S1
        "CGPA":[9.1,9.2,8.9,8.4,6.7,5.5,2.3,8.8,8.9,9.7,10,9.8,4.5,7.8,7.9,8.
  ↪1,7.6,7.5,8.9,5.5],
        "WEIGHTAGE_SUM":[91.0, 92.0, 89.0, 84.0, 67.0, 55.0, 23.0, 88.0, 89.0,␣
  ↪97.0, 100, 98.0, 45.0, 78.0, 79.0, 81.0, 76.0, 75.0, 89.0, 55.0]
       }
df = pd.DataFrame(data)
df
```

[58]:

```
   Name  CGPA  WEIGHTAGE_SUM
0    S1   9.1           91.0
1    S2   9.2           92.0
2    S3   8.9           89.0
3    S4   8.4           84.0
4    S5   6.7           67.0
5    S6   5.5           55.0
6    S7   2.3           23.0
7    S8   8.8           88.0
8    S9   8.9           89.0
9   S10   9.7           97.0
10  S11  10.0          100.0
11  S12   9.8           98.0
12  S13   4.5           45.0
13  S14   7.8           78.0
14  S15   7.9           79.0
15  S16   8.1           81.0
16  S17   7.6           76.0
17  S18   7.5           75.0
18  S19   8.9           89.0
19  S20   5.5           55.0
```

[63]:

```
df.to_csv("sample.csv",index =0)
```

[66]:

```
df1 = pd.read_csv("sample.csv")
```

[65]:

```
df1
```

```
[65]:     Name  CGPA  WEIGHTAGE_SUM
      0     S1   9.1           91.0
      1     S2   9.2           92.0
      2     S3   8.9           89.0
      3     S4   8.4           84.0
      4     S5   6.7           67.0
      5     S6   5.5           55.0
      6     S7   2.3           23.0
      7     S8   8.8           88.0
      8     S9   8.9           89.0
      9    S10   9.7           97.0
      10   S11  10.0          100.0
      11   S12   9.8           98.0
      12   S13   4.5           45.0
      13   S14   7.8           78.0
      14   S15   7.9           79.0
      15   S16   8.1           81.0
      16   S17   7.6           76.0
      17   S18   7.5           75.0
      18   S19   8.9           89.0
      19   S20   5.5           55.0
```

```
[67]: df['Grades'] = pd.cut(df["WEIGHTAGE_SUM"],bins =␣
        ↪[0,40,50,60,70,80,90,100],labels = ["F","E","D","C","B","A","S"])
```

```
[81]: df
```

```
[81]:     Name  CGPA  WEIGHTAGE_SUM Grades
      0     S1   9.1           91.0      S
      1     S2   9.2           92.0      S
      2     S3   8.9           89.0      A
      3     S4   8.4           84.0      A
      4     S5   6.7           67.0      C
      5     S6   5.5           55.0      D
      6     S7   2.3           23.0      F
      7     S8   8.8           88.0      A
      8     S9   8.9           89.0      A
      9    S10   9.7           97.0      S
      10   S11  10.0          100.0      S
      11   S12   9.8           98.0      S
      12   S13   4.5           45.0      E
      13   S14   7.8           78.0      B
      14   S15   7.9           79.0      B
      15   S16   8.1           81.0      A
      16   S17   7.6           76.0      B
      17   S18   7.5           75.0      B
      18   S19   8.9           89.0      A
```

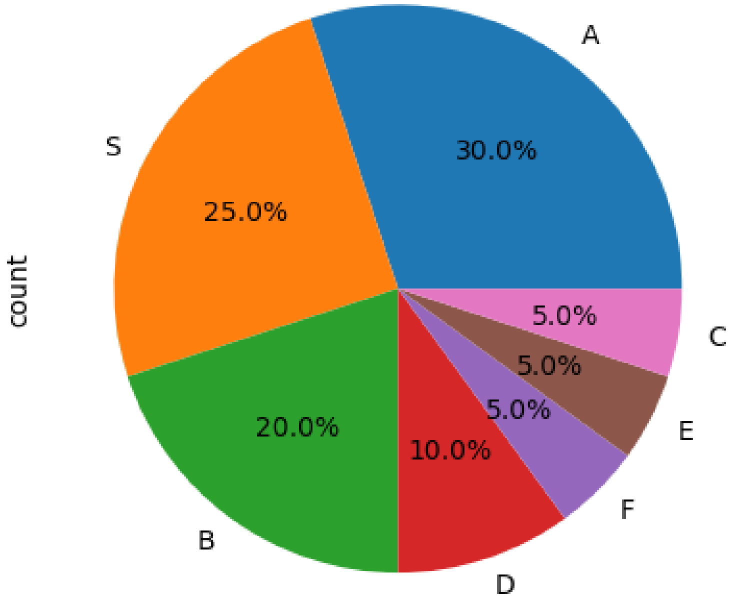
```
19   S20    5.5             55.0       D
```

```
[82]: grade_count = df["Grades"].value_counts()
```

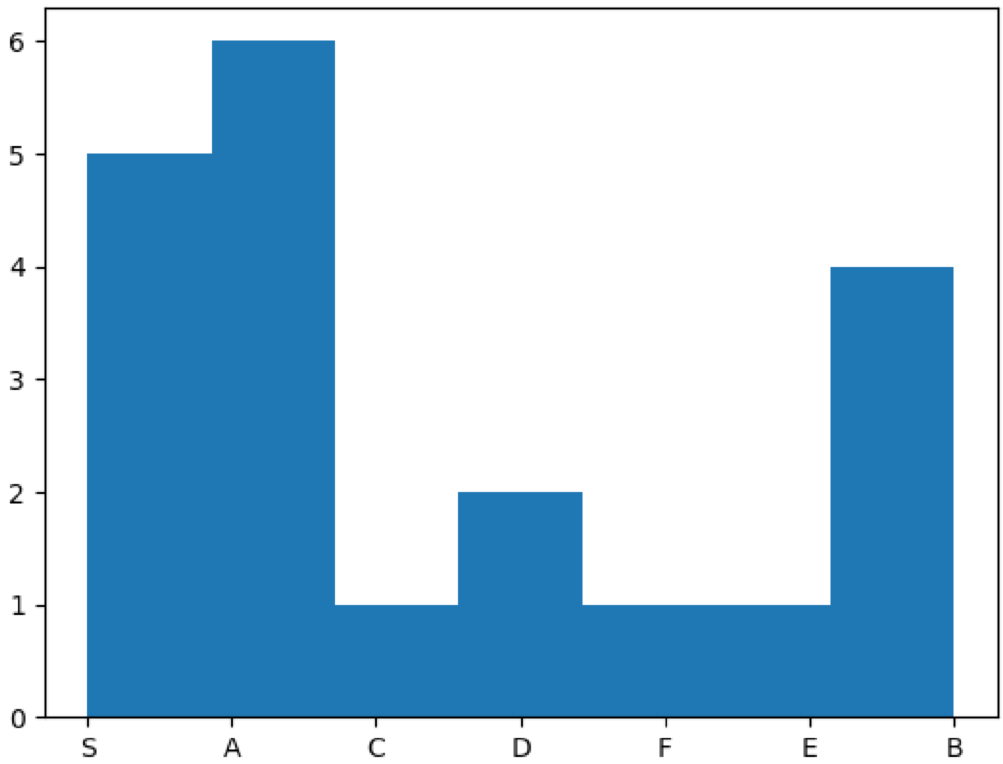
```
[83]: grade_count
```

```
[83]: Grades
      A    6
      S    5
      B    4
      D    2
      F    1
      E    1
      C    1
      Name: count, dtype: int64
```

```
[94]: grade_count.plot.pie(autopct="%1.1f%%")
      plt.show()
```

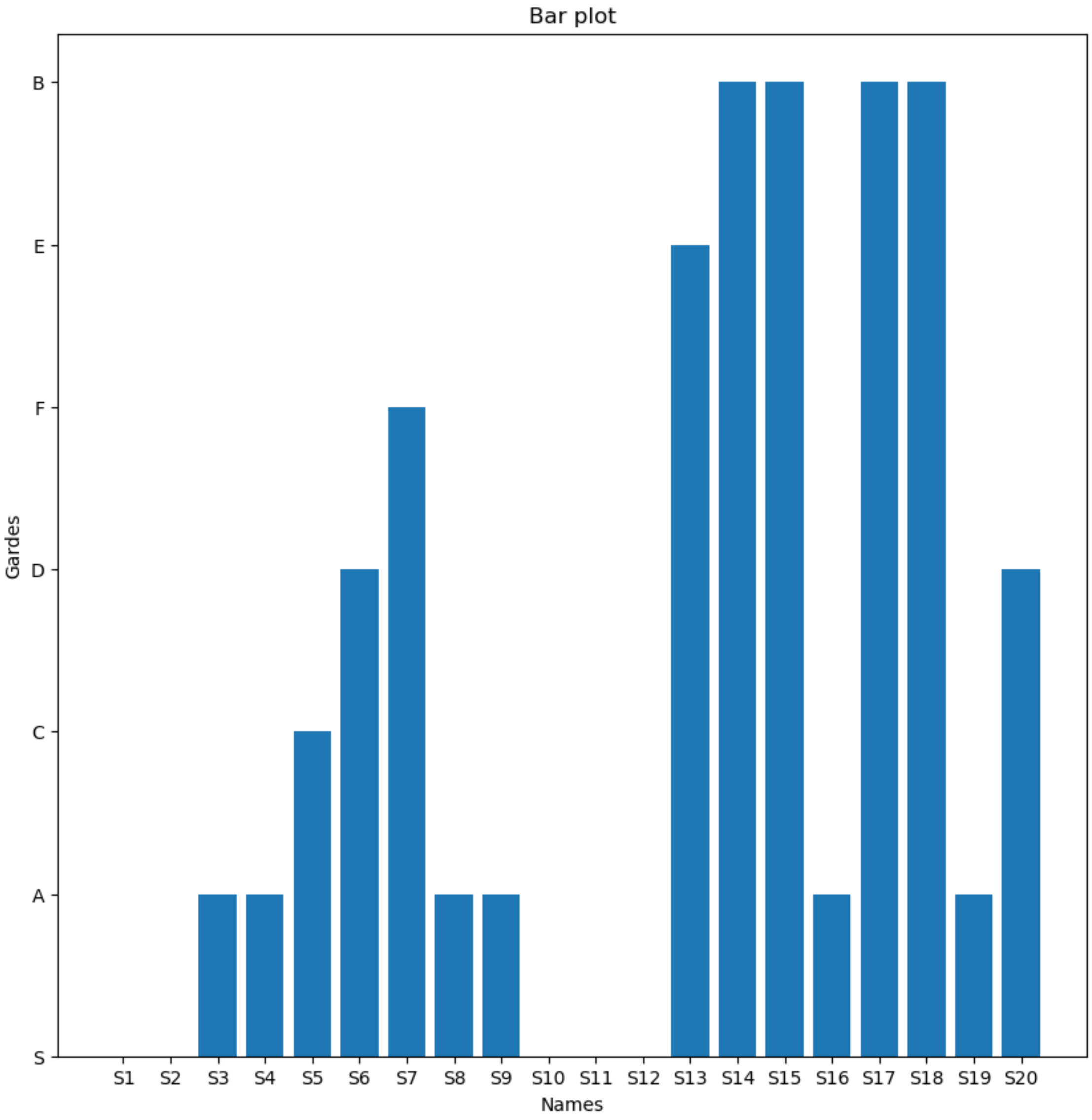


```
[101]: import matplotlib.pyplot as plt
       plt.hist(df["Grades"],bins=7)
       plt.show()
```

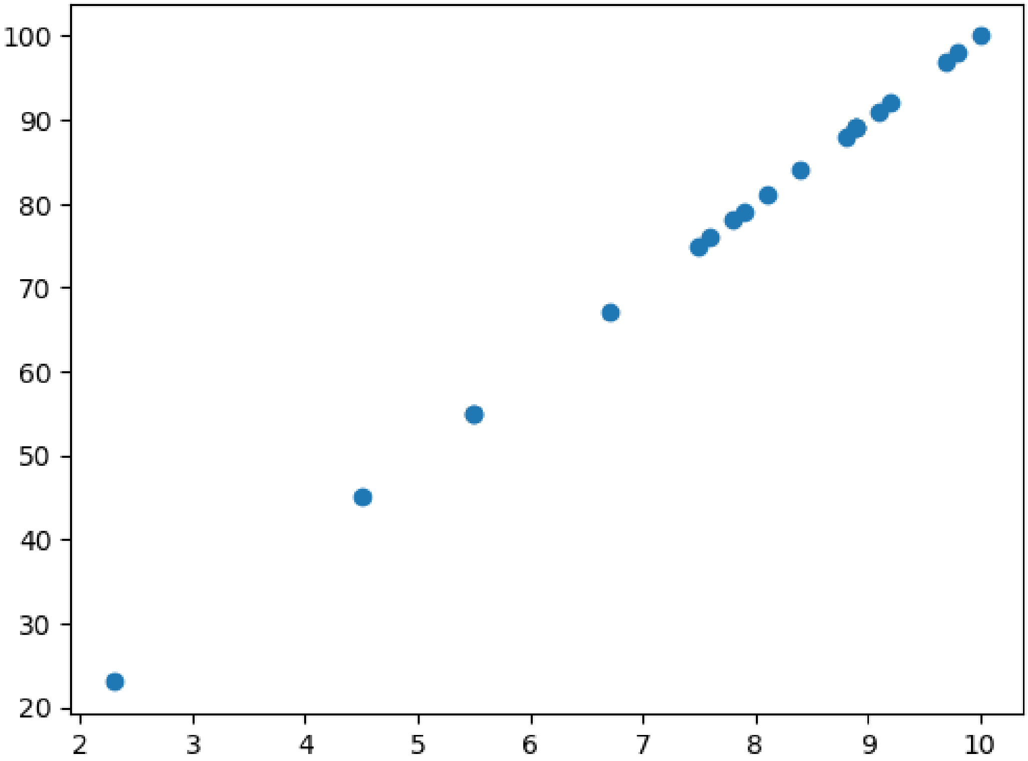


[109]:

```
plt.figure(figsize = (10,10))
plt.bar(df["Name"],df["Grades"])
plt.title("Bar plot")
plt.xlabel("Names")
plt.ylabel("Gardes")
plt.show()
```



[99]:
```
plt.scatter(df["CGPA"],df["WEIGHTAGE_SUM"])
plt.show()
```



[130]:

```
## Introducing missing Values

df.loc[2,"WEIGHTAGE_SUM"] = np.nan
df.loc[3,"WEIGHTAGE_SUM"] = np.nan
df.loc[7,"WEIGHTAGE_SUM"] = np.nan
df.loc[1,"WEIGHTAGE_SUM"] = np.nan
```

[131]:

```
df
```

[131]:

```
   Name  CGPA  WEIGHTAGE_SUM Grades
0    S1   9.1      91.000000      S
1    S2   9.2            NaN      S
2    S3   8.9            NaN      A
3    S4   8.4            NaN      A
4    S5   6.7      67.000000      C
5    S6   5.5      55.000000      D
6    S7   2.3      23.000000      F
7    S8   8.8            NaN      A
8    S9   8.9      89.000000      A
9   S10   9.7      97.000000      S
10  S11  10.0     100.000000      S
11  S12   9.8      98.000000      S
```

```
12  S13   4.5      45.000000      E
13  S14   7.8      78.000000      B
14  S15   7.9      79.000000      B
15  S16   8.1      81.000000      A
16  S17   7.6      76.000000      B
17  S18   7.5      75.000000      B
18  S19   8.9      89.000000      A
19  S20   5.5      55.000000      D
20   S2   9.2      92.000000      S
21   S3   8.9      75.882353      A
```

[134]:
```
df.dropna(inplace = True)
```

[135]:
```
df.isna().sum()
```

[135]:
```
Name             0
CGPA             0
WEIGHTAGE_SUM    0
Grades           0
dtype: int64
```

[122]:
```
## filling the missing values
df["WEIGHTAGE_SUM"].fillna(df["WEIGHTAGE_SUM"].mean())
df
```

[122]:
```
   Name  CGPA  WEIGHTAGE_SUM Grades
0    S1   9.1      91.000000      S
1    S2   9.2      75.882353      S
2    S3   8.9      75.882353      A
3    S4   8.4      75.882353      A
4    S5   6.7      67.000000      C
5    S6   5.5      55.000000      D
6    S7   2.3      23.000000      F
7    S8   8.8      75.882353      A
8    S9   8.9      89.000000      A
9   S10   9.7      97.000000      S
10  S11  10.0     100.000000      S
11  S12   9.8      98.000000      S
12  S13   4.5      45.000000      E
13  S14   7.8      78.000000      B
14  S15   7.9      79.000000      B
15  S16   8.1      81.000000      A
16  S17   7.6      76.000000      B
17  S18   7.5      75.000000      B
18  S19   8.9      89.000000      A
19  S20   5.5      55.000000      D
20   S2   9.2      92.000000      S
```

```
21   S3   8.9      75.882353      A
```

```
[123]: df.isna().sum()
```

```
[123]: Name             0
       CGPA             0
       WEIGHTAGE_SUM    0
       Grades           0
       dtype: int64
```

```
[110]: ## Duplicate  values
```

```
[112]: df = pd.concat([df,df.iloc[1:3]],ignore_index = True)
       df
```

```
[112]:    Name  CGPA  WEIGHTAGE_SUM Grades
       0    S1   9.1           91.0      S
       1    S2   9.2           92.0      S
       2    S3   8.9            NaN      A
       3    S4   8.4           84.0      A
       4    S5   6.7           67.0      C
       5    S6   5.5           55.0      D
       6    S7   2.3           23.0      F
       7    S8   8.8           88.0      A
       8    S9   8.9           89.0      A
       9   S10   9.7           97.0      S
       10  S11  10.0          100.0      S
       11  S12   9.8           98.0      S
       12  S13   4.5           45.0      E
       13  S14   7.8           78.0      B
       14  S15   7.9           79.0      B
       15  S16   8.1           81.0      A
       16  S17   7.6           76.0      B
       17  S18   7.5           75.0      B
       18  S19   8.9           89.0      A
       19  S20   5.5           55.0      D
       20   S2   9.2           92.0      S
       21   S3   8.9            NaN      A
```

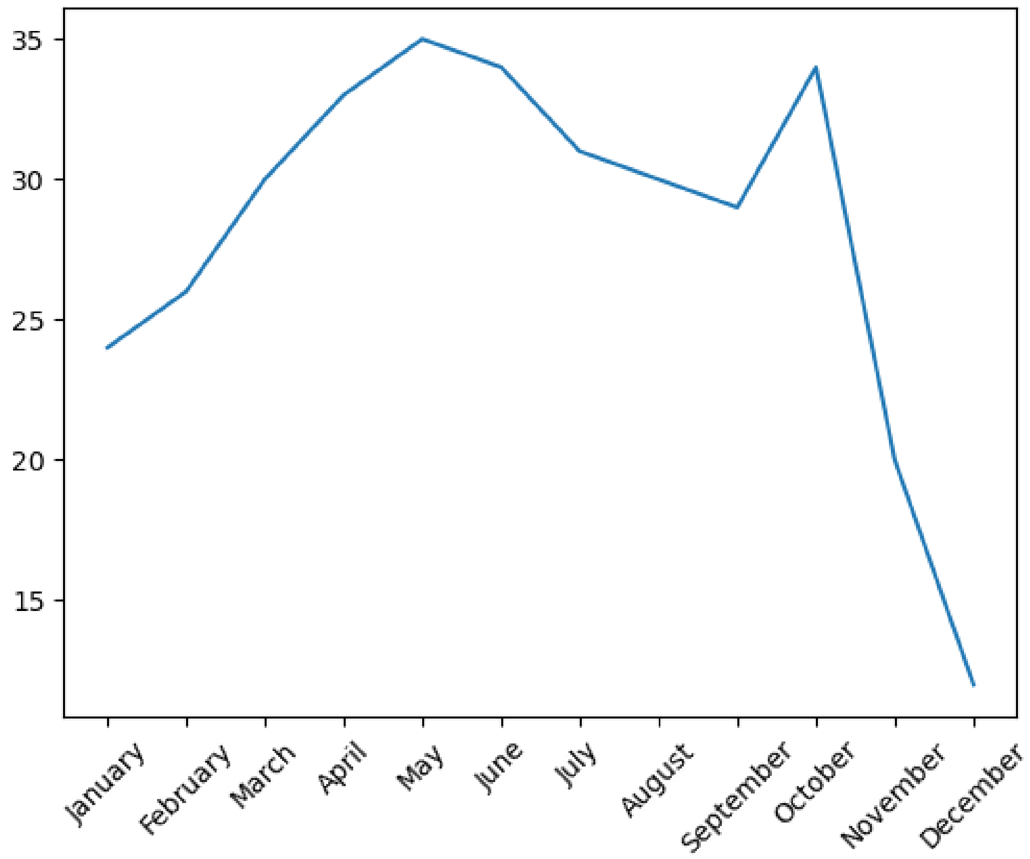
```
[113]: df.duplicated()
```

```
[113]: 0     False
       1     False
       2     False
       3     False
       4     False
       5     False
```

```
6     False
7     False
8     False
9     False
10    False
11    False
12    False
13    False
14    False
15    False
16    False
17    False
18    False
19    False
20     True
21     True
dtype: bool
```

[129]:

```
import matplotlib.pyplot as plt
# sample data
months =␣
 ↪['January','February','March','April','May','June','July','August','September','October','N
temperatures = [24,26,30,33,35,34,31,30,29,34,20,12]
plt.plot(months,temperatures)
plt.xticks(rotation = 45)
plt.show()
```



[ ]:

```python
import requests
## fetch data from public api
url = "https://randomuser.me/api"
response = requests.get(url)
if response.status_code == 200:
    data = response.json()
    print("Sample API Response")
    print(data)
```

[ ]:

```

```