Report for Python

- Task 2-A: Using the “Outside Temperature” values:

# a. What is the average time of hottest daily temperature (over month)

<u>Answer</u>:

```
x=df_data.groupby('Date')['Outside Temperature'].max()
a=x.reset_index(name='max_temp')
#lets get the other variables of max_temp values from the table
b=a.merge(df_data,how='left',left_on=['max_temp','Date'],right_on=['Outside Temperature','Date'])
# lets get the variable that we need for solving these problem as date time and temp
var = b[['Date','Time','max_temp']]
# lets get the format of month and date
var['Time'] = pd.to_datetime(var['Time'], format ='%H:%M')
var.index = pd.to_datetime(var['Date'], format ='%d/%m/%Y')
#average time of hottest daily temperature
c = var.groupby(var.index.month)['Time'].mean()
avg_temp=c.dt.time.reset_index(name='max_temp')
print(avg_temp[['Date','max_temp']])
with open('Outside_Temperature.txt','w+') as f:
   f.write('Month \t max_Temp_Time \n')
   for index, row in avg_temp.iterrows():
      row_data=[str(row['Date']),'\t\t',str(row['max_temp']),'\n']
      f.writelines(row_data)
   f.close()
```

<u>Explanation</u>: let’s find the max temperature for every day and get the other variables of max_temp values from the table. Get the format of month and date and calculate the average time of hottest daily temperature.

<u>Output:</u>

```
   Date  max_temp
0     5  14:40:00
1     6  13:38:00
2     7  08:50:00
```

# b. What time of the day is the most commonly occurring hottest time

Answer:

```
x=df_data.groupby('Date')['Outside Temperature'].max()

a=x.reset_index(name='max_temp')

b=a.merge(df_data,how='left',left_on=['max_temp','Date'],right_on=['Outside Temperature','Date'])

var = b[['Date','Time','max_temp']]

c=var.groupby('Time',as_index = False).size()

H=c['Time'].where(c['size']== max(c['size'])).dropna()

print(H)

with open('Outside_Temperature.txt','a') as f:

   f.write('2-A(b) time of the day is the most commonly occurring hottest time \n')

   for row in H:

      row_data=[str(row),'\t']

      f.writelines(row_data)

   f.close()
```

Explanation: Get the max outside temperature for each day. Merge the Max temp and outside temp for each day and get it into the variable. Group by time and then retrieve the max occurring temperature.

Output:

```
13    13:00
23    14:40
24    14:50
25    15:00
Name: Time, dtype: object
```

# c. Which are the Top Ten hottest times on distinct days, preferably sorted by date order.

Answer:

```
x=df_data.groupby('Date')['Outside Temperature'].max()

a=x.reset_index(name='max_temp')

b=a.merge(df_data,how='left',left_on=['max_temp','Date'],right_on=['Outside Temperature','Date'])

var = b[['Date','Time','max_temp']]

var= var.groupby('Date').max()

var=var.sort_values(by='max_temp',ascending=False).head(10)

var=var.sort_values(by='Date')

print(var)

with open('Outside_Temperature.txt','a') as f:

    f.write('2-A(c) the Top Ten hottest times on distinct days, preferably sorted by date order')

    f.write('Date \t\t time \t max_Temp \n')

    for index, row in var.iterrows():

        row_data=[str(index),'\t',str(row[0]),'\t\t\t',str(row[1]),'\n']

        f.writelines(row_data)

    f.close()
```

Explanation: Get the max outside temperature for each day. Merge the Max temp and outside temp for each day and get it into the variable. Group by date and then sort the max occurring temperature according to date and then display the first 10 rows.

Output:

```
             Time  max_temp
Date
10/6/2006   13:10      19.9
11/6/2006   11:20      22.4
12/6/2006   15:00      19.4
15/06/2006  14:00      21.1
17/06/2006  14:50      18.9
28/06/2006  10:50      21.4
3/6/2006    15:00      19.6
6/6/2006    14:20      23.2
7/6/2006    13:00      19.6
8/6/2006    16:50      20.7
```

#Task 2-B: Using the ‘Hi Temperature’ values produce a “Hi_Temperature.txt” file containing all of the Dates and Times where the “Hi Temperature” was within +/- 1 degree of 22.3 or the “Low Temperature” was within +/- 0.2 degree higher or lower of 10.3 over the first 9 days of June required or high and low temperatures of june with date and time

Answer:

```
a = df_data[['Date','Time','Hi Temperature','Low Temperature']]

a_jun = a[pd.to_datetime(a['Date'],format='%d/%m/%Y').dt.month == 6]

f9_days = a_jun['Date'].unique()[:9]

#print (f9_days)

a_jun = a_jun.loc[a_jun['Date'].isin(f9_days)]

#print(a_jun)

a_jun_condition = a_jun.loc[((a_jun['Hi Temperature']>= 21.3) & (a_jun['Hi Temperature']<= 23.3))]

print(a_jun_condition)

with open('Hi Temperature.txt','w') as f:

    f.write('Task 2 - B')

    f.write('Date \t\t\t Time \t Hi temp \t low temp \n')

    for index, row in a_jun_condition.iterrows():

        row_data=[str(row[0]),'\t\t',str(row[1]),'\t\t',str(row[2]),'\t\t\t',str(row[3]),'\n']

        f.writelines(row_data)

    f.close()
```

Explanation: Get the data for columns Date, Time, and Outside Temperature for the month of June and retrieve it for the first 9 days. Then print the outside temperature with >=21.3 and <=23.3.

Output:

```
                Date   Time  Hi Temperature  Low Temperature
Date
2006-06-06  6/6/2006  12:30            21.4             21.2
2006-06-06  6/6/2006  12:40            21.6             21.4
2006-06-06  6/6/2006  12:50            21.8             21.6
2006-06-06  6/6/2006  13:00            22.1             21.8
2006-06-06  6/6/2006  13:10            22.4             22.1
2006-06-06  6/6/2006  13:20            22.7             22.4
2006-06-06  6/6/2006  13:30            22.8             22.7
```

```
2006-06-06  6/6/2006  13:40            23.0            22.8
2006-06-06  6/6/2006  13:50            23.0            22.9
2006-06-06  6/6/2006  14:00            23.1            23.0
2006-06-06  6/6/2006  14:10            23.2            23.1
2006-06-06  6/6/2006  14:20            23.2            23.1
2006-06-06  6/6/2006  14:30            23.2            23.1
2006-06-06  6/6/2006  14:40            23.1            22.9
2006-06-06  6/6/2006  14:50            22.9            22.9
2006-06-06  6/6/2006  15:00            22.9            22.9
2006-06-06  6/6/2006  15:10            22.9            22.7
2006-06-06  6/6/2006  15:20            22.7            22.7
2006-06-06  6/6/2006  15:30            22.7            22.7
2006-06-06  6/6/2006  15:40            22.7            22.6
2006-06-06  6/6/2006  15:50            22.6            22.5
2006-06-06  6/6/2006  16:00            22.5            22.4
2006-06-06  6/6/2006  16:10            22.4            22.3
2006-06-06  6/6/2006  16:20            22.3            22.2
2006-06-06  6/6/2006  16:30            22.3            22.2
2006-06-06  6/6/2006  16:40            22.3            22.2
2006-06-06  6/6/2006  16:50            22.2            22.0
2006-06-06  6/6/2006  17:00            22.0            21.8
2006-06-06  6/6/2006  17:10            21.8            21.8
2006-06-06  6/6/2006  17:20            21.8            21.8
2006-06-06  6/6/2006  17:30            21.8            21.7
2006-06-06  6/6/2006  17:40            21.7            21.6
2006-06-06  6/6/2006  17:50            21.7            21.6
2006-06-06  6/6/2006  18:00            21.9            21.7
2006-06-06  6/6/2006  18:10            21.9            21.7
2006-06-06  6/6/2006  18:20            21.7            21.4
2006-06-06  6/6/2006  18:30            21.4            21.1
```

# Task 3-a: Visulalize the temperature for each month

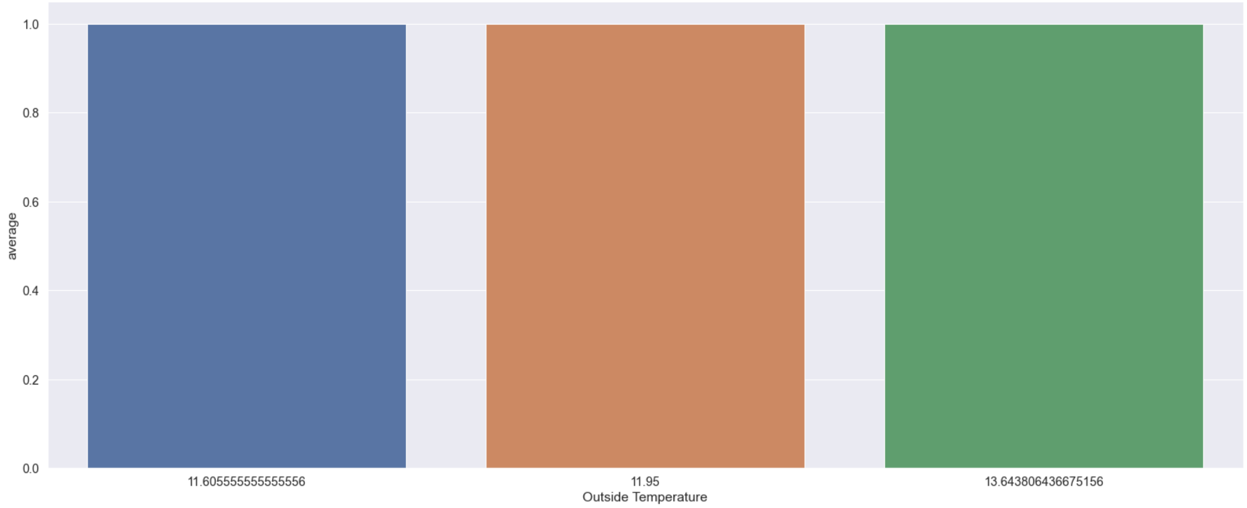
Answer:

```
sns.set_style('whitegrid')

sns.set(font_scale = 1.3)

plt.figure(figsize=(25,10))

dt_month= df_data

dt_month.index= pd.to_datetime(dt_month['Date'],format='%d/%m/%Y')

dt_month=dt_month.groupby(dt_month.index.month).mean()

print(dt_month['Outside Temperature'])

plot=sns.countplot(x='Outside Temperature', data=dt_month)

plot.set(xlabel='Outside Temperature',ylabel='average')
```

Explanation: Get the month data from the dataset by groupby function and then plot the bars using the countplot() function.

Output:

```
Date
5    11.950000
6    13.643806
7    11.605556
```



# Task 3-b: Display the time period on a bar plot which has highest temperature for the first 5 days of every month

Answer:

```
a = df_data[['Date','Time','Outside Temperature']]

a_jun = a[pd.to_datetime(a['Date'],format='%d/%m/%Y').dt.month == 6]

a_jul = a[pd.to_datetime(a['Date'],format='%d/%m/%Y').dt.month == 7]

f9_jundays= a_jun['Date'].unique()[:5]

f9_juldays = a_jul['Date'].unique()[:5]

#print(f9_jundays)

#print(f9_juldays)

a_jun = a_jun.loc[a_jun['Date'].isin(f9_jundays)]
```

```
a_jul = a_jul.loc[a_jul['Date'].isin(f9_juldays)]

a_jun.groupby('Date')

print(a_jun)

print(a_jul)

var['Time'] = pd.to_datetime(var['Time'], format ='%H:%M')

var.index = pd.to_datetime(var['Date'], format ='%d/%m/%Y')
```

Explanation: Get the data for columns Date, Time, and Outside Temperature for the month of June and July. Then gate the first 5 days' data using group by function.

Output:

```
         Date   Time  Outside Temperature
90   1/6/2006   0:00                  9.6
91   1/6/2006   0:10                  9.5
92   1/6/2006   0:20                  9.5
93   1/6/2006   0:30                  9.5
94   1/6/2006   0:40                  9.5
..        ...    ...                  ...
805  5/6/2006  23:10                 10.9
806  5/6/2006  23:20                 10.8
807  5/6/2006  23:30                 10.7
808  5/6/2006  23:40                 10.6
809  5/6/2006  23:50                 10.5

[720 rows x 3 columns]
          Date  Time  Outside Temperature
4409  1/7/2006  0:00                 11.6
4410  1/7/2006  0:10                 11.4
4411  1/7/2006  0:20                 11.4
4412  1/7/2006  0:30                 11.3
4413  1/7/2006  0:40                 11.2
4414  1/7/2006  0:50                 11.0
4415  1/7/2006  1:00                 10.8
4416  1/7/2006  1:10                 10.6
4417  1/7/2006  1:20                 10.4
4418  1/7/2006  1:30                 10.3
4419  1/7/2006  1:40                 10.2
4420  1/7/2006  1:50                 10.1
4421  1/7/2006  2:00                 10.1
4422  1/7/2006  2:10                 10.1
4423  1/7/2006  2:20                 10.2
4424  1/7/2006  2:30                 10.2
4425  1/7/2006  2:40                 10.1
4426  1/7/2006  2:50                 10.1
4427  1/7/2006  3:00                 10.1
4428  1/7/2006  3:10                 10.1
4429  1/7/2006  3:20                  9.9
```

```
4430  1/7/2006  3:30                   9.8
4431  1/7/2006  3:40                   9.6
4432  1/7/2006  3:50                   9.6
4433  1/7/2006  4:00                   9.7
4434  1/7/2006  4:10                   9.8
4435  1/7/2006  4:20                   9.9
4436  1/7/2006  4:30                  10.1
4437  1/7/2006  4:40                  10.3
4438  1/7/2006  4:50                  10.5
4439  1/7/2006  5:00                  10.7
4440  1/7/2006  5:10                  10.9
4441  1/7/2006  5:20                  11.0
4442  1/7/2006  5:30                  11.3
4443  1/7/2006  5:40                  11.6
4444  1/7/2006  5:50                  11.7
4445  1/7/2006  6:00                  11.9
4446  1/7/2006  6:10                  12.2
4447  1/7/2006  6:20                  12.6
4448  1/7/2006  6:30                  12.8
4449  1/7/2006  6:40                  12.8
4450  1/7/2006  6:50                  13.0
4451  1/7/2006  7:00                  13.3
4452  1/7/2006  7:10                  13.6
4453  1/7/2006  7:20                  13.8
4454  1/7/2006  7:30                  13.9
4455  1/7/2006  7:40                  14.1
4456  1/7/2006  7:50                  14.3
4457  1/7/2006  8:00                  14.4
4458  1/7/2006  8:10                  14.6
4459  1/7/2006  8:20                  14.8
4460  1/7/2006  8:30                  15.3
4461  1/7/2006  8:40                  15.6
4462  1/7/2006  8:50                  16.0
```

```
#Task4 plot the daily temeperature change

avg_day= df_data.groupby('Date')['Outside Temperature'].mean().reset_index(name='Avg temp')

fig, ax1 = plt.subplots(figsize=(25, 10))

plot=sns.barplot(x="Date",y="Avg temp", data= avg_day)

plot.set(xlabel='days number in year', ylabel='Average outside temperature')

sns.despine(fig)

###plot avg temp for outside temperature (joint plot) ######


sns.set_theme(style="whitegrid")
```

```
s = sns.jointplot(x="Date",y="Avg temp", data= avg_day,xlim=(20,30), ylim=(0,15),color="y", height=7)
```

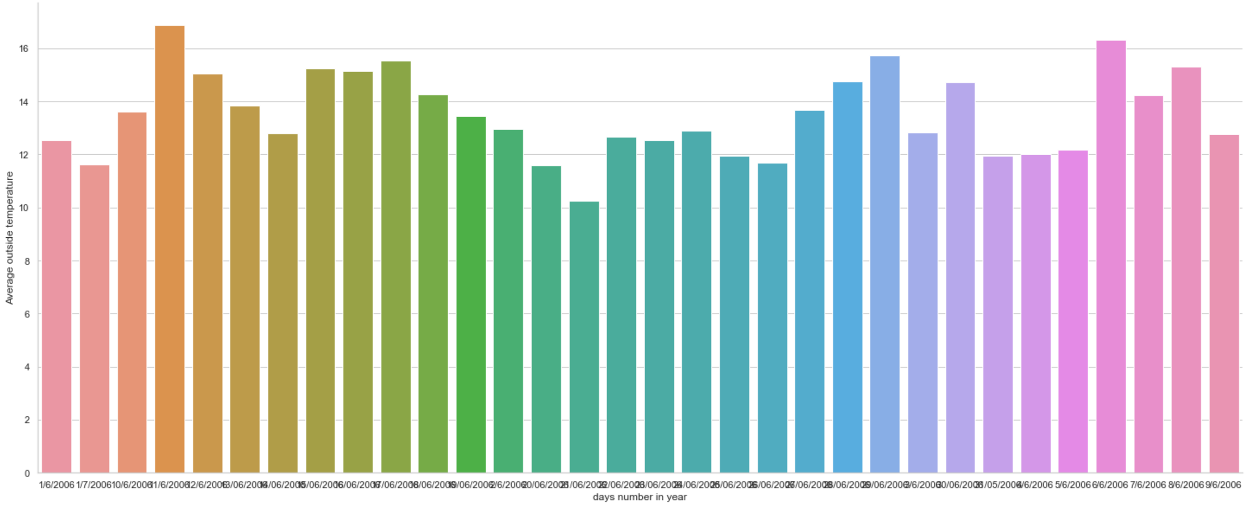
```
##########################end code for Task 4
```

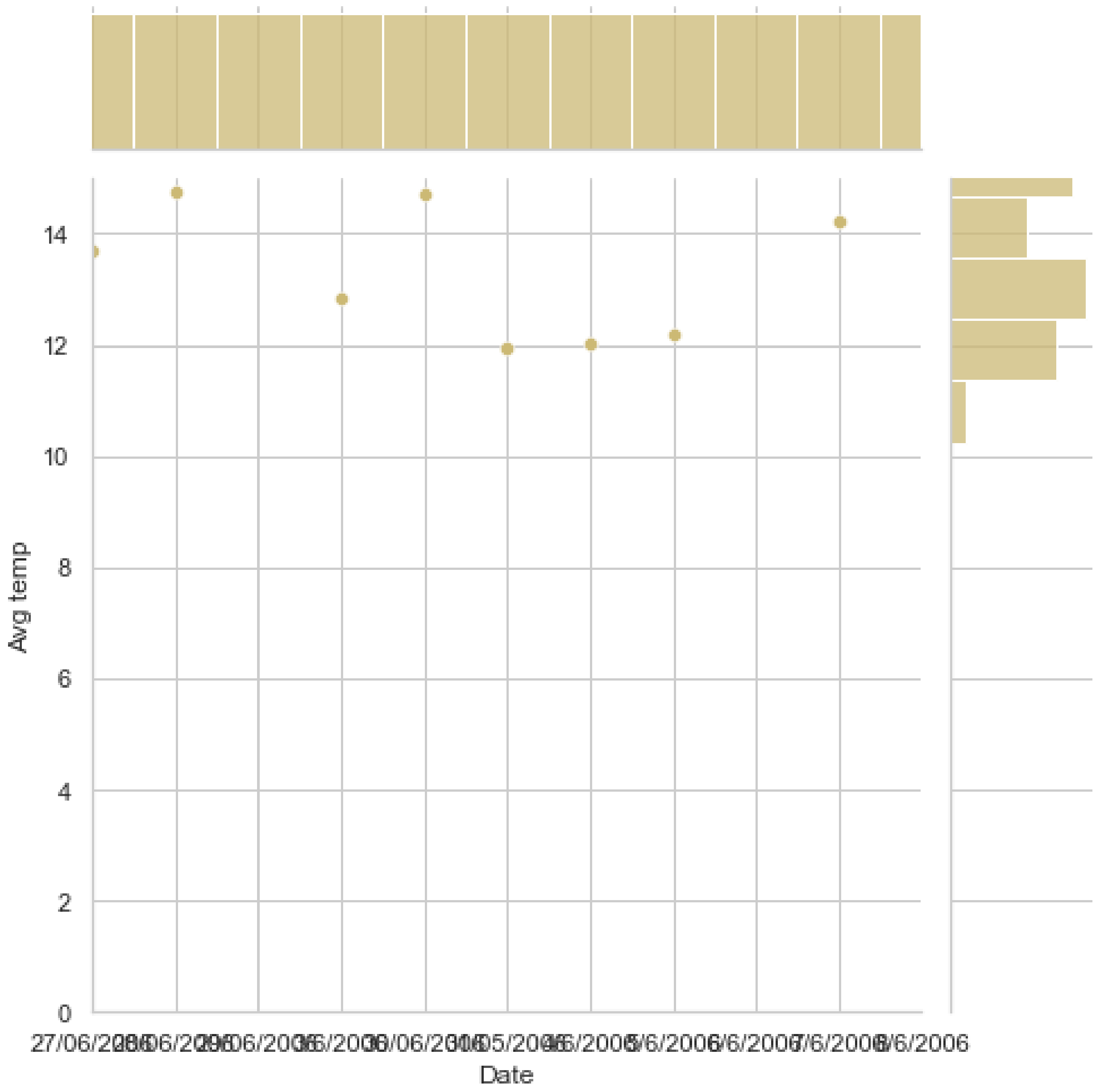
Explanation: X-axis: days in a year,

Y-axis: Avg Outside temperature

Here we are ploting the bar plot b/w x and y axis were it tells about the avg temperature of each day from the data set to each day.

Were groupby of date is done and avg temp is calculated based on Outside Temperature





X-axis is Date

Y-axis is avg temp

The joint plot is plotted b/w x and y-axis where it determines about the mean temperature based on “Outside temperature”

Were each point is the measured value of day to mean Temperature.