If ur just trying to complete the assessment... Remove all the codes and just paste below code

```
ws_std = 13.06
p_range = 12.20
corr = 0.76
dew_month = 1
max_gust_month = 2
max_gust_value = 34.50
avg_temp = 45.33
temp_range = 44.80
max_p_range_day = '2018-03-23'
num_days_std = 2092
median_b_days = 534
```

Now press ctrl + s, then go pack to hackerrank and click on test. Once you get all 11 test cases passed, click on submit

Below Codes are only for study purpose. (Do not paste the code in assessment tab)

```
import pandas as pd
import numpy as np
import seaborn as sns
%matplotlib inline
```

```
!gdown --id 1EKcA7vVG8ea9KSkjq98RpwPMpASPJuD8
df = pd.read_csv('data.csv')
```

```
/usr/local/lib/python3.7/dist-packages/gdown/cli.py:131: FutureWarning: Option `--id` was deprecated in version 4.3.1 and will be
  category=FutureWarning,
Downloading...
From: https://drive.google.com/uc?id=1EKcA7vVG8ea9KSkjq98RpwPMpASPJuD8
To: /content/data.csv
100% 333k/333k [00:00<00:00, 2.91MB/s]
```

What is the standard deviation of maximum windspeed across all the days

```
ws_std = round(df['Maximum windspeed (mph)'].std(), 2)
ws_std
```

```
13.06
```

What is the difference between 50th percentile and 75th percentile of average temperature

```
p_range = round(df['Average temperature (°F)'].quantile(0.75)-df['Average temperature (°F)'].quantile(0.5), 2)
p_range
```

```
12.2
```

What is the pearson correlation between average dew point and average temperature

```
corr = round(df['Average dewpoint (°F)'].corr(df['Average temperature (°F)'], method='pearson'), 2)
corr
```

```
0.76
```

Out of all the available records which month has the lowest average humidity.

```
 - Assign your answer as month index, for example if its July index is 7
```

```
df['datetime'] = pd.to_datetime(df['Day'], format='%d/%m/%Y')
df['Month'] = df['datetime'].dt.month
mask = df['Average humidity (%)']==df['Average humidity (%)'].min()
dew = df.loc[mask]
dew_month = dew['Month'][dew.index[0]]
dew_month
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-1-fc42236549ce> in <module>()
----> 1 df['datetime'] = pd.to_datetime(df['Day'], format='%d/%m/%Y')
      2 df['Month'] = df['datetime'].dt.month
      3 mask = df['Average humidity (%)']==df['Average humidity (%)'].min()
      4 dew = df.loc[mask]
      5 dew_month = dew['Month'][dew.index[0]]

NameError: name 'pd' is not defined
```

Which month has the highest median for maximum_gust_speed out of all the available records. Also find the repective value

- hint: group by month

```
df['Month1'] = df['datetime'].dt.month
median1 = df.groupby(df['Month']).median()
max_gust_value = median1['Maximum gust speed (mph)'].max()
mask =  median1['Maximum gust speed (mph)']==median1['Maximum gust speed (mph)'].max()
max_gust_month = median1.loc[mask]
max_gust_month = max_gust_month['Month1']
max_gust_month = int(max_gust_month[2])
print(max_gust_value)
print(max_gust_month)
```

```
34.5
2
```

Determine the average temperature between the months of March 2010 to May 2012 (including both the months)

```
mask = (df['datetime']>='2010/03/01') & (df['datetime']<'2012/06/01')
mask_data = df.loc[mask]
avg_temp = round(mask_data['Average temperature (°F)'].mean(), 2)
avg_temp
```

```
45.33
```

Find the range of averange temperature on Dec 2010

```
mask = (df['datetime']>='2010/12/01') & (df['datetime']<'2011/01/01')
mask_data = df.loc[mask]
temp_min = mask_data['Average temperature (°F)'].min()
temp_max = mask_data['Average temperature (°F)'].max()
temp_range = round(temp_max-temp_min, 2)
temp_range
```

```
44.8
```

Out of all available records which day has the highest difference between maximum_pressure and minimum_pressure

- assign the date in string format as 'yyyy-mm-dd'. Make sure you enclose it with single quote

```
df['pressure_diff'] = df['Maximum pressure ']-df['Minimum pressure ']
max_p_range_day = df[df['pressure_diff']==df['pressure_diff'].max()].datetime
max_p_range_day = max_p_range_day.to_string(index=False)
max_p_range_day
```

```
'2018-03-23'
```

How many days falls under median (i.e equal to median value) of barrometer reading.

```
med = df[df['Average barometer (in)']==df['Average barometer (in)'].median()]
median_b_days = len(med)
median_b_days
```

```
534
```

Out of all the available records how many days are within one standard deviation of average temperaturem

```
std = df['Average temperature (°F)'].std()
mean = df['Average temperature (°F)'].mean()
no_days = df[(df['Average temperature (°F)'] >= mean-std) & (df['Average temperature (°F)'] <= mean + std)]
num_days_std =  len(no_days)
num_days_std
```

```
2092
```

```
#1 understanding the data
df.head()
```

	Day	Average temperature (°F)	Average humidity (%)	Average dewpoint (°F)	Average barometer (in)	Average windspeed (mph)	Average gustspeed (mph)	Average direction (°deg)	Rainfall for month (in)	Rai for
0	1/01/2009	37.8	35	12.7	29.7	26.4	36.8	274	0.0	
1	2/01/2009	43.2	32	14.7	29.5	12.8	18.0	240	0.0	
2	3/01/2009	25.7	60	12.7	29.7	8.3	12.2	290	0.0	
3	4/01/2009	9.3	67	0.1	30.4	2.9	4.5	47	0.0	
4	5/01/2009	23.5	30	-5.3	29.9	16.7	23.1	265	0.0	

```
df.tail()
```

	Average	Average	Average	Average	Average	Average	Average	Rainfall for

```
df.shape
```

```
(3280, 20)
```

```
df.describe()
```

	Average temperature (°F)	Average humidity (%)	Average dewpoint (°F)	Average barometer (in)	Average windspeed (mph)	Average gustspeed (mph)	Average direction (°deg)	Rain for m
count	3280.000000	3280.000000	3280.000000	3280.000000	3280.000000	3280.000000	3280.000000	3280.00
mean	45.162896	48.511585	23.350427	29.891616	5.759909	10.253537	212.144207	0.47
std	15.188165	17.495676	14.649178	0.252187	4.093984	15.226675	101.055441	0.63
min	-12.100000	9.000000	-22.200000	28.200000	0.000000	0.000000	0.000000	0.00
25%	34.500000	36.000000	12.300000	29.700000	2.600000	4.400000	109.750000	0.05
50%	45.900000	47.000000	22.900000	29.900000	4.600000	7.100000	253.000000	0.23
75%	58.100000	60.000000	35.700000	30.000000	8.000000	12.100000	282.000000	0.69
max	76.300000	92.000000	55.100000	31.000000	26.400000	240.400000	360.000000	4.48

```
df.columns
```

```
Index(['Day', 'Average temperature (°F)', 'Average humidity (%)',
       'Average dewpoint (°F)', 'Average barometer (in)',
       'Average windspeed (mph)', 'Average gustspeed (mph)',
       'Average direction (°deg)', 'Rainfall for month (in)',
       'Rainfall for year (in)', 'Maximum rain per minute ',
       'Maximum temperature (°F)', 'Minimum temperature (°F)',
       'Maximum humidity (%)', 'Minimum humidity (%)', 'Maximum pressure ',
       'Minimum pressure ', 'Maximum windspeed (mph)',
       'Maximum gust speed (mph)', 'Maximum heat index (°F)'],
      dtype='object')
```

```
df.nunique()
```

```
17
```

```
df['Day'].unique()
```

```
array(['1/01/2009', '2/01/2009', '3/01/2009', ..., '26/10/2018',
       '27/10/2018', '28/10/2018'], dtype=object)
```

```
#Cleaning the data
df.isnull().sum()
```

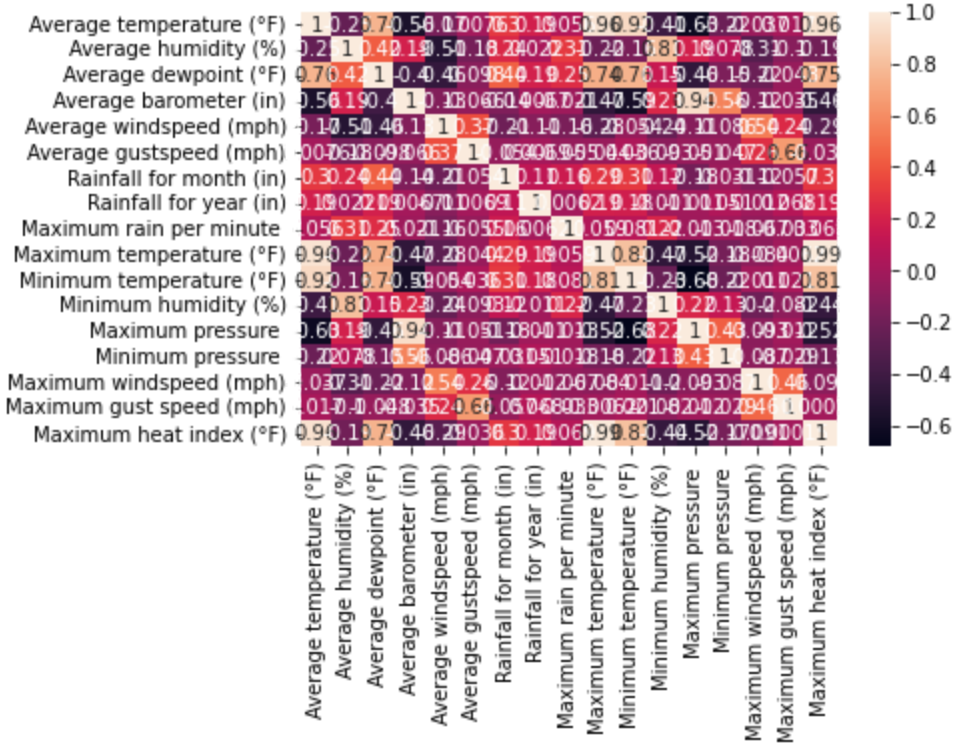
```
Day                          0
Average temperature (°F)     0
Average humidity (%)         0
Average dewpoint (°F)        0
Average barometer (in)       0
Average windspeed (mph)      0
Average gustspeed (mph)      0
Average direction (°deg)     0
Rainfall for month (in)      0
Rainfall for year (in)       0
Maximum rain per minute      0
Maximum temperature (°F)     0
Minimum temperature (°F)     0
Maximum humidity (%)         0
Minimum humidity (%)         0
Maximum pressure             0
Minimum pressure             0
Maximum windspeed (mph)      0
Maximum gust speed (mph)     0
Maximum heat index (°F)      0
dtype: int64
```

```
#DRopping the columns, axis =1 means columns
st= df.drop(['Average direction (°deg)','Maximum humidity (%)'],axis=1)
st.head()
```

	Day	Average temperature (°F)	Average humidity (%)	Average dewpoint (°F)	Average barometer (in)	Average windspeed (mph)	Average gustspeed (mph)	Rainfall for month (in)	Rainfall for year (in)	Maxi r mir
0	1/01/2009	37.8	35	12.7	29.7	26.4	36.8	0.0	0.0	
1	2/01/2009	43.2	32	14.7	29.5	12.8	18.0	0.0	0.0	

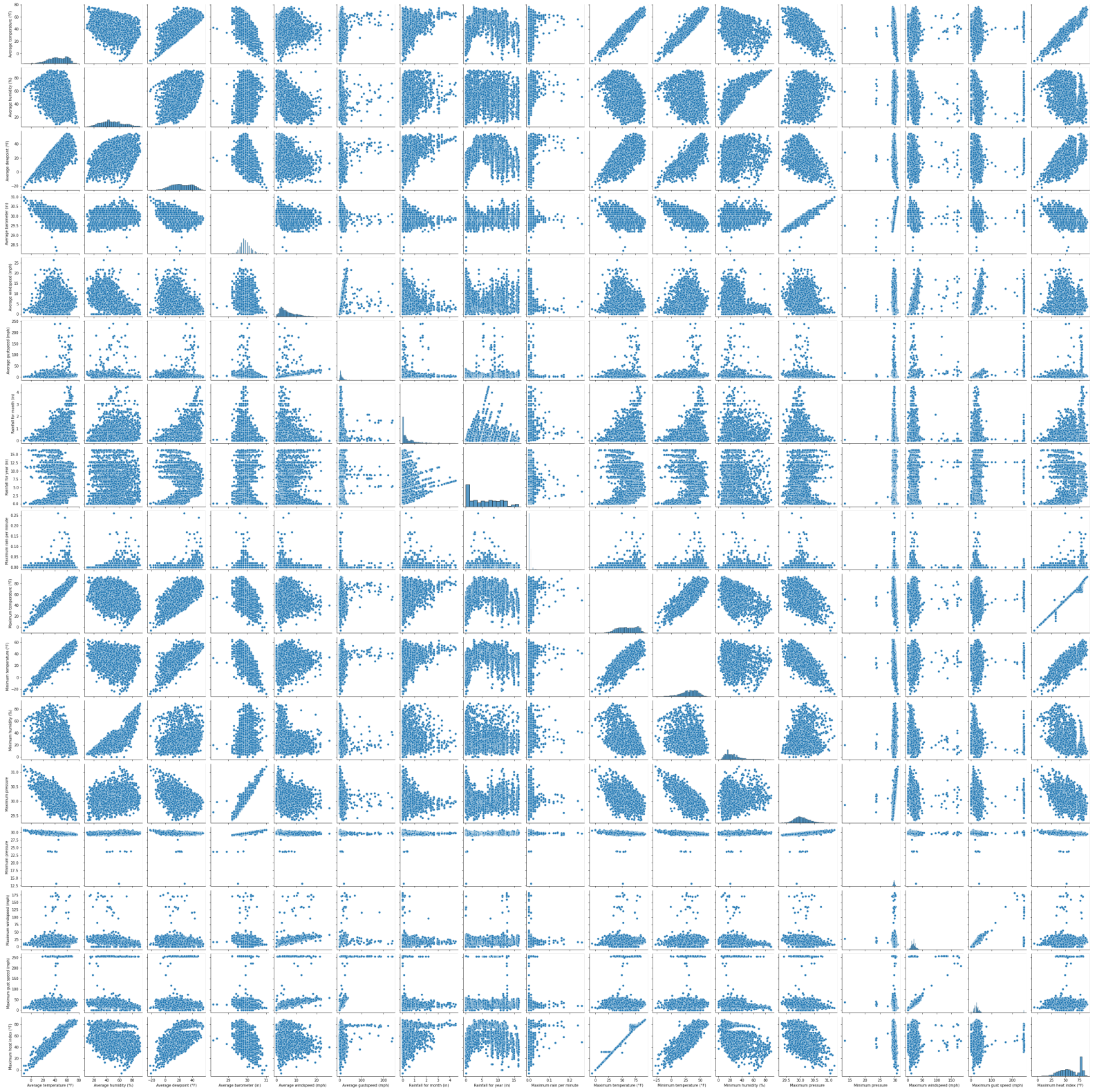
```
#relationship analysis
co = st.corr()
sns.heatmap(co, annot= True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fc62182a850>
```




```
sns.pairplot(st)
```

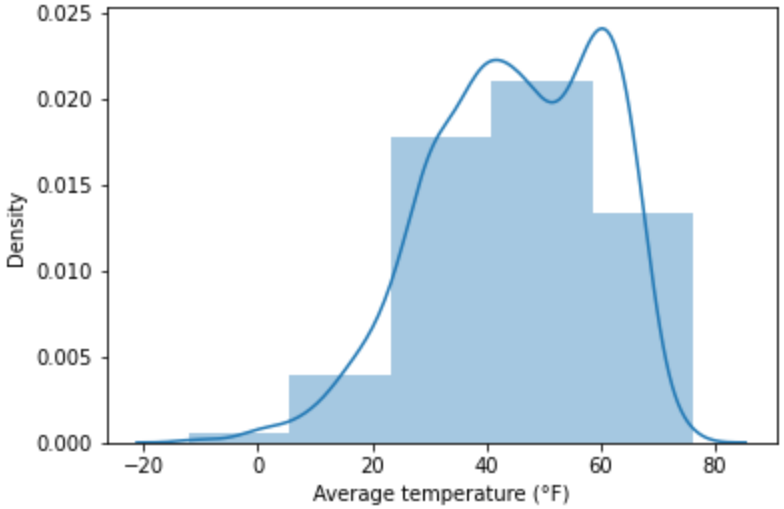
<seaborn.axisgrid.PairGrid at 0x7fc61dd41dd0>

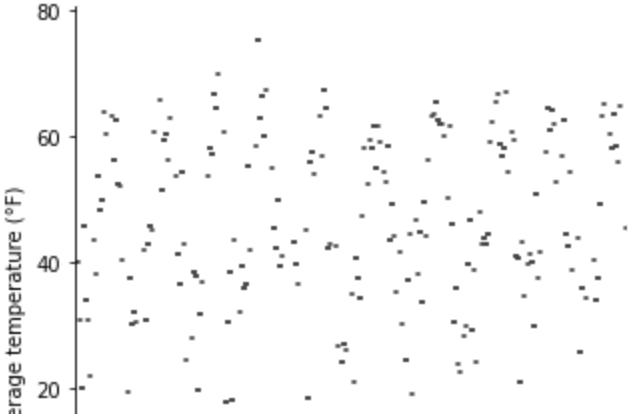


```
sns.relplot(x='',y='', hu1e='',data=st) #scatterplot


sns.distplot(st['Average temperature (°F)'],bins=5) #histogram
sns.catplot(x='Day', kind='box',data=st) #boxplot
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning: `distplot` is a de
  warnings.warn(msg, FutureWarning)
<seaborn.axisgrid.FacetGrid at 0x7f8da0009450>
```






```
st['datetime'] = pd.to_datetime(st['Day'], format='%d/%m/%Y')
st['Year'] = st['datetime'].dt.year
year = st.groupby(st['Year'])
year.head()
year[['Day','Year','Average temperature (°F)']].head()
```

	Day	Year	Average temperature (°F)
0	1/01/2009	2009	37.8
1	2/01/2009	2009	43.2
2	3/01/2009	2009	25.7
3	4/01/2009	2009	9.3
4	5/01/2009	2009	23.5
331	1/01/2010	2010	32.1
332	2/01/2010	2010	32.1
333	3/01/2010	2010	23.1
334	4/01/2010	2010	25.7
335	5/01/2010	2010	34.3
692	1/01/2011	2011	6.6
693	2/01/2011	2011	20.5
694	3/01/2011	2011	24.5
695	4/01/2011	2011	26.9
696	5/01/2011	2011	27.8
942	1/01/2012	2012	32.4
943	2/01/2012	2012	36.6
944	3/01/2012	2012	43.7
945	4/01/2012	2012	45.1
946	5/01/2012	2012	43.5
1279	1/01/2013	2013	13.4
1280	2/01/2013	2013	16.1
1281	3/01/2013	2013	13.2
1282	4/01/2013	2013	19.1
1283	5/01/2013	2013	20.2

1599	1/01/2014	2014	26.9
1600	2/01/2014	2014	35.2
1601	3/01/2014	2014	39.4
1602	4/01/2014	2014	12.0
1603	5/01/2014	2014	3.3
1956	1/01/2015	2015	8.1
1957	2/01/2015	2015	20.8
1958	3/01/2015	2015	24.2
1959	4/01/2015	2015	26.4
1960	5/01/2015	2015	38.9
2307	1/01/2016	2016	7.5
2308	2/01/2016	2016	15.7
2309	3/01/2016	2016	31.0
2310	4/01/2016	2016	25.0

```
sns.boxplot(x='Year',y='Average temperature (°F)',data=st)
```