8.write python propgram using pandas

a) to handle missing values in datasets

b) to remove duplicate values from the dataset

c) to count the NaN values in the dataset

In [20]:

```
# import the dataset
import pandas as pd
import numpy as np
df=pd.read_csv('weather.csv',parse_dates=['day'])
print(df)
```

```
         day  temprature  windspeed   event
0 2021-01-01        32.0        6.0    Rain
1 2021-01-04         NaN        9.0   Sunny
2 2021-01-05        28.0        NaN    Snow
3 2021-01-06         NaN        7.0     NaN
4 2021-01-07        32.0        NaN    Rain
5 2021-01-08         NaN        NaN   Sunny
6 2021-01-09         NaN        NaN     NaN
7 2021-01-10        34.0        8.0  Cloudy
8 2021-01-11        40.0       12.0   Sunny
```

In [25]:

```
df.set_index('day',inplace=True)
print(df)
```

```
            temprature  windspeed   event
day
2021-01-01        32.0        6.0    Rain
2021-01-04         NaN        9.0   Sunny
2021-01-05        28.0        NaN    Snow
2021-01-06         NaN        7.0     NaN
2021-01-07        32.0        NaN    Rain
2021-01-08         NaN        NaN   Sunny
2021-01-09         NaN        NaN     NaN
2021-01-10        34.0        8.0  Cloudy
2021-01-11        40.0       12.0   Sunny
```

In [26]:

```
#count NaN values in the datset
print(df.isnull().sum())
print('total :',df.isnull().sum().sum())
```

```
temprature    4
windspeed     4
event         2
dtype: int64
total : 10
```

In [27]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 9 entries, 2021-01-01 to 2021-01-11
Data columns (total 3 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   temprature  5 non-null      float64
 1   windspeed   5 non-null      float64
 2   event       7 non-null      object
dtypes: float64(2), object(1)
memory usage: 288.0+ bytes
```

In [28]:

```
df.isnull() # finds null that is NaN values in dataset
```

Out[28]:

	temprature	windspeed	event
day			
2021-01-01	False	False	False
2021-01-04	True	False	False
2021-01-05	False	True	False
2021-01-06	True	False	True
2021-01-07	False	True	False
2021-01-08	True	True	False
2021-01-09	True	True	True
2021-01-10	False	False	False
2021-01-11	False	False	False

In [29]:

```
#handle missing datasets
#by dropping the na rows
df1=df.dropna()
df1
```

Out[29]:

	temprature	windspeed	event
day			
2021-01-01	32.0	6.0	Rain
2021-01-10	34.0	8.0	Cloudy
2021-01-11	40.0	12.0	Sunny

In [30]:

```
new_df2=df.dropna(how='all')
new_df2
```

Out[30]:

	temprature	windspeed	event
day			
2021-01-01	32.0	6.0	Rain
2021-01-04	NaN	9.0	Sunny
2021-01-05	28.0	NaN	Snow
2021-01-06	NaN	7.0	NaN
2021-01-07	32.0	NaN	Rain

	temprature	windspeed	event
day			
2021-01-08	NaN	NaN	Sunny
2021-01-10	34.0	8.0	Cloudy
2021-01-11	40.0	12.0	Sunny

In [31]:

```
new_df3=df.dropna(how='any')
new_df3
```

Out[31]:

	temprature	windspeed	event
day			
2021-01-01	32.0	6.0	Rain
2021-01-10	34.0	8.0	Cloudy
2021-01-11	40.0	12.0	Sunny

In [32]:

```
print(df)
new_df2=df.dropna(how='all',thresh=2)
new_df2
```

```
            temprature  windspeed   event
day
2021-01-01        32.0        6.0    Rain
2021-01-04         NaN        9.0   Sunny
2021-01-05        28.0        NaN    Snow
2021-01-06         NaN        7.0     NaN
2021-01-07        32.0        NaN    Rain
2021-01-08         NaN        NaN   Sunny
2021-01-09         NaN        NaN     NaN
2021-01-10        34.0        8.0  Cloudy
2021-01-11        40.0       12.0   Sunny
```

Out[32]:

	temprature	windspeed	event
day			
2021-01-01	32.0	6.0	Rain
2021-01-04	NaN	9.0	Sunny
2021-01-05	28.0	NaN	Snow
2021-01-07	32.0	NaN	Rain
2021-01-10	34.0	8.0	Cloudy
2021-01-11	40.0	12.0	Sunny

In [33]:

```
new_df3=df.fillna('0')
new_df3
```

Out[33]:

	temprature	windspeed	event

day	temprature	windspeed	event
day			
2021-01-01	32.0	6.0	Rain
2021-01-04	0	9.0	Sunny
2021-01-05	28.0	0	Snow
2021-01-06	0	7.0	0
2021-01-07	32.0	0	Rain
2021-01-08	0	0	Sunny
2021-01-09	0	0	0
2021-01-10	34.0	8.0	Cloudy
2021-01-11	40.0	12.0	Sunny

In [34]:

```
df1=df.fillna(method='bfill')
df1
```

Out[34]:

	temprature	windspeed	event
day			
2021-01-01	32.0	6.0	Rain
2021-01-04	28.0	9.0	Sunny
2021-01-05	28.0	7.0	Snow
2021-01-06	32.0	7.0	Rain
2021-01-07	32.0	8.0	Rain
2021-01-08	34.0	8.0	Sunny
2021-01-09	34.0	8.0	Cloudy
2021-01-10	34.0	8.0	Cloudy
2021-01-11	40.0	12.0	Sunny

In [35]:

```
df1=df.fillna(method='bfill',limit=1)
df1
```

Out[35]:

	temprature	windspeed	event
day			
2021-01-01	32.0	6.0	Rain
2021-01-04	28.0	9.0	Sunny
2021-01-05	28.0	7.0	Snow
2021-01-06	32.0	7.0	Rain

	temprature	windspeed	event
day			
2021-01-07	32.0	NaN	Rain
2021-01-08	NaN	NaN	Sunny
2021-01-09	34.0	8.0	Cloudy
2021-01-10	34.0	8.0	Cloudy
2021-01-11	40.0	12.0	Sunny

In [36]:

```
df2=df.fillna(method='ffill',limit=1)
df1
```

Out[36]:

	temprature	windspeed	event
day			
2021-01-01	32.0	6.0	Rain
2021-01-04	28.0	9.0	Sunny
2021-01-05	28.0	7.0	Snow
2021-01-06	32.0	7.0	Rain
2021-01-07	32.0	NaN	Rain
2021-01-08	NaN	NaN	Sunny
2021-01-09	34.0	8.0	Cloudy
2021-01-10	34.0	8.0	Cloudy
2021-01-11	40.0	12.0	Sunny

In [37]:

```
df.describe()
```

Out[37]:

	temprature	windspeed
count	5.00000	5.000000
mean	33.20000	8.400000
std	4.38178	2.302173
min	28.00000	6.000000
25%	32.00000	7.000000
50%	32.00000	8.000000
75%	34.00000	9.000000
max	40.00000	12.000000

In [38]:

```
df2=df.fillna(df.mean())
```

```
df2
```

```
<ipython-input-38-12b346d86ac1>:1: FutureWarning: Dropping of nuisance columns in DataFr
ame reductions (with 'numeric_only=None') is deprecated; in a future version this will r
aise TypeError.  Select only valid columns before calling the reduction.
  df2=df.fillna(df.mean())
```

Out[38]:

	temprature	windspeed	event
day			
2021-01-01	32.0	6.0	Rain
2021-01-04	33.2	9.0	Sunny
2021-01-05	28.0	8.4	Snow
2021-01-06	33.2	7.0	NaN
2021-01-07	32.0	8.4	Rain
2021-01-08	33.2	8.4	Sunny
2021-01-09	33.2	8.4	NaN
2021-01-10	34.0	8.0	Cloudy
2021-01-11	40.0	12.0	Sunny

In [52]:

```
print(df['temprature'].fillna(df['temprature'].mode()[0]))
print(df['event'].fillna(df['event'].mode()[0]))
```

```
day
2021-01-01    32.0
2021-01-04    32.0
2021-01-05    28.0
2021-01-06    32.0
2021-01-07    32.0
2021-01-08    32.0
2021-01-09    32.0
2021-01-10    34.0
2021-01-11    40.0
Name: temprature, dtype: float64
day
2021-01-01      Rain
2021-01-04     Sunny
2021-01-05      Snow
2021-01-06     Sunny
2021-01-07      Rain
2021-01-08     Sunny
2021-01-09     Sunny
2021-01-10    Cloudy
2021-01-11     Sunny
Name: event, dtype: object
```

In [44]:

```
#removing duplicate values
import pandas as pd

d = {'A': [1, 1, 11, 12], 'B': [12, 12, 12, 13], 'C': [1, 1, 11, 12]}

d1 = pd.DataFrame(d)
print('Source DataFrame:\n', d1)
```

```
Source DataFrame:
    A   B   C
0   1  12   1
1   1  12   1
2  11  12  11
3  12  13  12
```

In [47]:

```python
dup=d1.duplicated() # to find duplicate objects
print(dup)
```

```
0    False
1     True
2    False
3    False
dtype: bool
```

In [57]:

```python
dup=d1[d1.duplicated()] # duplicate row
print(dup)
```

```
   A   B  C
1  1  12  1
```

In [58]:

```python
dup_first=d1.duplicated(keep='first')
print(dup_first)
```

```
0    False
1     True
2    False
3    False
dtype: bool
```

In [56]:

```python
dup_last=d1.duplicated(keep='last')
print(dup_last)
```

```
0     True
1    False
2    False
3    False
dtype: bool
```

In [60]:

```python
# Select duplicate rows except last occurrence based on all columns
dup_last=d1[d1.duplicated(keep='last')]
print(dup_last)
```

```
   A   B  C
0  1  12  1
```

In [61]:

```python
print(d1)
print("******************")
dup_col=d1.duplicated(subset='C')
print(dup_col)
```

```
    A   B   C
0   1  12   1
1   1  12   1
2  11  12  11
3  12  13  12
******************
```

```
0    False
1     True
2    False
3    False
dtype: bool
```

In [63]:

```
#Drop Duplicate Rows Keeping the First One
# This is the default behavior when no arguments are passed.

result_df = d1.drop_duplicates()
print('Result DataFrame:\n', result_df)
```

```
Result DataFrame:
     A   B   C
0   1  12   1
2  11  12  11
3  12  13  12
```

In [64]:

```
#Drop Duplicates and Keep Last Row
print('Source DataFrame:\n', d1)
print("************")
result_df = d1.drop_duplicates(keep='last')
print('Result DataFrame:\n', result_df)
```

```
Source DataFrame:
     A   B   C
0   1  12   1
1   1  12   1
2  11  12  11
3  12  13  12
************
Result DataFrame:
     A   B   C
1   1  12   1
2  11  12  11
3  12  13  12
```

In [65]:

```
#Delete All Duplicate Rows from DataFrame

result_df = d1.drop_duplicates(keep=False)
print('Result DataFrame:\n', result_df)
```

```
Result DataFrame:
     A   B   C
2  11  12  11
3  12  13  12
```