In [1]:
```
import numpy as np
```

In [2]:
```
import pandas as pd
```

Pre-Processing

In [4]:
```
data=pd.read_csv(r"C:\Users\user\Downloads\7_uber.csv")
data
```

Out[4]:

	Unnamed: 0	key	fare_amount	pickup_datetime	pickup_longitude	pickup_
0	24238194	2015-05-07 19:52:06.0000003	7.5	2015-05-07 19:52:06 UTC	-73.999817	40
1	27835199	2009-07-17 20:04:56.0000002	7.7	2009-07-17 20:04:56 UTC	-73.994355	40
2	44984355	2009-08-24 21:45:00.00000061	12.9	2009-08-24 21:45:00 UTC	-74.005043	40
3	25894730	2009-06-26 08:22:21.0000001	5.3	2009-06-26 08:22:21 UTC	-73.976124	40
4	17610152	2014-08-28 17:47:00.000000188	16.0	2014-08-28 17:47:00 UTC	-73.925023	40
...	...	...	...	...	...	...
199995	42598914	2012-10-28 10:49:00.00000053	3.0	2012-10-28 10:49:00 UTC	-73.987042	40
199996	16382965	2014-03-14 01:09:00.0000008	7.5	2014-03-14 01:09:00 UTC	-73.984722	40
199997	27804658	2009-06-29 00:42:00.00000078	30.9	2009-06-29 00:42:00 UTC	-73.986017	40
199998	20259894	2015-05-20 14:56:25.0000004	14.5	2015-05-20 14:56:25 UTC	-73.997124	40
199999	11951496	2010-05-15 04:08:00.00000076	14.1	2010-05-15 04:08:00 UTC	-73.984395	40

200000 rows × 9 columns

In [5]:
```
data.head()
```

Out[5]:

	key	fare_amount	pickup_datetime	pickup_longitude	pickup_latitude	dropoff_longitude	drc
0	2015-05-07)6.0000003	7.5	2015-05-07 19:52:06 UTC	-73.999817	40.738354	-73.999512	
1	2009-07-17 ;6.0000002	7.7	2009-07-17 20:04:56 UTC	-73.994355	40.728225	-73.994710	
2	2009-08-24).00000061	12.9	2009-08-24 21:45:00 UTC	-74.005043	40.740770	-73.962565	
3	2009-06-26 :1.0000001	5.3	2009-06-26 08:22:21 UTC	-73.976124	40.790844	-73.965316	
4	2014-08-28 000000188	16.0	2014-08-28 17:47:00 UTC	-73.925023	40.744085	-73.973082	

In [6]:
```
data.tail()
```

Out[6]:

	Unnamed: 0	key	fare_amount	pickup_datetime	pickup_longitude	pickup_la
199995	42598914	2012-10-28 10:49:00.00000053	3.0	2012-10-28 10:49:00 UTC	-73.987042	40.7
199996	16382965	2014-03-14 01:09:00.0000008	7.5	2014-03-14 01:09:00 UTC	-73.984722	40.7
199997	27804658	2009-06-29 00:42:00.00000078	30.9	2009-06-29 00:42:00 UTC	-73.986017	40.7
199998	20259894	2015-05-20 14:56:25.0000004	14.5	2015-05-20 14:56:25 UTC	-73.997124	40.7
199999	11951496	2010-05-15 04:08:00.00000076	14.1	2010-05-15 04:08:00 UTC	-73.984395	40.7

In [7]:
```
data.describe()
```

Out[7]:

	Unnamed: 0	fare_amount	pickup_longitude	pickup_latitude	dropoff_longitude	dropof
count	2.000000e+05	200000.000000	200000.000000	200000.000000	199999.000000	1999
mean	2.771250e+07	11.359955	-72.527638	39.935885	-72.525292	3
std	1.601382e+07	9.901776	11.437787	7.720539	13.117408	
min	1.000000e+00	-52.000000	-1340.648410	-74.015515	-3356.666300	-8
25%	1.382535e+07	6.000000	-73.992065	40.734796	-73.991407	4
50%	2.774550e+07	8.500000	-73.981823	40.752592	-73.980093	4
75%	4.155530e+07	12.500000	-73.967154	40.767158	-73.963658	4
max	5.542357e+07	499.000000	57.418457	1644.421482	1153.572603	8

In [8]:
```
print(np.shape(data))
```

```
(200000, 9)
```

In [9]:
```
print(np.size(data))
```

```
1800000
```

In [12]:
```
print(data.isna())
```

```
        Unnamed: 0    key  fare_amount  pickup_datetime  pickup_longitude  \
0            False  False        False            False             False
1            False  False        False            False             False
2            False  False        False            False             False
3            False  False        False            False             False
4            False  False        False            False             False
...            ...    ...          ...              ...               ...
199995       False  False        False            False             False
199996       False  False        False            False             False
199997       False  False        False            False             False
199998       False  False        False            False             False
199999       False  False        False            False             False

        pickup_latitude  dropoff_longitude  dropoff_latitude  passenger_count
0                 False              False             False            False
1                 False              False             False            False
2                 False              False             False            False
3                 False              False             False            False
4                 False              False             False            False
...                 ...                ...               ...              ...
199995            False              False             False            False
199996            False              False             False            False
199997            False              False             False            False
199998            False              False             False            False
199999            False              False             False            False

[200000 rows x 9 columns]
```

In [13]:
```
data.fillna(value=0)
```

Out[13]:

	Unnamed: 0	key	fare_amount	pickup_datetime	pickup_longitude	pickup_
0	24238194	2015-05-07 19:52:06.0000003	7.5	2015-05-07 19:52:06 UTC	-73.999817	40
1	27835199	2009-07-17 20:04:56.0000002	7.7	2009-07-17 20:04:56 UTC	-73.994355	40
2	44984355	2009-08-24 21:45:00.00000061	12.9	2009-08-24 21:45:00 UTC	-74.005043	40
3	25894730	2009-06-26 08:22:21.0000001	5.3	2009-06-26 08:22:21 UTC	-73.976124	40
4	17610152	2014-08-28 17:47:00.000000188	16.0	2014-08-28 17:47:00 UTC	-73.925023	40
...	...	...	...	...	...	
199995	42598914	2012-10-28 10:49:00.00000053	3.0	2012-10-28 10:49:00 UTC	-73.987042	40
199996	16382965	2014-03-14 01:09:00.0000008	7.5	2014-03-14 01:09:00 UTC	-73.984722	40
199997	27804658	2009-06-29 00:42:00.00000078	30.9	2009-06-29 00:42:00 UTC	-73.986017	40
199998	20259894	2015-05-20 14:56:25.0000004	14.5	2015-05-20 14:56:25 UTC	-73.997124	40
199999	11951496	2010-05-15 04:08:00.00000076	14.1	2010-05-15 04:08:00 UTC	-73.984395	40

200000 rows × 9 columns

In [14]:
```
data.dropna()
```

Out[14]:

	Unnamed: 0	key	fare_amount	pickup_datetime	pickup_longitude	pickup_
0	24238194	2015-05-07 19:52:06.0000003	7.5	2015-05-07 19:52:06 UTC	-73.999817	40
1	27835199	2009-07-17 20:04:56.0000002	7.7	2009-07-17 20:04:56 UTC	-73.994355	40
2	44984355	2009-08-24 21:45:00.00000061	12.9	2009-08-24 21:45:00 UTC	-74.005043	40
3	25894730	2009-06-26 08:22:21.0000001	5.3	2009-06-26 08:22:21 UTC	-73.976124	40
4	17610152	2014-08-28 17:47:00.000000188	16.0	2014-08-28 17:47:00 UTC	-73.925023	40
...	...	...	...	...	...	
199995	42598914	2012-10-28 10:49:00.00000053	3.0	2012-10-28 10:49:00 UTC	-73.987042	40
199996	16382965	2014-03-14 01:09:00.0000008	7.5	2014-03-14 01:09:00 UTC	-73.984722	40
199997	27804658	2009-06-29 00:42:00.00000078	30.9	2009-06-29 00:42:00 UTC	-73.986017	40
199998	20259894	2015-05-20 14:56:25.0000004	14.5	2015-05-20 14:56:25 UTC	-73.997124	40
199999	11951496	2010-05-15 04:08:00.00000076	14.1	2010-05-15 04:08:00 UTC	-73.984395	40

199999 rows × 9 columns

Visualization

In [15]:
```
import matplotlib.pyplot as pp
```

In [17]:
```
data
```

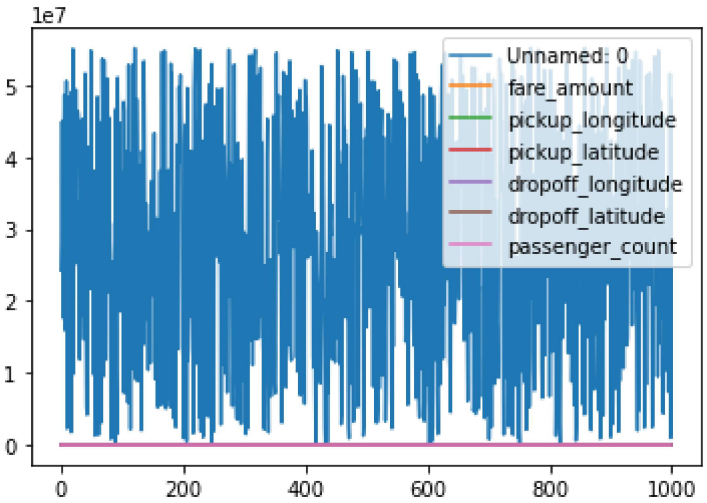
Out[17]:

	Unnamed: 0	key	fare_amount	pickup_datetime	pickup_longitude	pickup_
0	24238194	2015-05-07 19:52:06.0000003	7.5	2015-05-07 19:52:06 UTC	-73.999817	40
1	27835199	2009-07-17 20:04:56.0000002	7.7	2009-07-17 20:04:56 UTC	-73.994355	40
2	44984355	2009-08-24 21:45:00.00000061	12.9	2009-08-24 21:45:00 UTC	-74.005043	40
3	25894730	2009-06-26 08:22:21.0000001	5.3	2009-06-26 08:22:21 UTC	-73.976124	40
4	17610152	2014-08-28 17:47:00.000000188	16.0	2014-08-28 17:47:00 UTC	-73.925023	40
...	...	...	...	...	...	...
199995	42598914	2012-10-28 10:49:00.00000053	3.0	2012-10-28 10:49:00 UTC	-73.987042	40
199996	16382965	2014-03-14 01:09:00.0000008	7.5	2014-03-14 01:09:00 UTC	-73.984722	40
199997	27804658	2009-06-29 00:42:00.00000078	30.9	2009-06-29 00:42:00 UTC	-73.986017	40
199998	20259894	2015-05-20 14:56:25.0000004	14.5	2015-05-20 14:56:25 UTC	-73.997124	40
199999	11951496	2010-05-15 04:08:00.00000076	14.1	2010-05-15 04:08:00 UTC	-73.984395	40

200000 rows × 9 columns

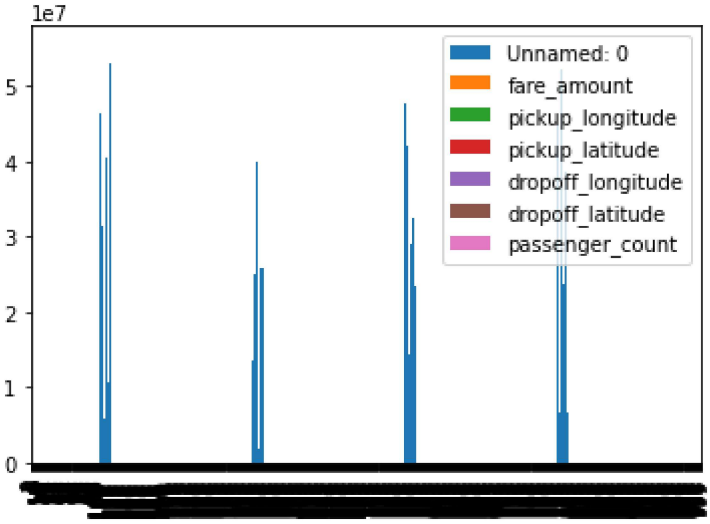
In [20]:
```
da=data.head(1000)
da.plot.line()
```

Out[20]: <AxesSubplot:>



In [21]:
```
da.plot.bar()
```

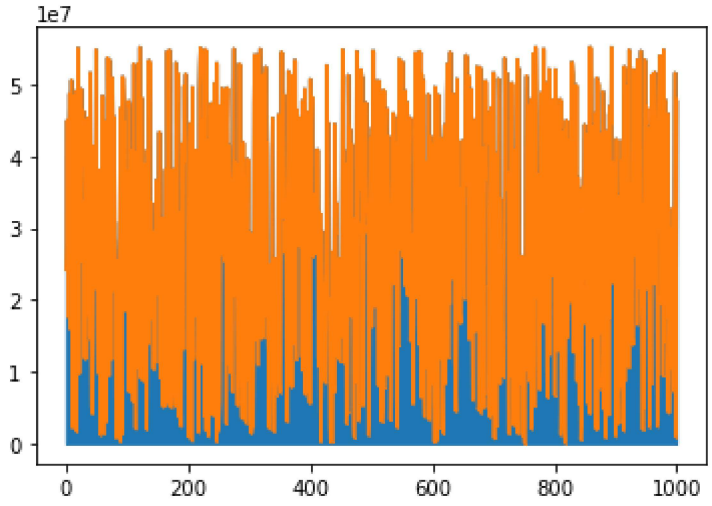
Out[21]: <AxesSubplot:>



In [23]:
```
da.plot.area()
```

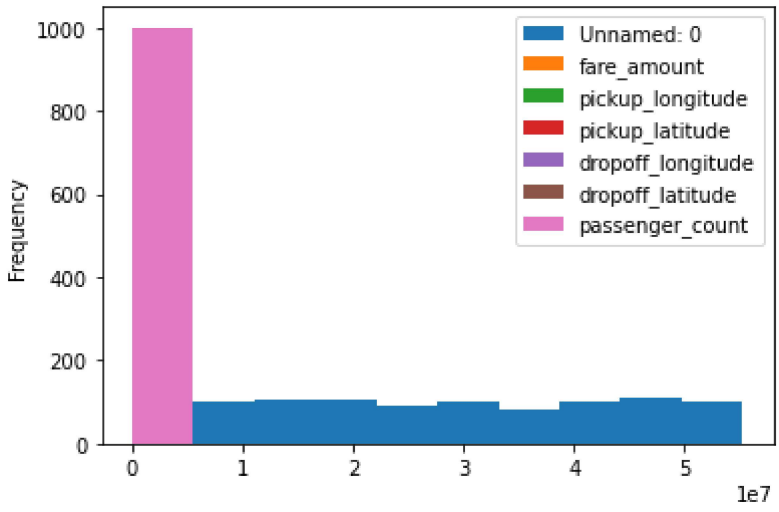
```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-23-c863071d6b60> in <module>
----> 1 da.plot.area()

C:\ProgramData\Anaconda3\lib\site-packages\pandas\plotting\_core.py in area(s
elf, x, y, **kwargs)
   1477             >>> ax = df.plot.area(x='day')
   1478         """
-> 1479         return self(kind="area", x=x, y=y, **kwargs)
   1480
   1481     def pie(self, **kwargs):

C:\ProgramData\Anaconda3\lib\site-packages\pandas\plotting\_core.py in __call
__(self, *args, **kwargs)
    953                     data.columns = label_name
    954
--> 955         return plot_backend.plot(data, kind=kind, **kwargs)
    956
    957     __call__.__doc__ = __doc__

C:\ProgramData\Anaconda3\lib\site-packages\pandas\plotting\_matplotlib\__init
__.py in plot(data, kind, **kwargs)
     59             kwargs["ax"] = getattr(ax, "left_ax", ax)
     60     plot_obj = PLOT_CLASSES[kind](data, **kwargs)
---> 61     plot_obj.generate()
     62     plot_obj.draw()
     63     return plot_obj.result

C:\ProgramData\Anaconda3\lib\site-packages\pandas\plotting\_matplotlib\core.p
y in generate(self)
    278         self._compute_plot_data()
    279         self._setup_subplots()
--> 280         self._make_plot()
    281         self._add_table()
    282         self._make_legend()

C:\ProgramData\Anaconda3\lib\site-packages\pandas\plotting\_matplotlib\core.p
y in _make_plot(self)
   1147             kwds["label"] = label
   1148
-> 1149             newlines = plotf(
   1150                 ax,
   1151                 x,

C:\ProgramData\Anaconda3\lib\site-packages\pandas\plotting\_matplotlib\core.p
y in _plot(cls, ax, x, y, style, column_num, stacking_id, is_errorbar, **kwd
s)
   1317         if column_num == 0:
   1318             cls._initialize_stacker(ax, stacking_id, len(y))
-> 1319         y_values = cls._get_stacked_values(ax, stacking_id, y, kwds
["label"])
   1320
   1321         # need to remove label, because subplots uses mpl legend as i
t is

C:\ProgramData\Anaconda3\lib\site-packages\pandas\plotting\_matplotlib\core.p
```

```
y in _get_stacked_values(cls, ax, stacking_id, values, label)
   1231             return ax._stacker_neg_prior[stacking_id] + values
   1232
-> 1233         raise ValueError(
   1234             "When stacked is True, each column must be either "
   1235             "all positive or negative."

ValueError: When stacked is True, each column must be either all positive or
negative.pickup_longitude contains both positive and negative values
```



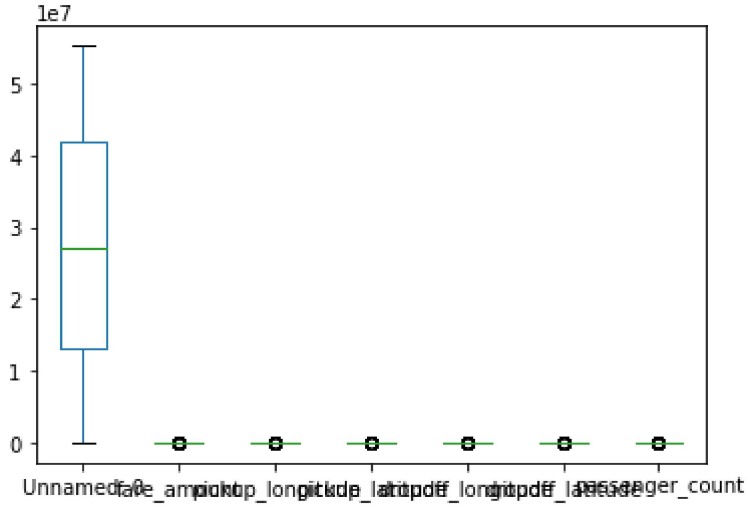
In [24]:
```
da.plot.hist()
```

Out[24]: <AxesSubplot:ylabel='Frequency'>



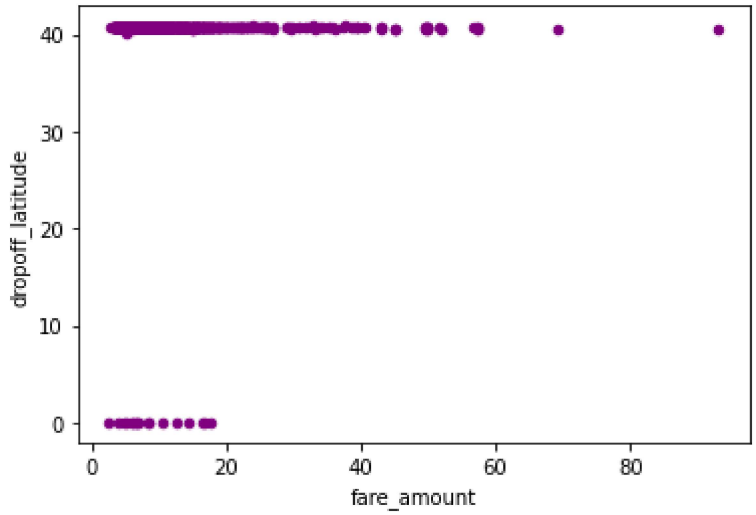
In [25]:
```
da.plot.box()
```

Out[25]: <AxesSubplot:>



In [29]:
```
da.plot.scatter(x='fare_amount',y='dropoff_latitude',color='purple')
```

Out[29]: <AxesSubplot:xlabel='fare_amount', ylabel='dropoff_latitude'>



In [30]:
```
data.plot.pie(subplots=True)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-30-7adddc1e29a1> in <module>
----> 1 data.plot.pie(subplots=True)

C:\ProgramData\Anaconda3\lib\site-packages\pandas\plotting\_core.py in pie(se
lf, **kwargs)
   1532         ):
   1533             raise ValueError("pie requires either y column or 'subplo
ts=True'")
-> 1534         return self(kind="pie", **kwargs)
   1535
   1536     def scatter(self, x, y, s=None, c=None, **kwargs):

C:\ProgramData\Anaconda3\lib\site-packages\pandas\plotting\_core.py in __call
__(self, *args, **kwargs)
    953                     data.columns = label_name
    954
--> 955         return plot_backend.plot(data, kind=kind, **kwargs)
    956
    957     __call__.__doc__ = __doc__

C:\ProgramData\Anaconda3\lib\site-packages\pandas\plotting\_matplotlib\__init
__.py in plot(data, kind, **kwargs)
     58                 ax = plt.gca()
     59             kwargs["ax"] = getattr(ax, "left_ax", ax)
---> 60     plot_obj = PLOT_CLASSES[kind](data, **kwargs)
     61     plot_obj.generate()
     62     plot_obj.draw()

C:\ProgramData\Anaconda3\lib\site-packages\pandas\plotting\_matplotlib\core.p
y in __init__(self, data, kind, **kwargs)
   1538     def __init__(self, data, kind=None, **kwargs):
   1539         data = data.fillna(value=0)
-> 1540         if (data < 0).any().any():
   1541             raise ValueError(f"{kind} doesn't allow negative values")
   1542         MPLPlot.__init__(self, data, kind=kind, **kwargs)

C:\ProgramData\Anaconda3\lib\site-packages\pandas\core\ops\common.py in new_m
ethod(self, other)
     63         other = item_from_zerodim(other)
     64
---> 65         return method(self, other)
     66
     67     return new_method

C:\ProgramData\Anaconda3\lib\site-packages\pandas\core\arraylike.py in __lt__
(self, other)
     35     @unpack_zerodim_and_defer("__lt__")
     36     def __lt__(self, other):
---> 37         return self._cmp_method(other, operator.lt)
     38
     39     @unpack_zerodim_and_defer("__le__")

C:\ProgramData\Anaconda3\lib\site-packages\pandas\core\frame.py in _cmp_metho
d(self, other, op)
   5969
```

```
   5970         # See GH#4537 for discussion of scalar op behavior
-> 5971         new_data = self._dispatch_frame_op(other, op, axis=axis)
   5972         return self._construct_result(new_data)
   5973

C:\ProgramData\Anaconda3\lib\site-packages\pandas\core\frame.py in _dispatch_
frame_op(self, right, func, axis)
   6006         if not is_list_like(right):
   6007             # i.e. scalar, faster than checking np.ndim(right) == 0
-> 6008             bm = self._mgr.apply(array_op, right=right)
   6009             return type(self)(bm)
   6010

C:\ProgramData\Anaconda3\lib\site-packages\pandas\core\internals\managers.py
in apply(self, f, align_keys, ignore_failures, **kwargs)
    423             try:
    424                 if callable(f):
--> 425                     applied = b.apply(f, **kwargs)
    426                 else:
    427                     applied = getattr(b, f)(**kwargs)

C:\ProgramData\Anaconda3\lib\site-packages\pandas\core\internals\blocks.py in
apply(self, func, **kwargs)
    376         """
    377         with np.errstate(all="ignore"):
--> 378             result = func(self.values, **kwargs)
    379
    380         return self._split_op_result(result)

C:\ProgramData\Anaconda3\lib\site-packages\pandas\core\ops\array_ops.py in co
mparison_op(left, right, op)
    241
    242     elif is_object_dtype(lvalues.dtype):
--> 243         res_values = comp_method_OBJECT_ARRAY(op, lvalues, rvalues)
    244
    245     else:

C:\ProgramData\Anaconda3\lib\site-packages\pandas\core\ops\array_ops.py in co
mp_method_OBJECT_ARRAY(op, x, y)
     53         result = libops.vec_compare(x.ravel(), y.ravel(), op)
     54     else:
---> 55         result = libops.scalar_compare(x.ravel(), y, op)
     56     return result.reshape(x.shape)
     57

pandas\_libs\ops.pyx in pandas._libs.ops.scalar_compare()

TypeError: '<' not supported between instances of 'str' and 'int'
```

In [ ]: