```
import pandas as pd
import numpy as np
import datetime
import matplotlib as plt
Dates = ["2018-08-14","2018-08-14","2020-11-14","2020-05-17","2021-09-
15","2021-12-14"]
Courses =["Spark","PySpark","Hadoop","Python","Pandas","Hadoop"]
print(type(Dates))
print(type(Courses))

df = pd.DataFrame({'InsertedDate':
pd.to_datetime(Dates)},index=Courses)
print("DataFrame:\n", df)
print(type(df))
```

```
<class 'list'>
<class 'list'>
DataFrame:
         InsertedDate
Spark     2018-08-14
PySpark   2018-08-14
Hadoop    2020-11-14
Python    2020-05-17
Pandas    2021-09-15
Hadoop    2021-12-14
<class 'pandas.core.frame.DataFrame'>
```

```
df['Year'] = df['InsertedDate'].dt.strftime('%Y')  # string , =str
df['Month'] = df['InsertedDate'].dt.strftime('%m')
df['Date'] = df['InsertedDate'].dt.strftime('%d')
df['Day'] = df['InsertedDate'].dt.strftime('%A')  # A fro days
df["Amount"] = [1232,2112,322,1221,2123,1320]

df

#print("Get month and year from datetime column:\n", df)
```

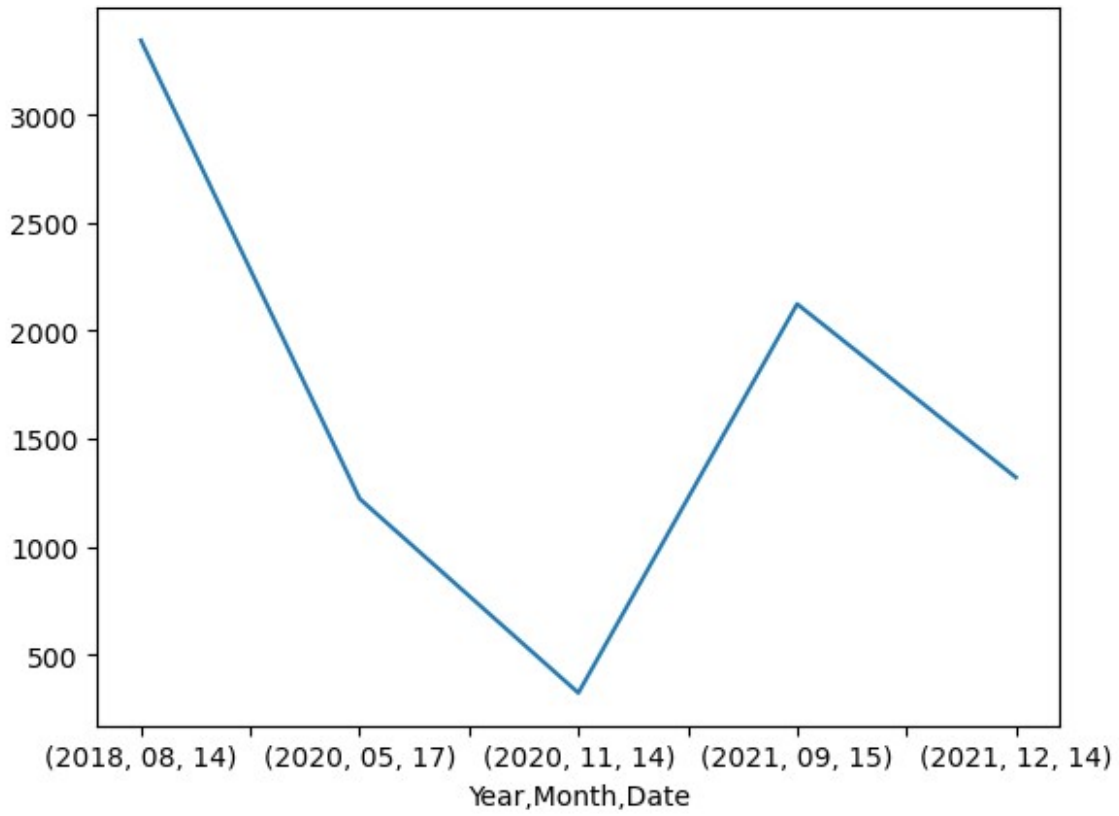
```
        InsertedDate  Year Month Date        Day  Amount
Spark     2018-08-14  2018    08   14    Tuesday    1232
PySpark   2018-08-14  2018    08   14    Tuesday    2112
Hadoop    2020-11-14  2020    11   14   Saturday     322
Python    2020-05-17  2020    05   17     Sunday    1221
Pandas    2021-09-15  2021    09   15  Wednesday    2123
Hadoop    2021-12-14  2021    12   14    Tuesday    1320
```

```
df.set_index(['Year'])
```

```
     InsertedDate Month Date        Day  Amount
Year
2018   2018-08-14    08   14    Tuesday    1232
2018   2018-08-14    08   14    Tuesday    2112
```

```
2020   2020-11-14    11   14   Saturday     322
2020   2020-05-17    05   17     Sunday    1221
2021   2021-09-15    09   15  Wednesday    2123
2021   2021-12-14    12   14    Tuesday    1320
```

```
x=df.groupby(['Year','Month',"Date"])['Amount'].sum().plot()
#aggregiate function sum, mean,median,mode,min,max,std,count,
#print(tuple(x))
# In pandas, the groupby function is used to split a DataFrame into groups based on some criteria,
#apply a function to each group independently, and then combine the results back into a DataFrame. .
#This is a powerful tool for data analysis and aggregation.
```




```
df.head
```

```
<bound method NDFrame.head of         InsertedDate  Year Month Date
Day  Amount
Spark     2018-08-14  2018    08   14    Tuesday    1232
PySpark   2018-08-14  2018    08   14    Tuesday    2112
Hadoop    2020-11-14  2020    11   14   Saturday     322
Python    2020-05-17  2020    05   17     Sunday    1221
Pandas    2021-09-15  2021    09   15  Wednesday    2123
Hadoop    2021-12-14  2021    12   14    Tuesday    1320>
```

```
data structure
# list
# dictionary
# String
# Tuple
# Set


lst = [1,-2,"anwar",100]
print(lst)
```

```
  Cell In[18], line 1
    data structure
         ^
SyntaxError: invalid syntax
```

```
lst = [1,-2,"anwar",100]
print(lst)
print(type(lst))
# indexing positive last element will be eliminate
# indixing negative -1 element will be eliminate
lst[0]=-1
print(lst)
print(lst[0:5])
print(lst[-4:-1])
```

```
[1, -2, 'anwar', 100]
<class 'list'>
[-1, -2, 'anwar', 100]
[-1, -2, 'anwar', 100]
[-1, -2, 'anwar']
```

```
string = "I love Pakistan"
print(string)
print(type(string))
print(string[-15:])
print(string[0:])
```

```
I love Pakistan
<class 'str'>
I love Pakistan
I love Pakistan
```