1 Introduction to the Data Exploration Components (Series and Data Frames) using Pandas in python

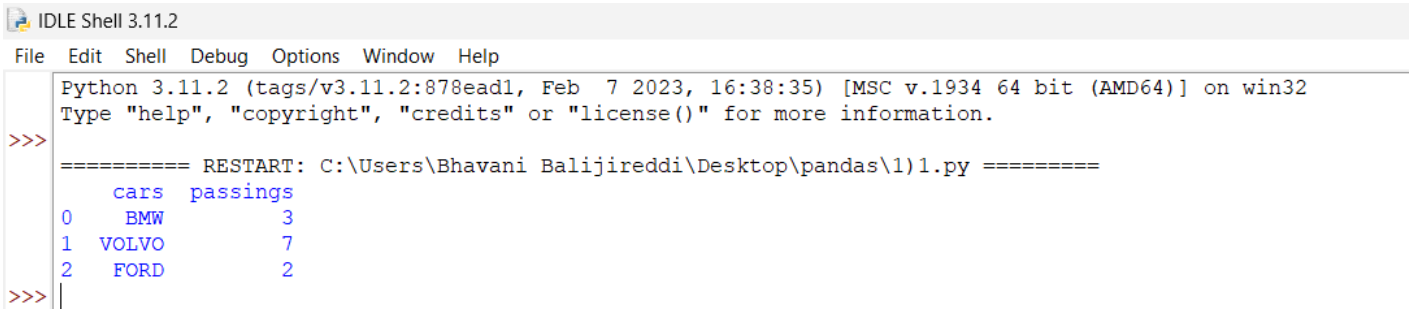
a.Import Pandas
b.Loading the data various formats (.XLS, .TXT, .CSV, JSON) using Pandas
c.Describe Data, Modify Data, Grouping Data, Filtering Data
d.Converting a variable to a different data type back to a CSV, JSON, or SQL

1.1 Import Pandas

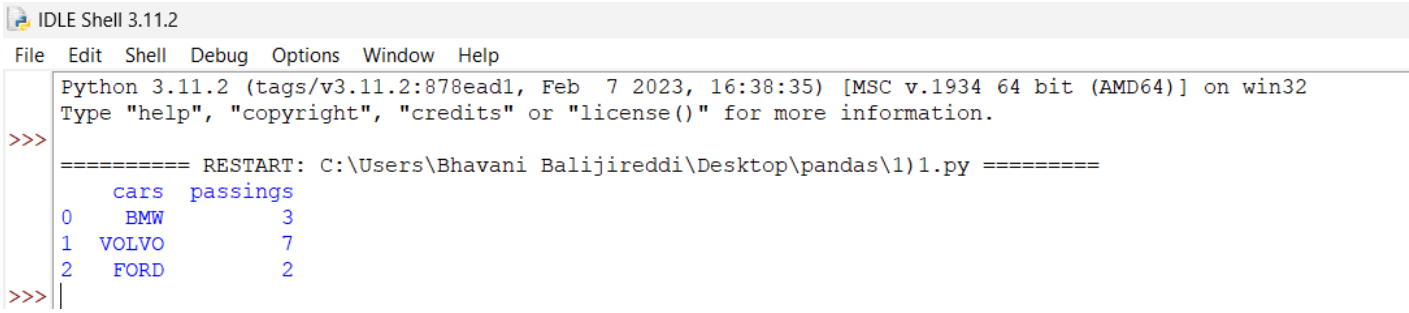
Program:

```
import pandas
mydataset={'cars':["BMW","VOLVO","FORD"], 'passings':[3,7,2]}
myvar=pandas.DataFrame(mydataset)
print(myvar)
```

Expected output:



Observed output:

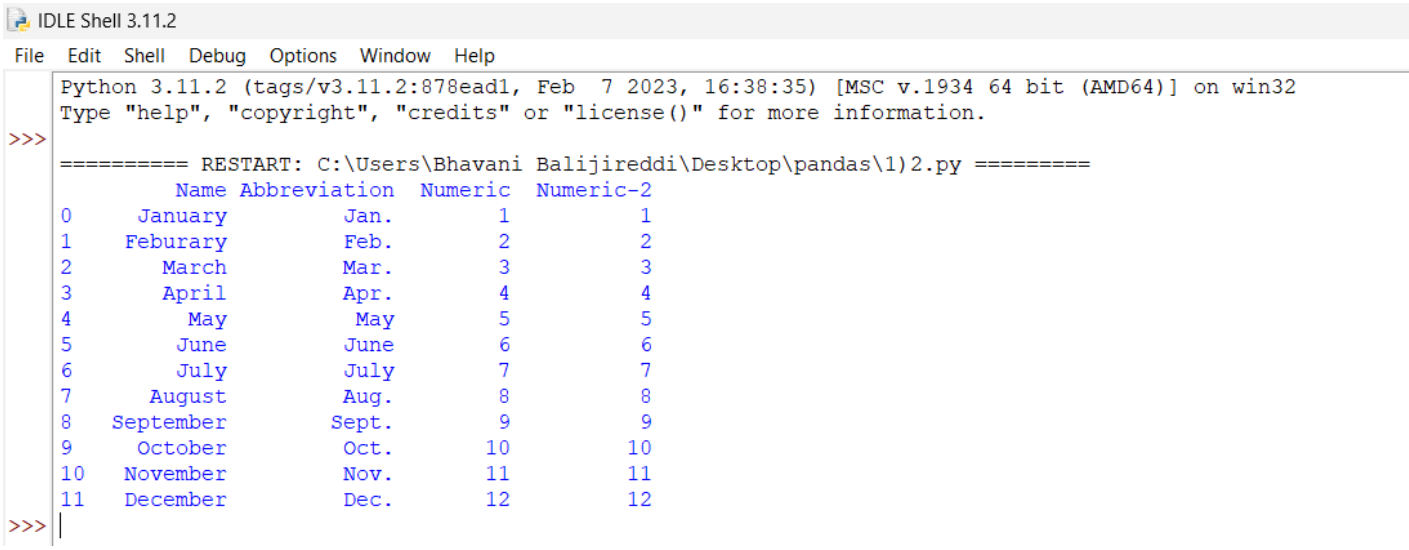


1.2 Loading the data various formats (.XLS, .TXT, .CSV, JSON) using Pandas

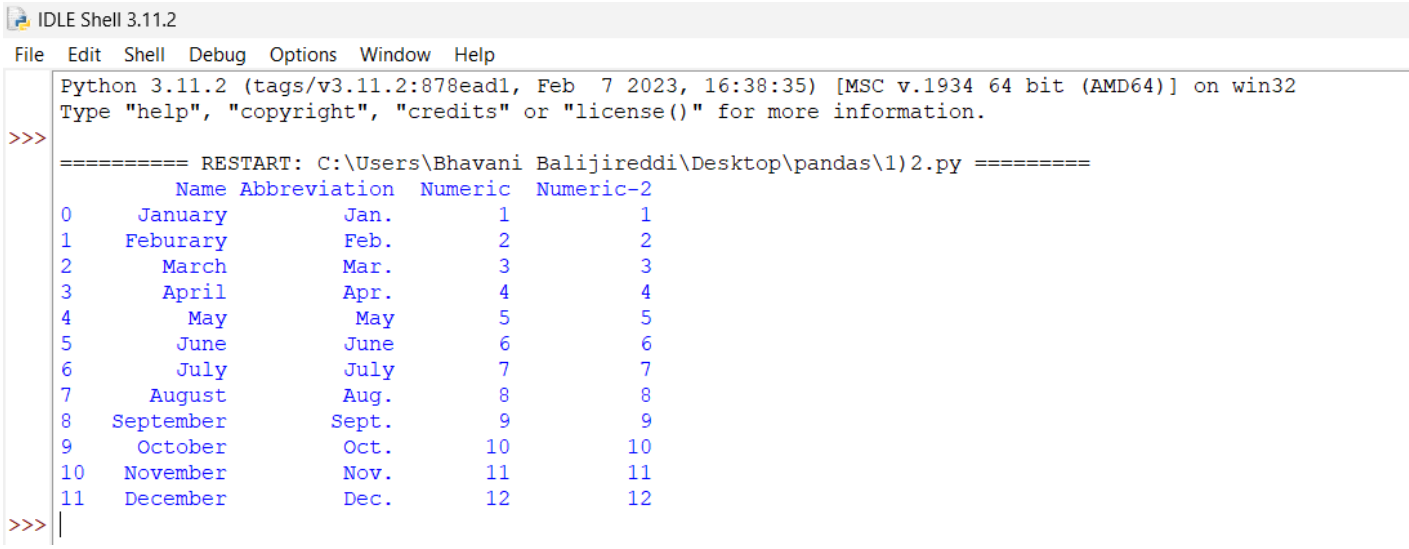
Program:

```
import pandas as pd
d=pd.read_csv("month.csv")
df=pd.DataFrame(d)
print(df)
```

Expected output:



Observed output:



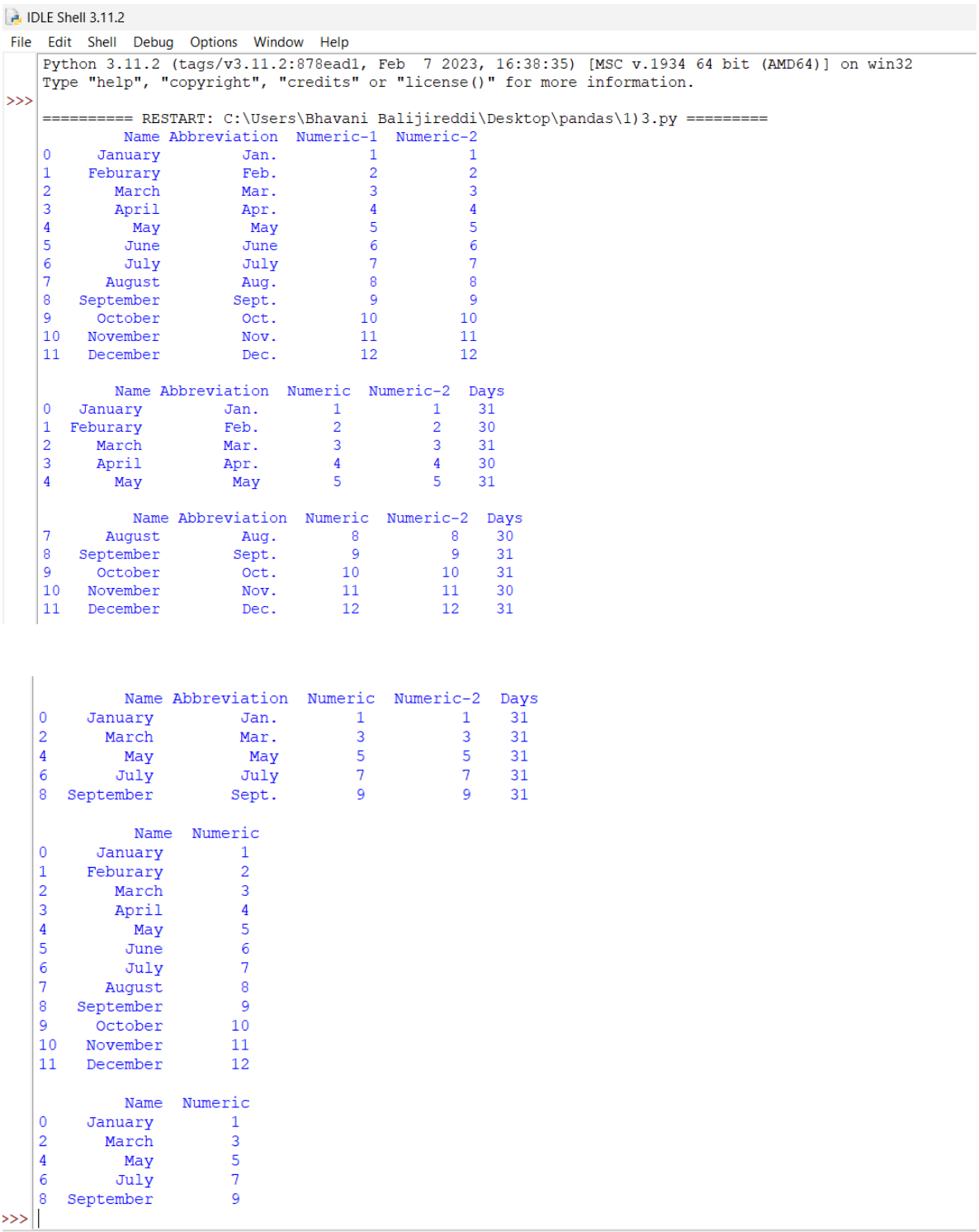
1.3 Describe Data, Modify Data, Grouping Data, Filtering Data

Program:

```
import pandas as pd
d=pd.read_csv("month.csv")
df=pd.DataFrame(d)
print(df.rename(columns={'Numeric':'Numeric-1'}))
```

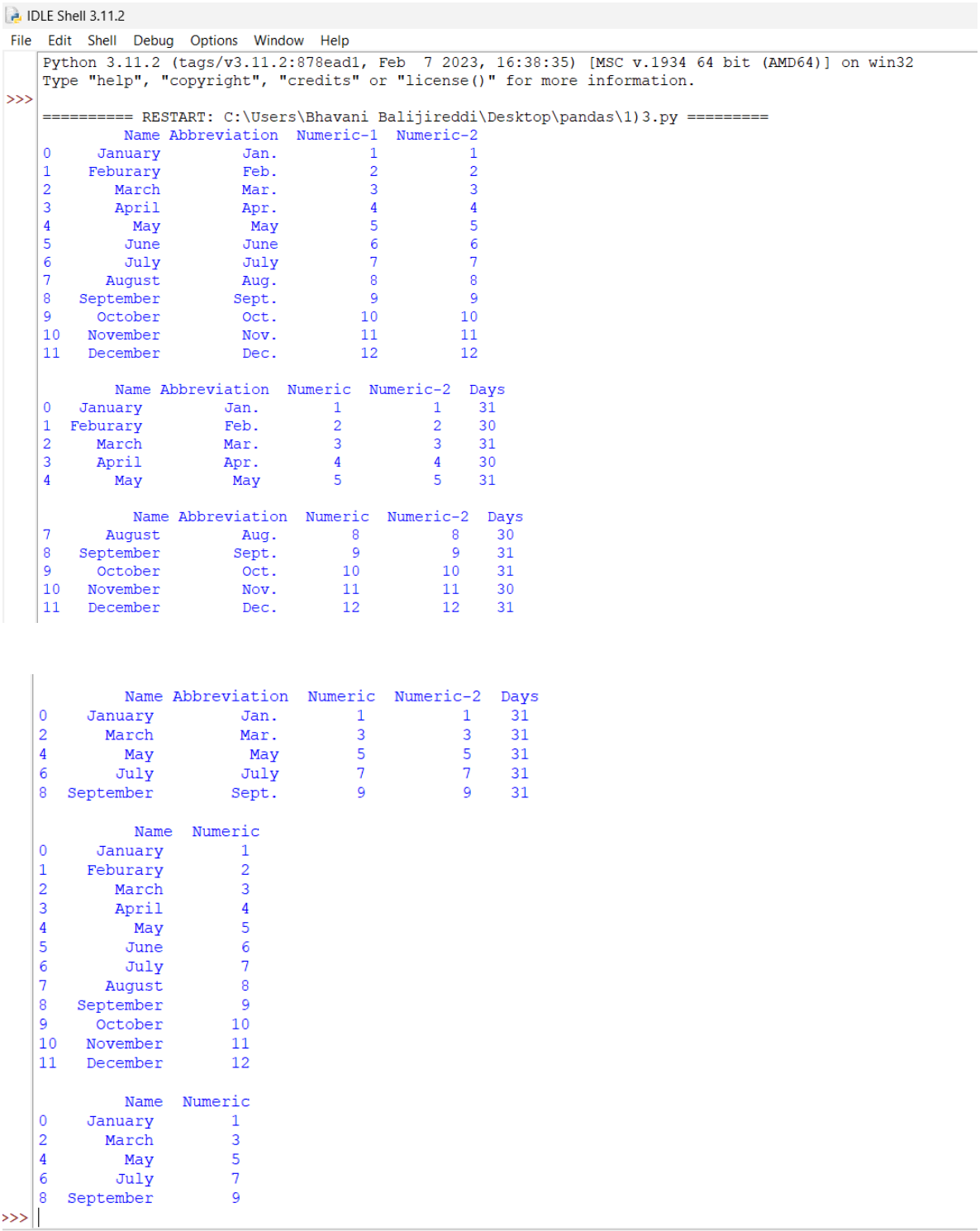
```
df['Days']=[31,30,31,30,31,30,31,30,31,31,30,31]
print( '\n',df.head())
print('\n',df.tail())
print('\n',df[0:10:2])
print('\n',df[['Name','Numeric']])
print('\n',df[['Name','Numeric']][0:10:2])
```

Expected output:



```
IDLE Shell 3.11.2
File  Edit  Shell  Debug  Options  Window  Help
Python 3.11.2 (tags/v3.11.2:878ead1, Feb  7 2023, 16:38:35) [MSC v.1934 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: C:\Users\Bhavani Balijireddi\Desktop\pandas\1)3.py =========
         Name Abbreviation  Numeric-1  Numeric-2
0     January         Jan.          1          1
1    Feburary         Feb.          2          2
2       March         Mar.          3          3
3       April         Apr.          4          4
4         May          May          5          5
5        June         June          6          6
6        July         July          7          7
7      August         Aug.          8          8
8   September        Sept.          9          9
9     October         Oct.         10         10
10   November         Nov.         11         11
11   December         Dec.         12         12

        Name Abbreviation  Numeric  Numeric-2  Days
0    January         Jan.        1          1    31
1   Feburary         Feb.        2          2    30
2      March         Mar.        3          3    31
3      April         Apr.        4          4    30
4        May          May        5          5    31

          Name Abbreviation  Numeric  Numeric-2  Days
7      August         Aug.        8          8    30
8   September        Sept.        9          9    31
9     October         Oct.       10         10    31
10   November         Nov.       11         11    30
11   December         Dec.       12         12    31


         Name Abbreviation  Numeric  Numeric-2  Days
0     January         Jan.        1          1    31
2       March         Mar.        3          3    31
4         May          May        5          5    31
6        July         July        7          7    31
8   September        Sept.        9          9    31

          Name  Numeric
0     January        1
1    Feburary        2
2       March        3
3       April        4
4         May        5
5        June        6
6        July        7
7      August        8
8   September        9
9     October       10
10   November       11
11   December       12

          Name  Numeric
0     January        1
2       March        3
4         May        5
6        July        7
8   September        9
>>>
```

Observed output:

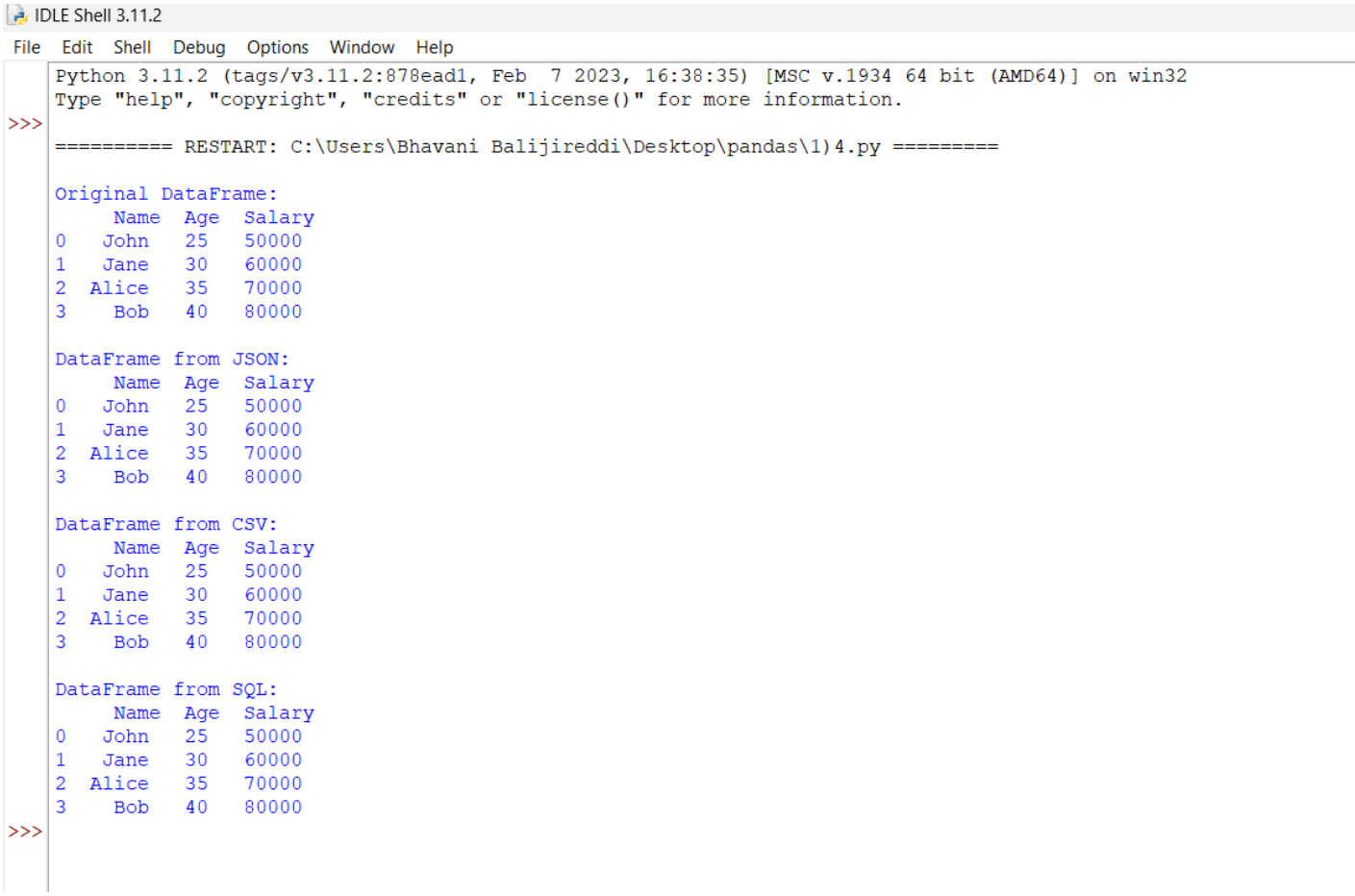


1.4 Converting a variable to a different data type back to a CSV, JSON, or SQL

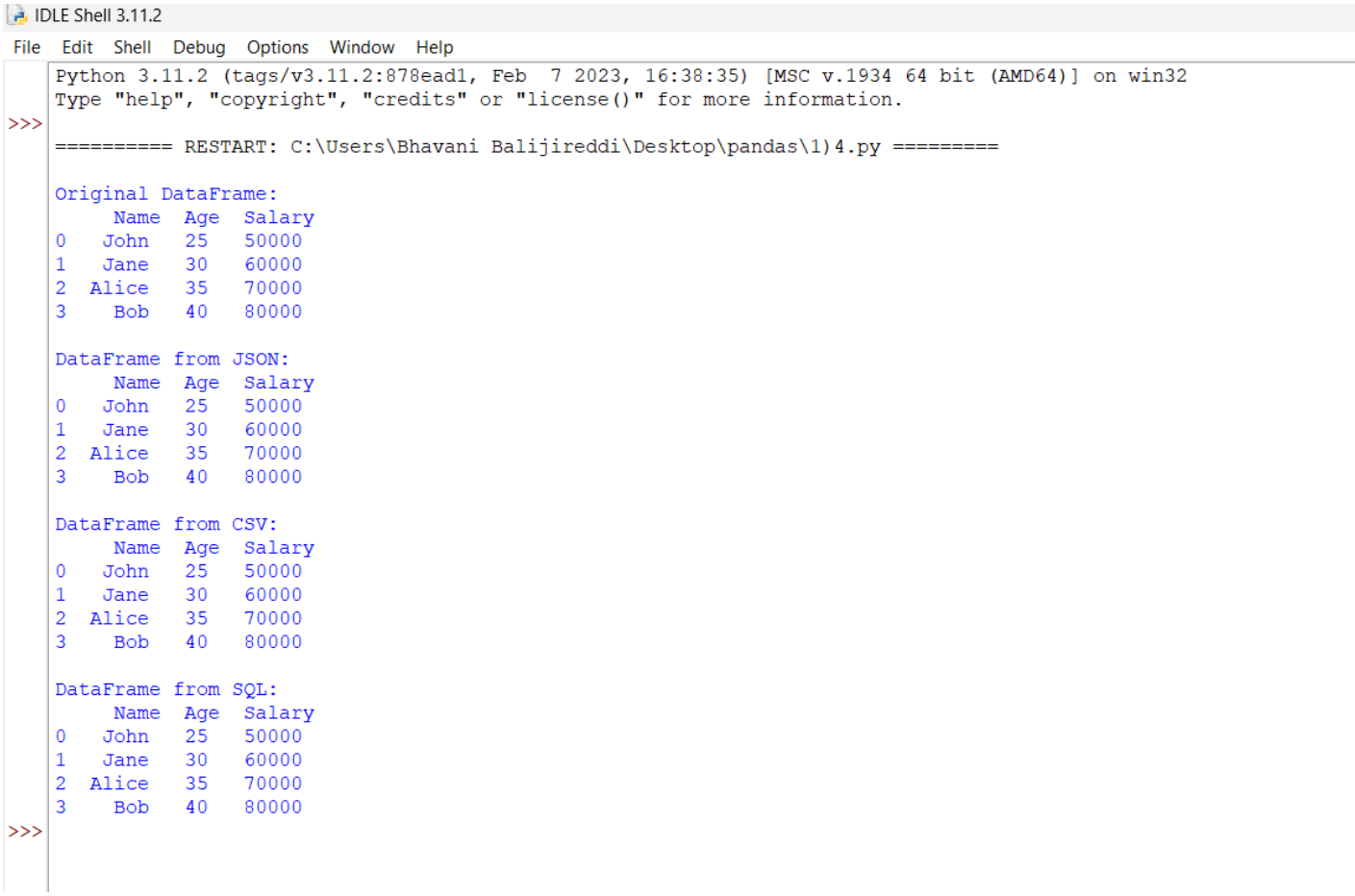
Program:

```
import pandas as pd
import io
import sqlite3
sample_data = {
'Name': ['John', 'Jane', 'Alice', 'Bob'],
'Age': [25, 30, 35, 40],
'Salary': [50000, 60000, 70000, 80000]
}
df = pd.DataFrame(sample_data)
json_data = df.to_json()
df_from_json = pd.read_json(json_data)
csv_data = df.to_csv(index=False)
df_from_csv = pd.read_csv(io.StringIO(csv_data))
conn = sqlite3.connect('example.db')
df.to_sql('employee', conn, if_exists='replace', index=False)
df_from_sql = pd.read_sql('SELECT * FROM employee', conn)
print('\n Original DataFrame:\n', df)
print('\n DataFrame from JSON:\n', df_from_json)
print('\n DataFrame from CSV:\n', df_from_csv)
print('\n DataFrame from SQL:\n', df_from_sql)
```

Expected output:



Observed output:



2 Reading and writing files

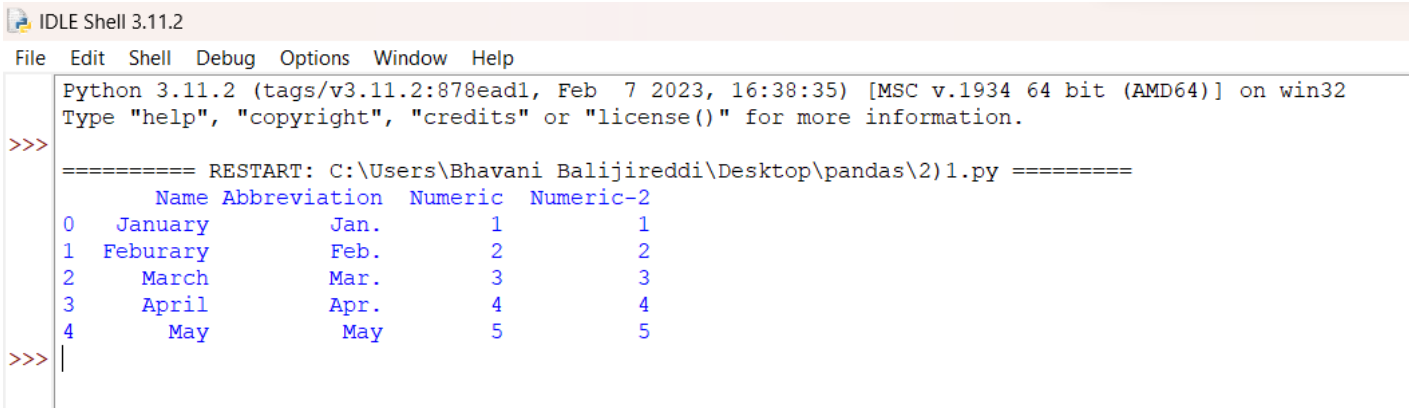
a. Reading a CSV File
b. Writing content of data frames to CSV File
c. Reading an Excel File
d. Writing content of data frames to Excel File

2.1 Reading a CSV File

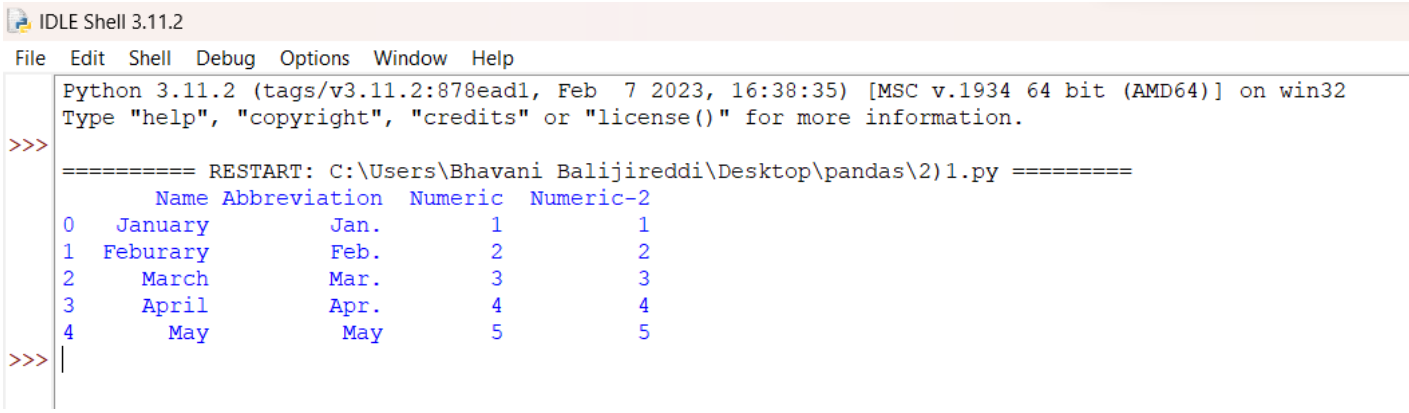
Program:

```
import pandas as pd
df = pd.read_csv("month.csv")
print(df.head())
```

Expected output:



Observed output:

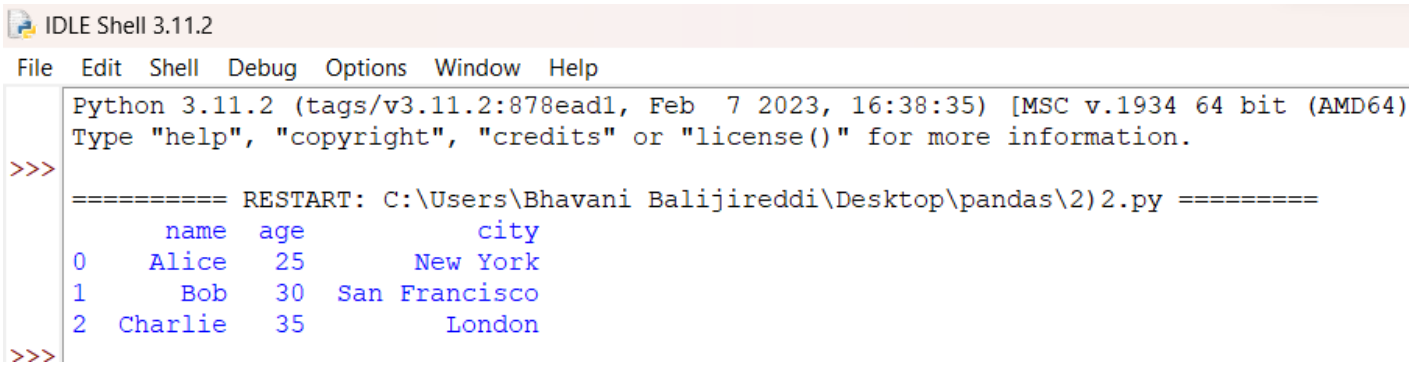


2.2 Writing content of data frames to CSV File

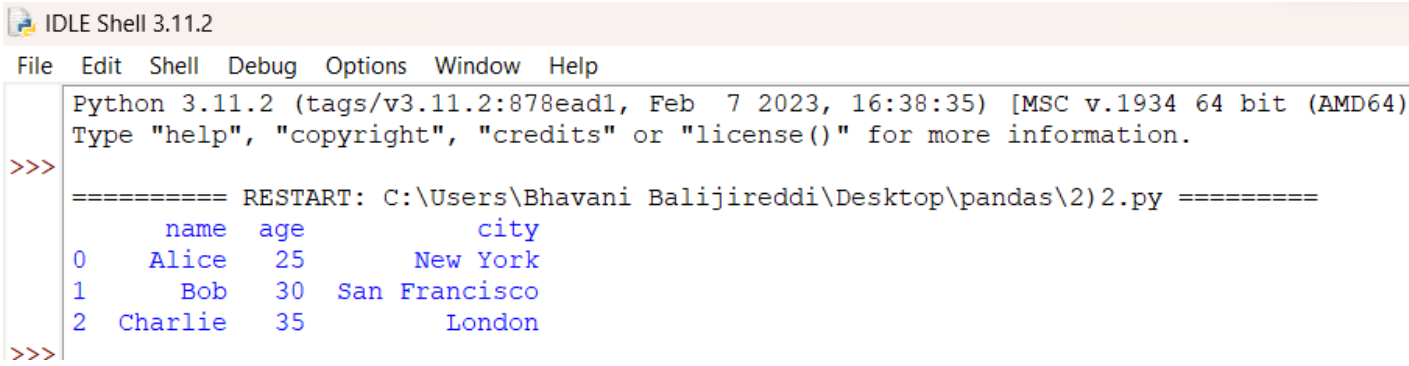
Program:

```
import pandas as pd
data = {'name': ['Alice', 'Bob', 'Charlie'],
'age': [25, 30, 35],
'city': ['New York', 'San Francisco', 'London'] }
df = pd.DataFrame(data)
df.to_csv('my_data.csv', index=False)
print(df)
```

Expected output:



Observed output:

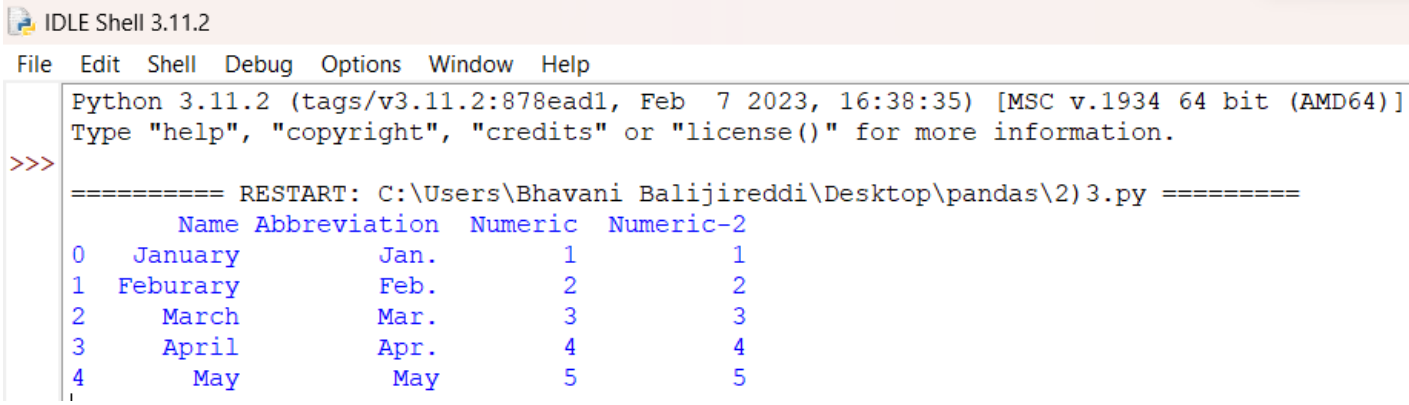


2.3 Reading an Excel File

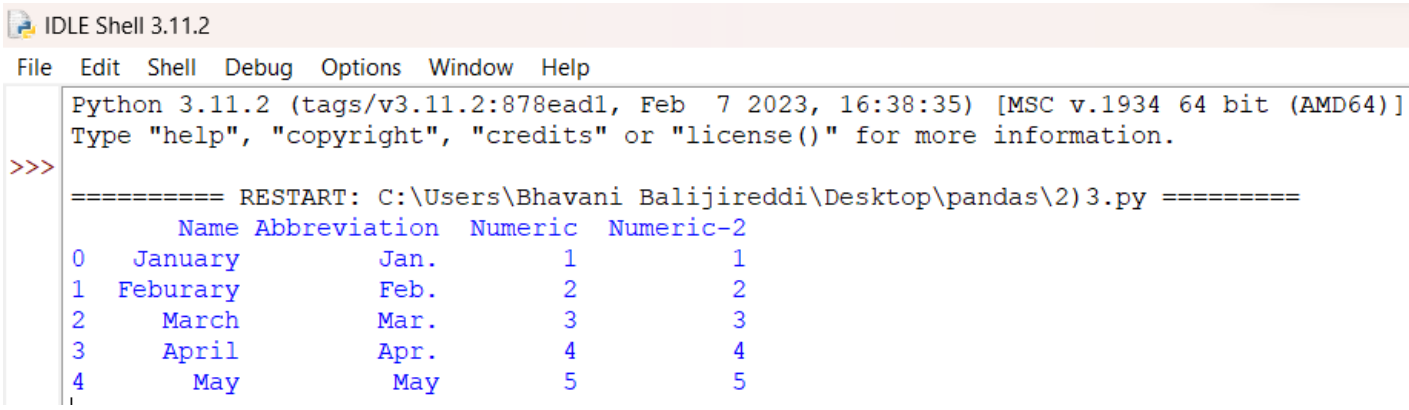
Program:

```
import pandas as pd
df = pd.read_excel('month.xlsx')
print(df.head())
```

Expected output:



Observed output:

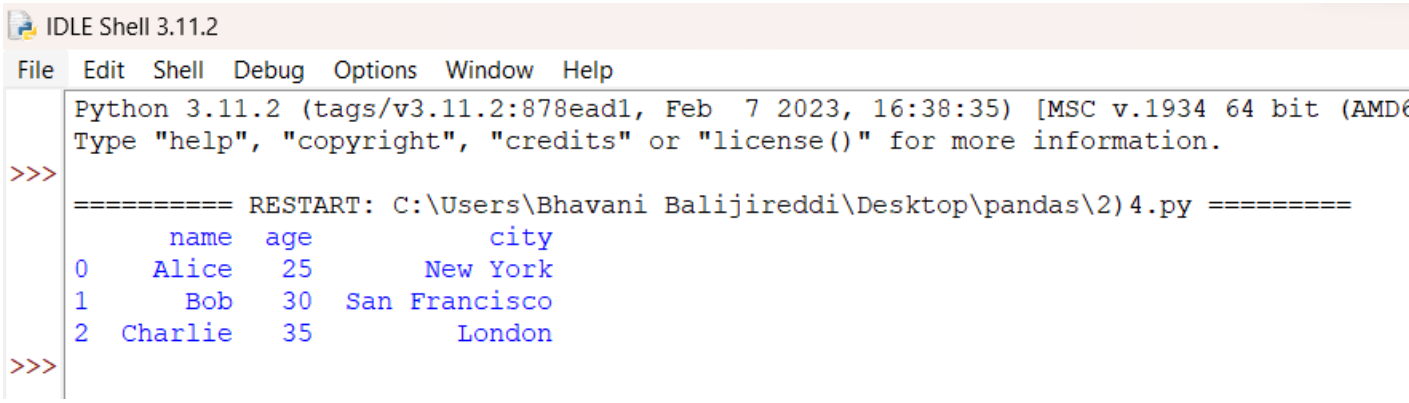


2.4 Writing content of data frames to Excel File

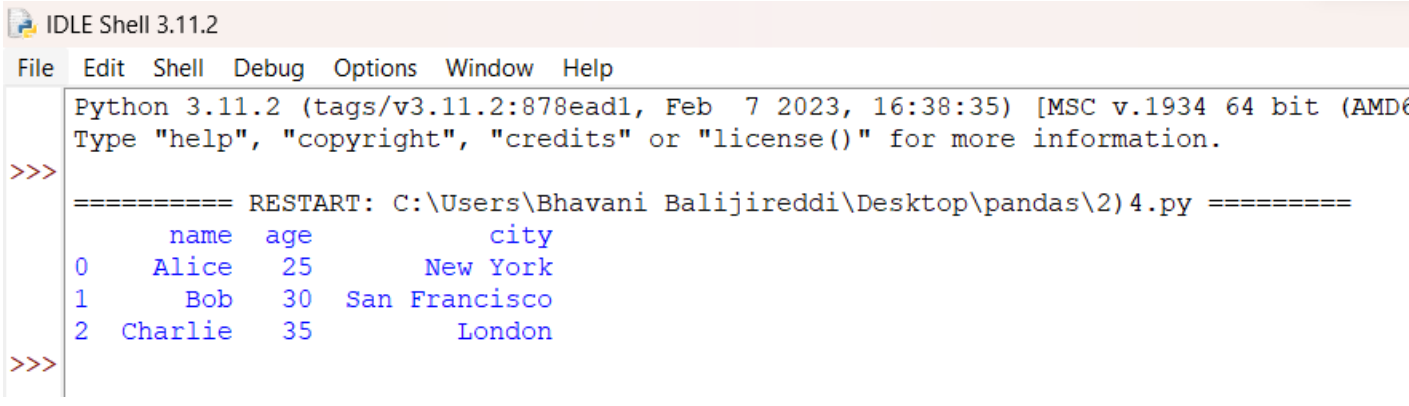
Program:

```
import pandas as pd
data = {'name': ['Alice', 'Bob', 'Charlie'],
'age': [25, 30, 35],
'city': ['New York', 'San Francisco', 'London']}
df = pd.DataFrame(data)
df.to_excel('data.xlsx', index=False)
print(df)
```

Expected output:



Observed output:



3 Getting the Dataset

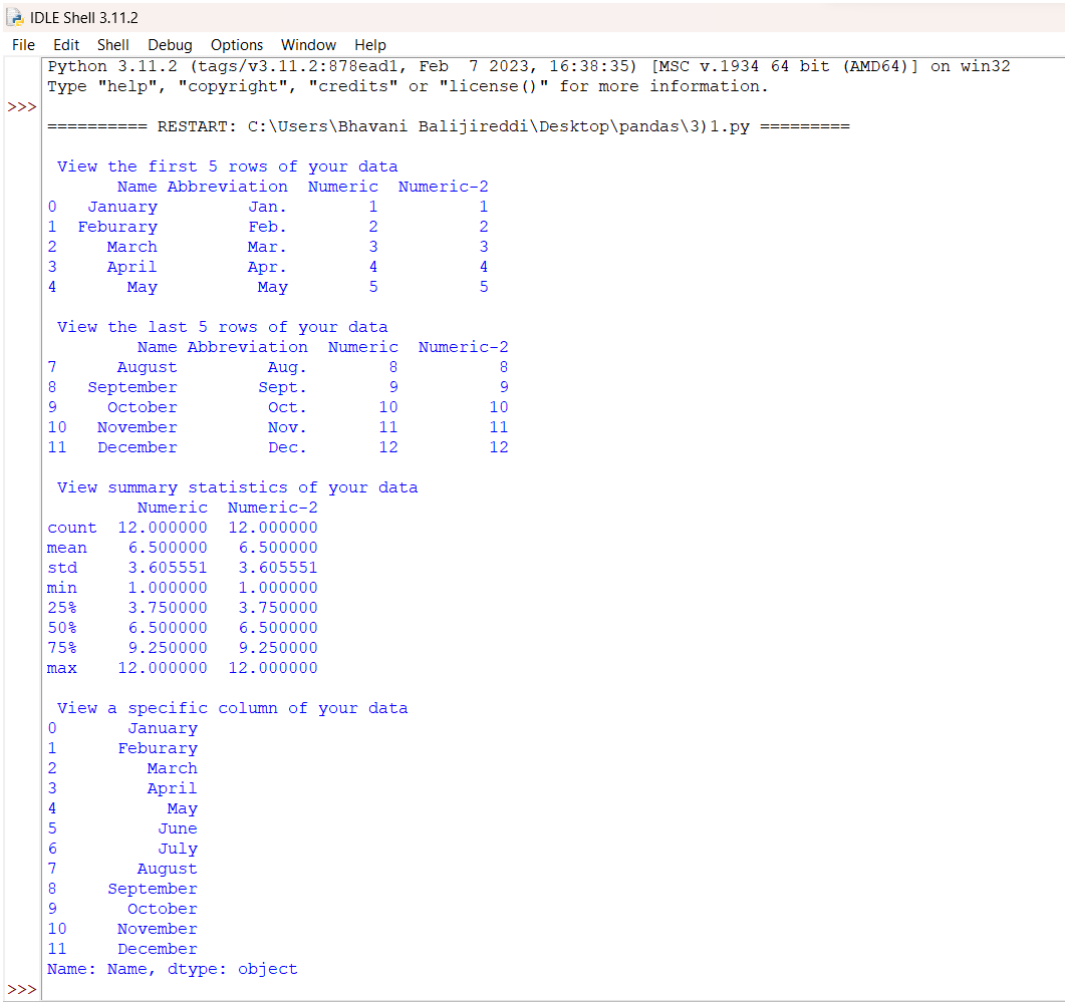
a. Viewing your data
b. Data Set Description
c. Describe as category
d. Handling duplicates
e. Number of observations Per Category
f. Column cleanup

3.1 Viewing your data

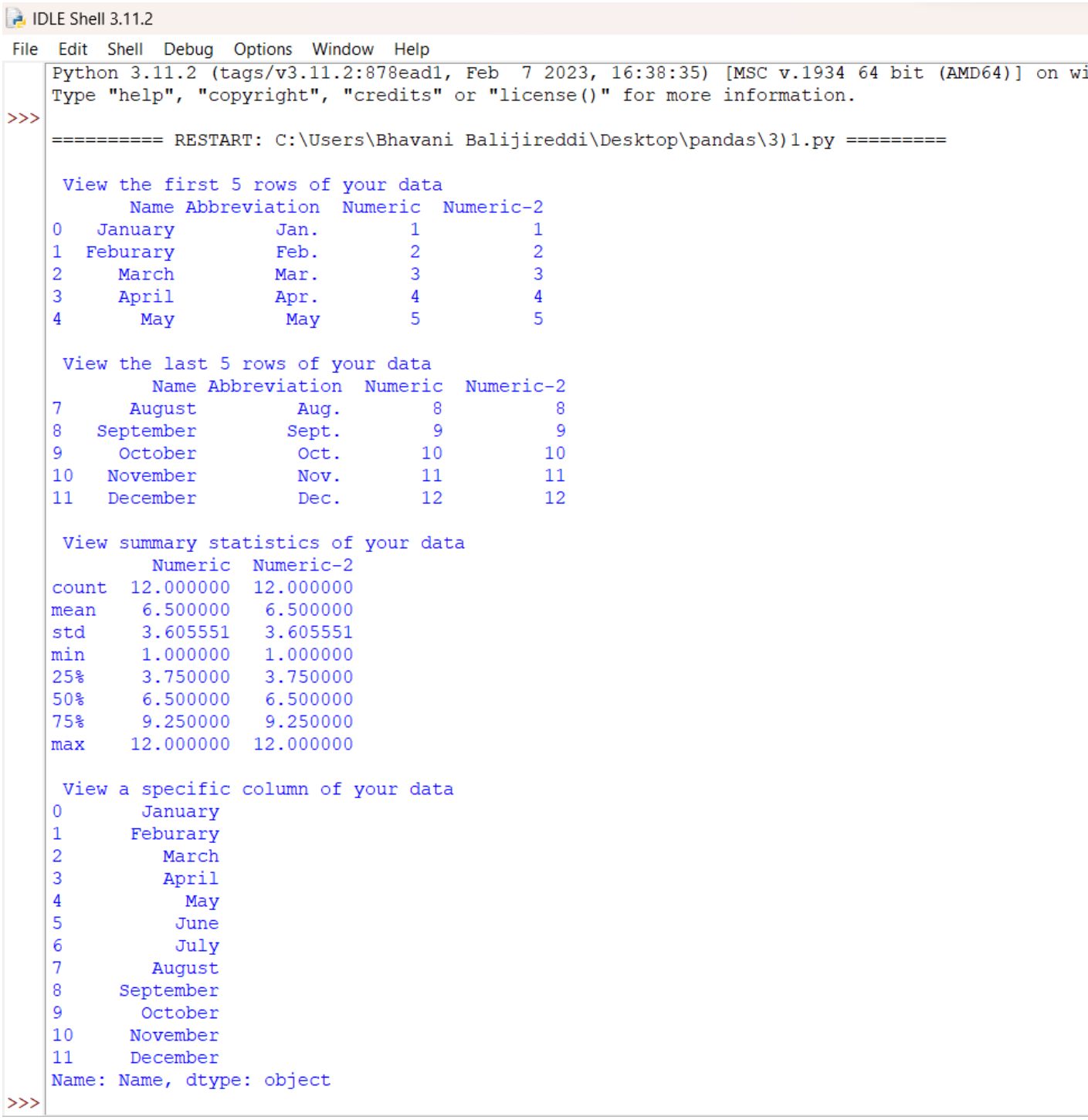
Program:

```
import pandas as pd
data = pd.read_csv('month.csv')
print('\n', "View the first 5 rows of your data")
print(data.head())
print('\n',"View the last 5 rows of your data")
print(data.tail())
print('\n',"View summary statistics of your data")
print(data.describe())
print('\n',"View a specific column of your data")
print(data['Name'])
```

Expected output:



Observed output:

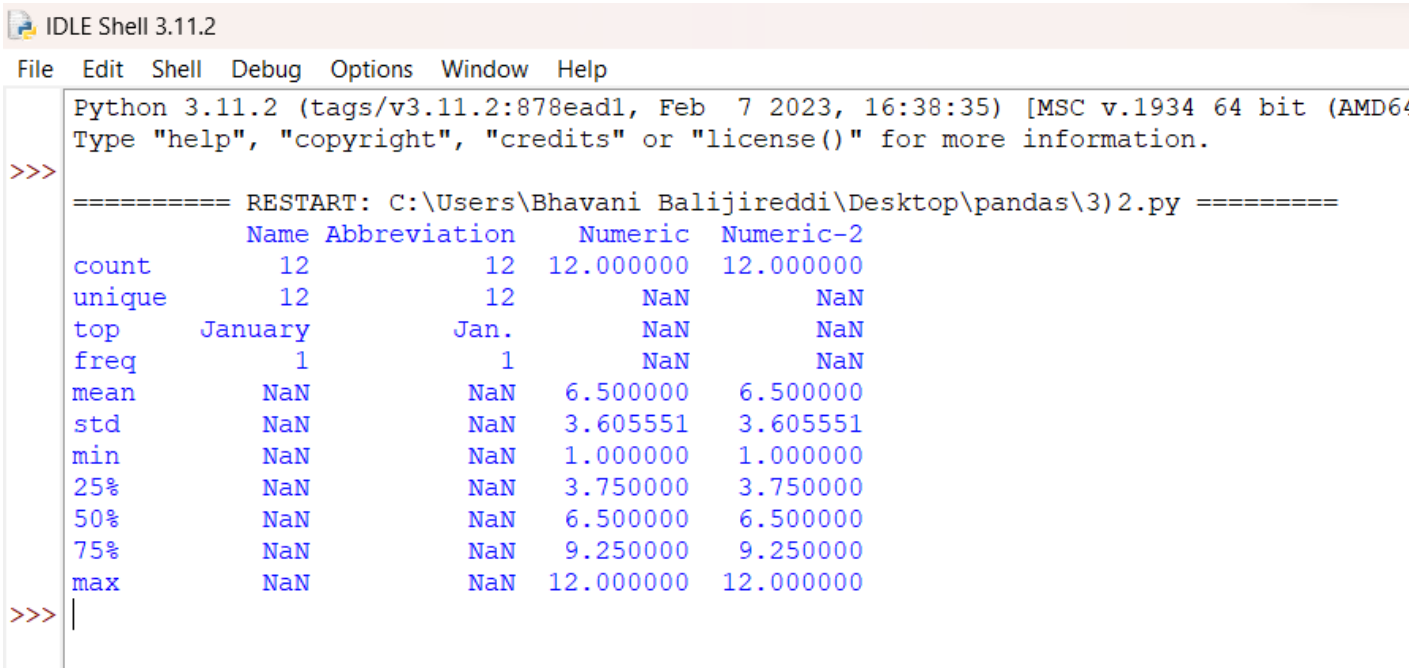


3.2 Data Set Description

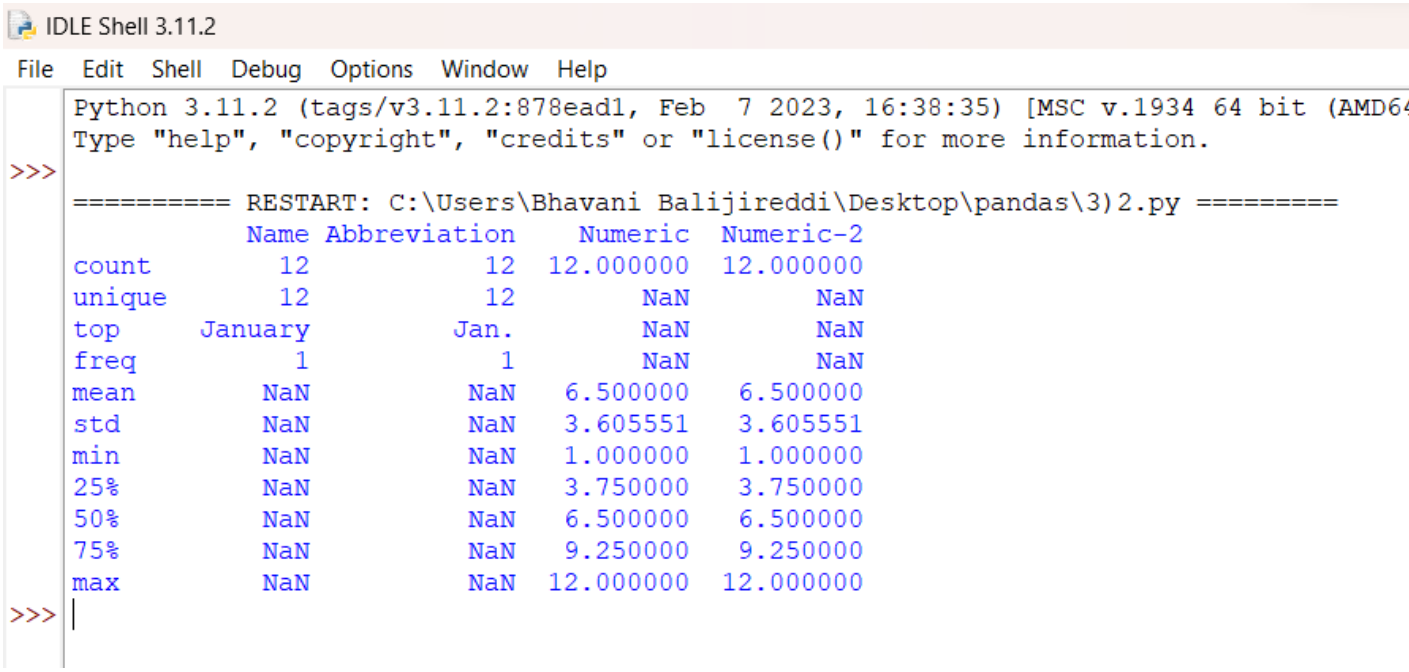
Program:

```
import pandas as pd
data = pd.read_csv('month.csv')
print(data.describe(include='all'))
```

Expected output:



Observed output:

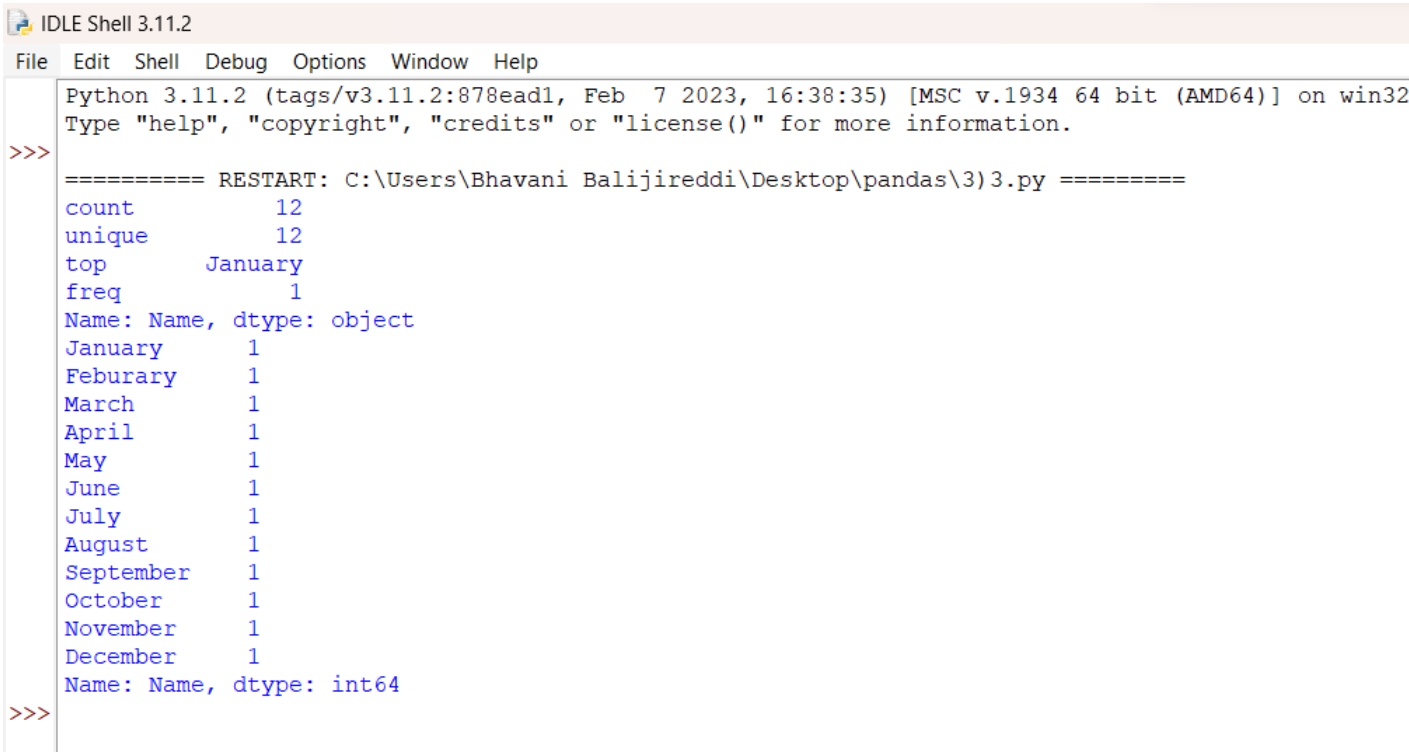


3.3 Describe as category

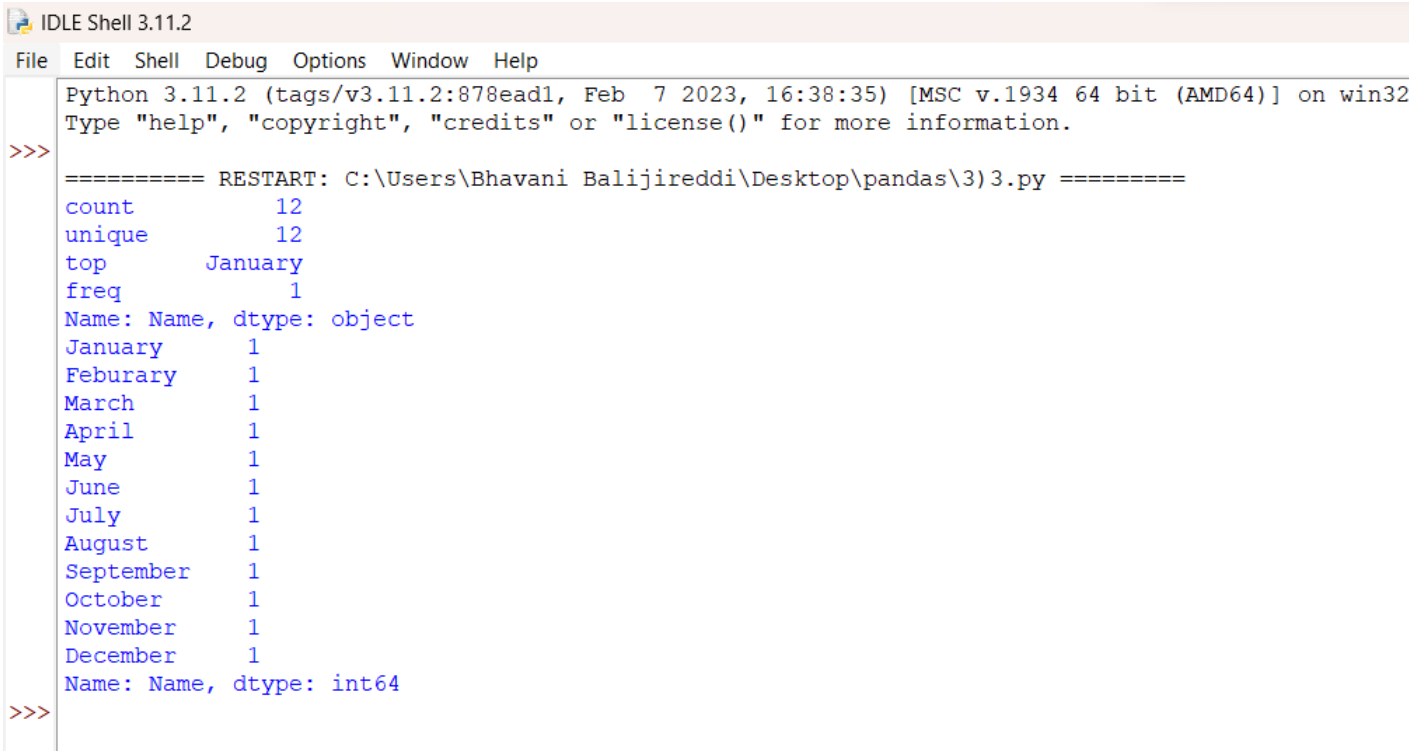
Program:

```
import pandas as pd
data = pd.read_csv('month.csv')
print(data['Name'].describe(include='Name'))
print(data['Name'].value_counts())
```

Expected output:



Observed output:

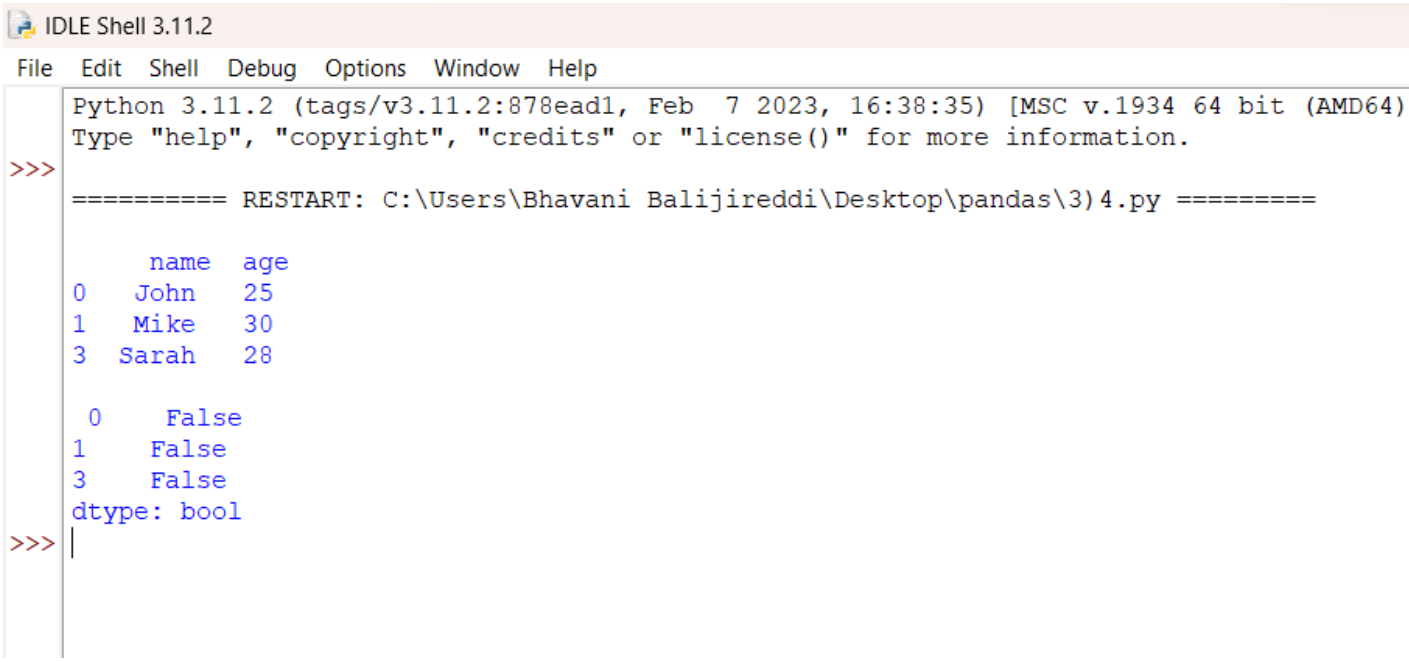


3.4 Handling duplicates

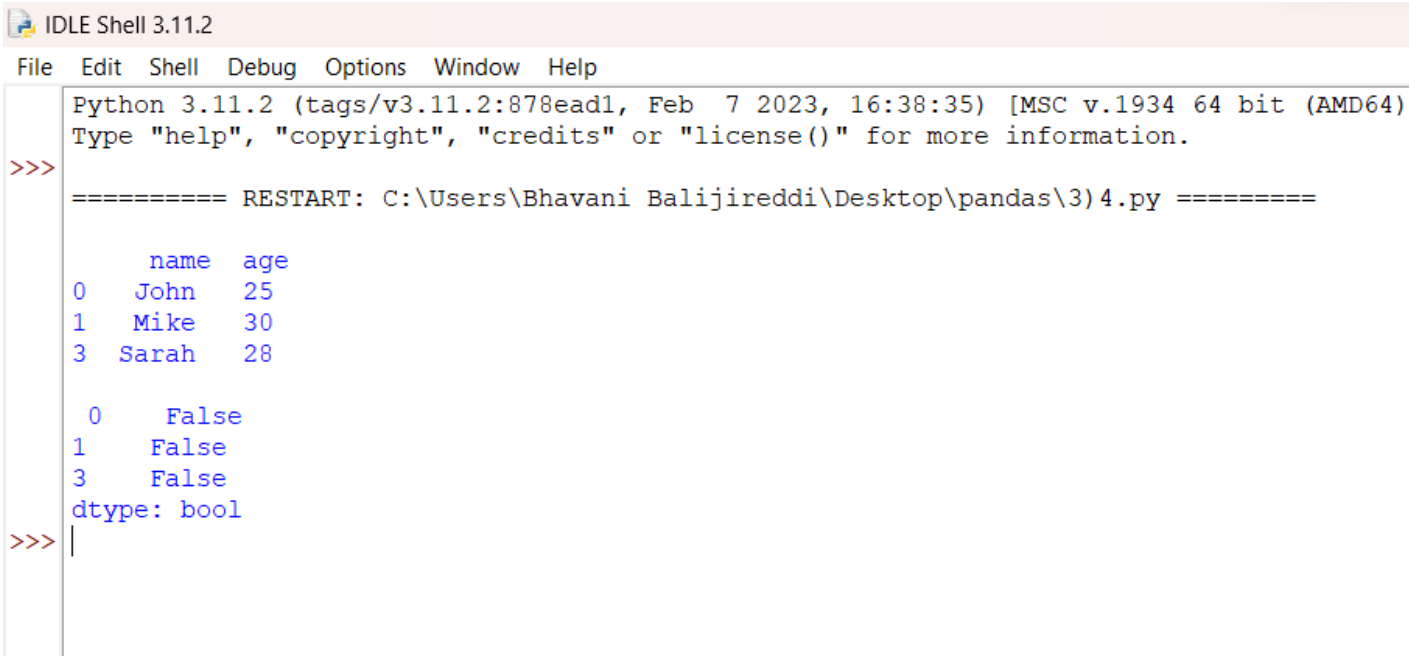
Program:

```
import pandas as pd data = {'name': ['John', 'Mike', 'John', 'Sarah', 'Mike'], 'age': [25, 30, 25, 28,
30]}
df = pd.DataFrame(data)
df = df.drop_duplicates()
print('\n',df)
duplicates = df.duplicated()
print('\n',duplicates)
```

Expected output:



Observed output:

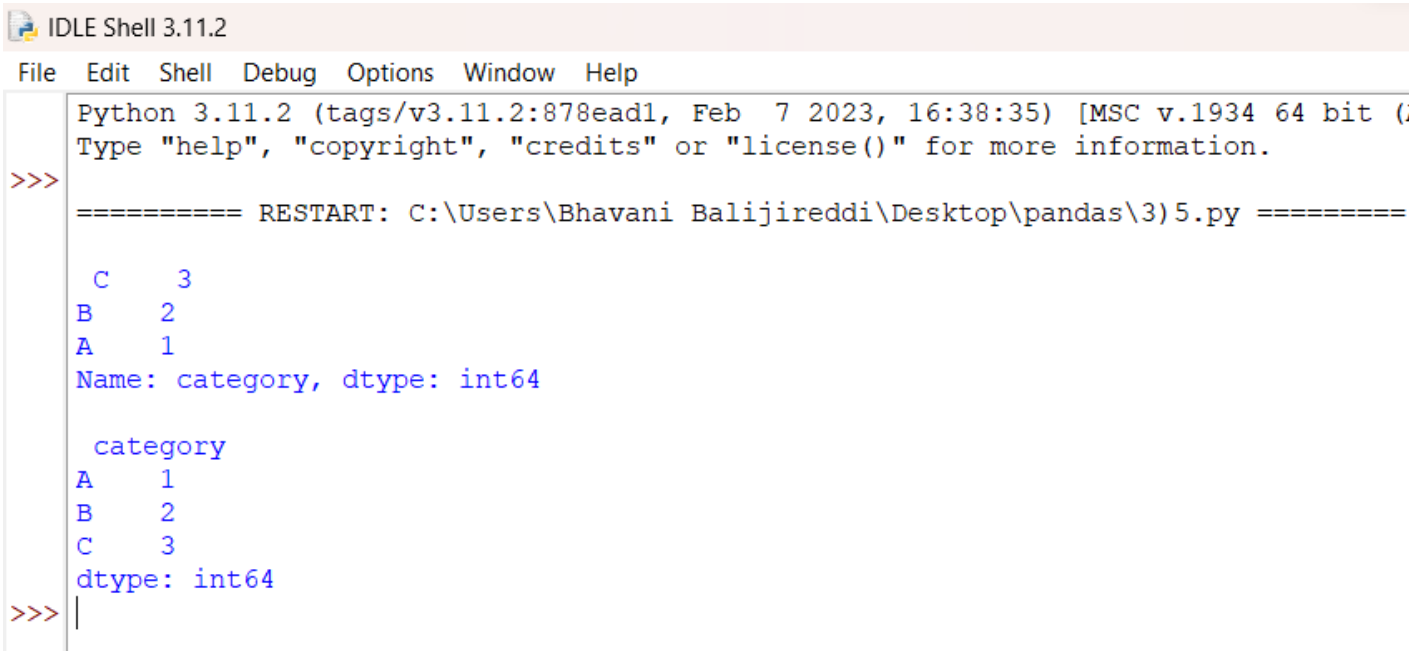


3.5 Number of observations Per Category

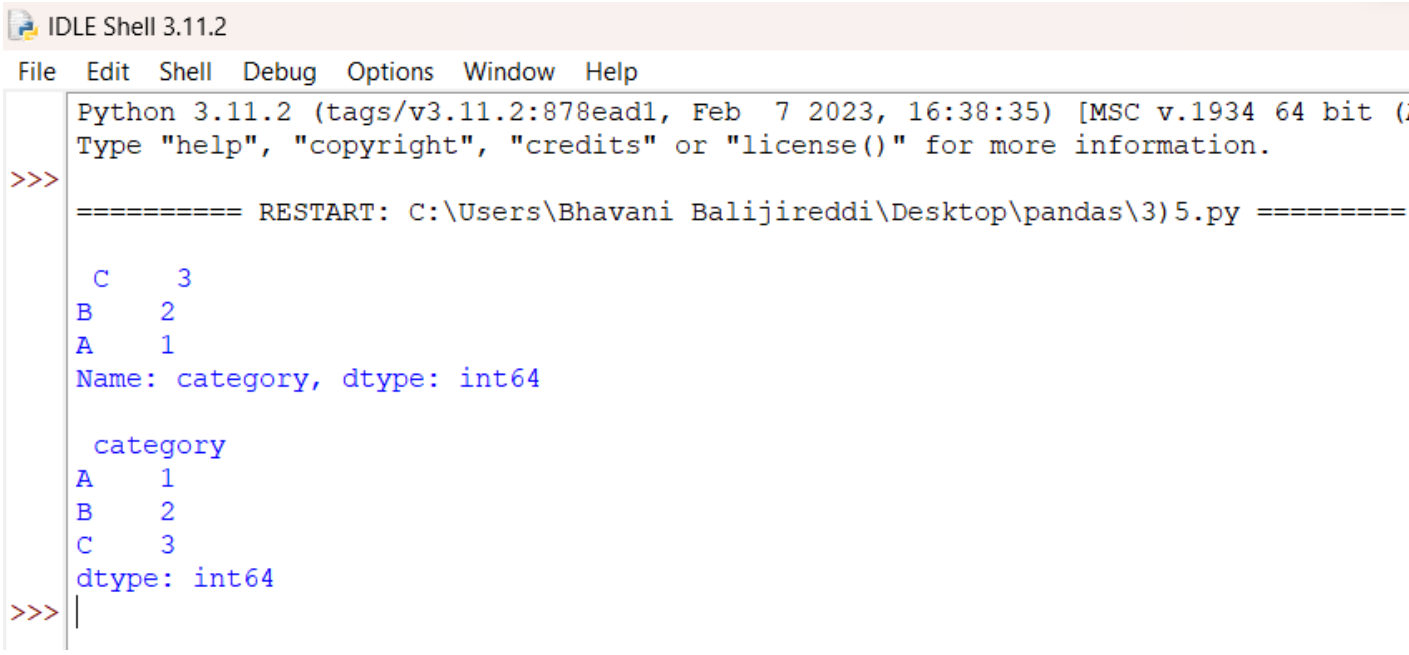
Program:

```
import pandas as pd
data = {'category': ['A', 'B', 'B', 'C', 'C', 'C']}
df = pd.DataFrame(data)
counts = df['category'].value_counts()
print('\n',counts)
counts = df.groupby('category').size()
print('\n',counts)
```

Expected output:



Observed output:

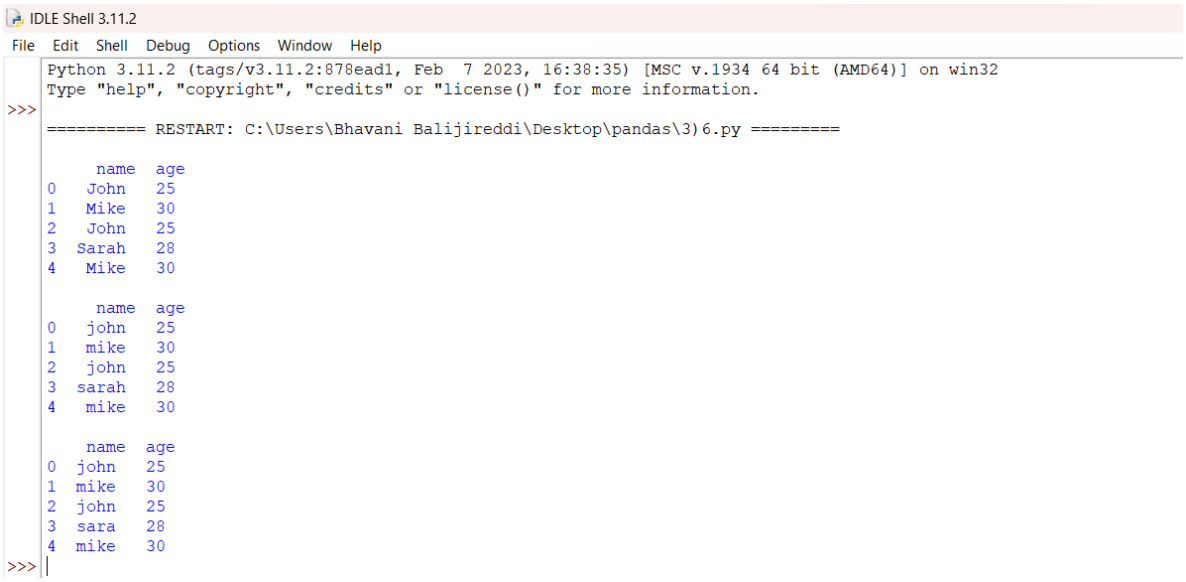


3.6 Column cleanup

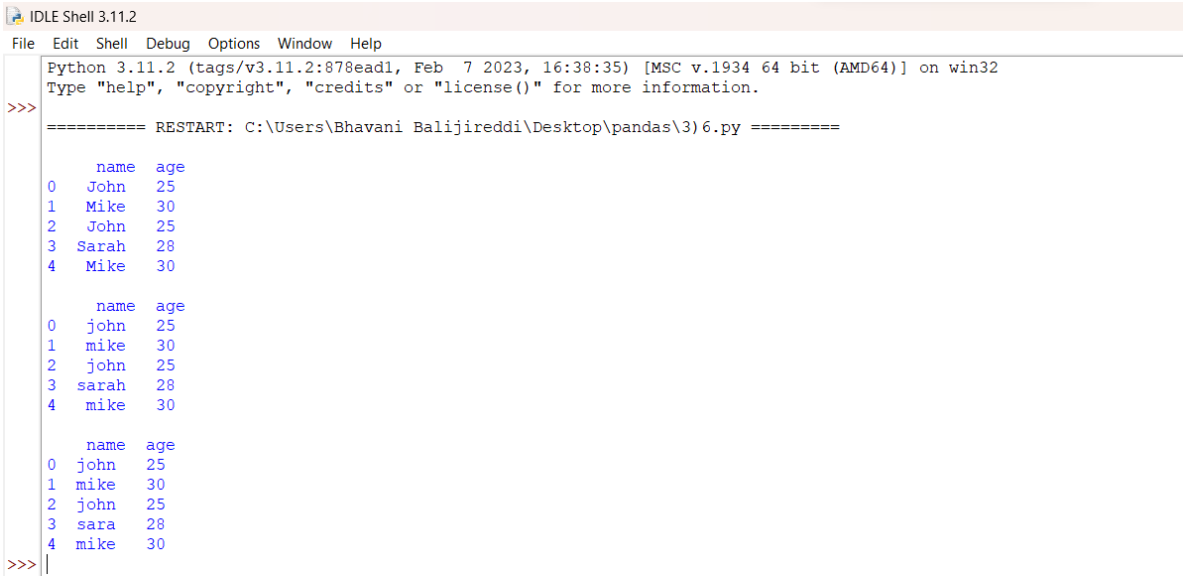
Program:

```
import pandas as pd
data = {'name': [' John', 'Mike ', ' John ', 'Sarah ', 'Mike '], 'age': [25, 30, 25, 28, 30]}
df = pd.DataFrame(data)
df['name'] = df['name'].str.strip()
print('\n',df)
df['name'] = df['name'].str.lower()
print('\n',df)
df['name'] = df['name'].replace('sarah', 'sara')
print('\n',df)
```

Expected output:



Observed output:



4 Getting the Dataset continuation

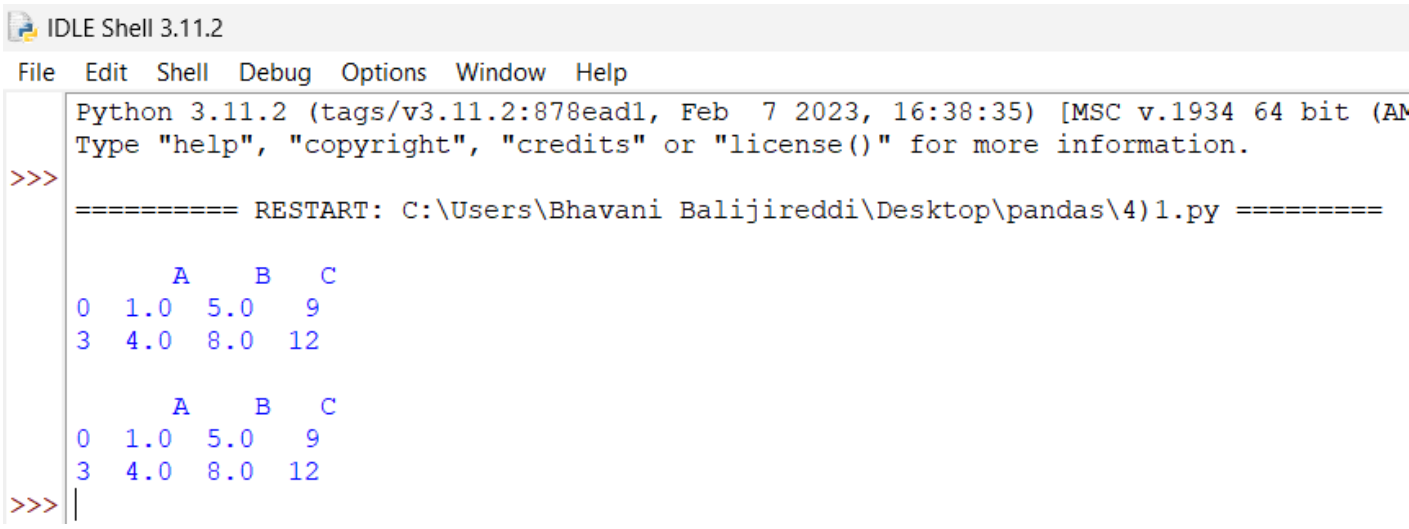
a. Removing null values b. Understanding your variables c. Relationships between continuous variables d. DataFrame slicing, selecting, extracting e. Conditional selections

4.1 Removing null values

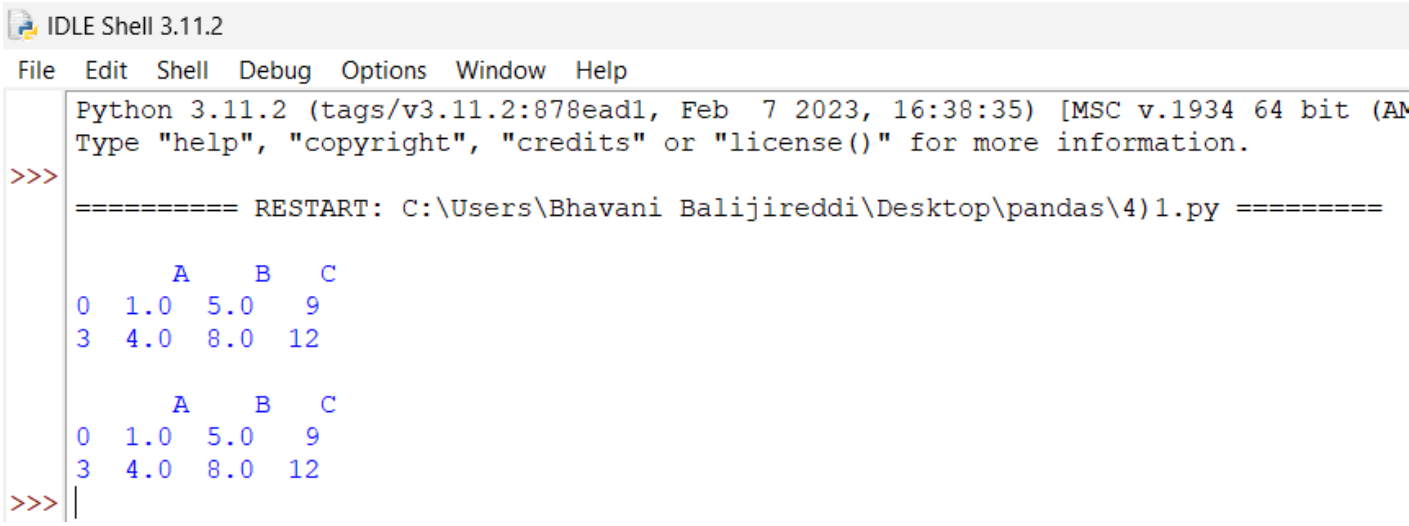
Program:

```
import pandas as pd
df = pd.DataFrame({'A': [1, 2, None, 4],
'B': [5, None, 7, 8],
'C': [9, 10, 11, 12]})
df = df.dropna()
print('\n',df)
df = df.dropna(axis=1)
print('\n',df)
```

Expected output:



Observed output:

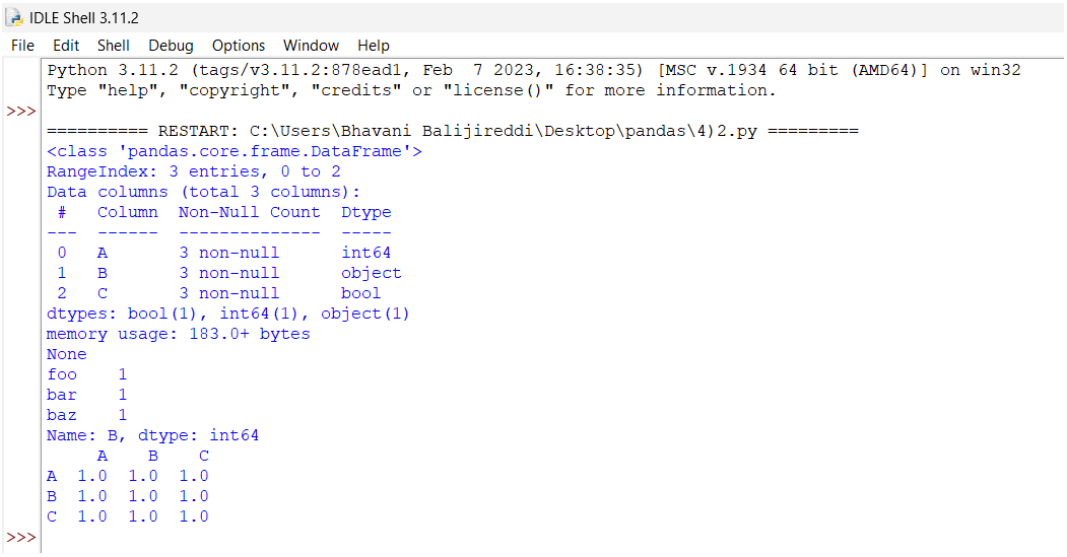


4.2 Understanding your variables

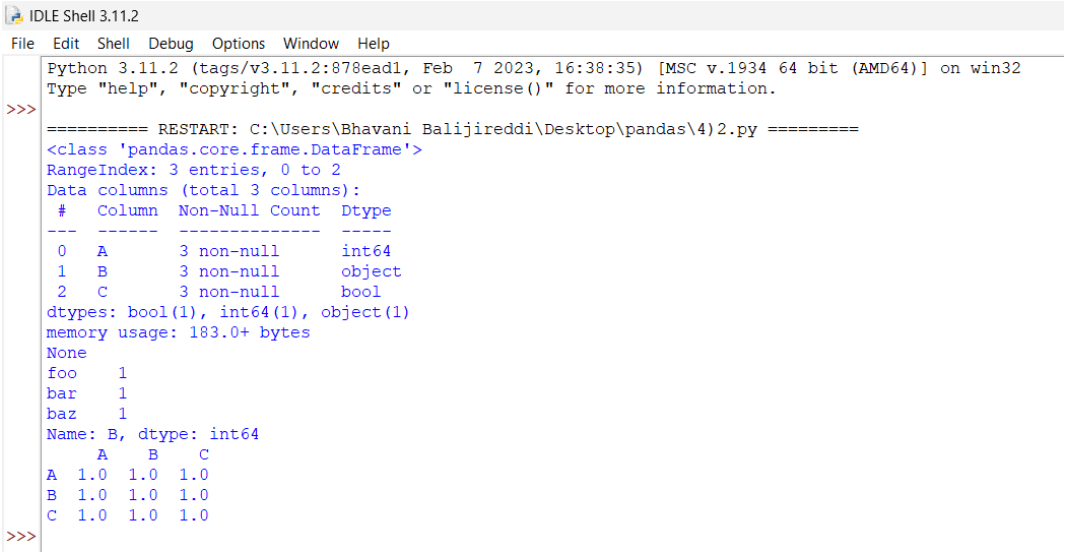
Program:

```
import pandas as pd
df = pd.DataFrame({'A': [1, 2, 3],
'B': ['foo', 'bar', 'baz'],
'C': [True, False, True]})
df.info()
print(df.info())
df['B'].value_counts()
print(df['B'].value_counts())
df = pd.DataFrame({'A': [1, 2, 3],
'B': [4, 5, 6],
'C': [7, 8, 9]})
print(df.corr())
```

Expected output:



Observed output:

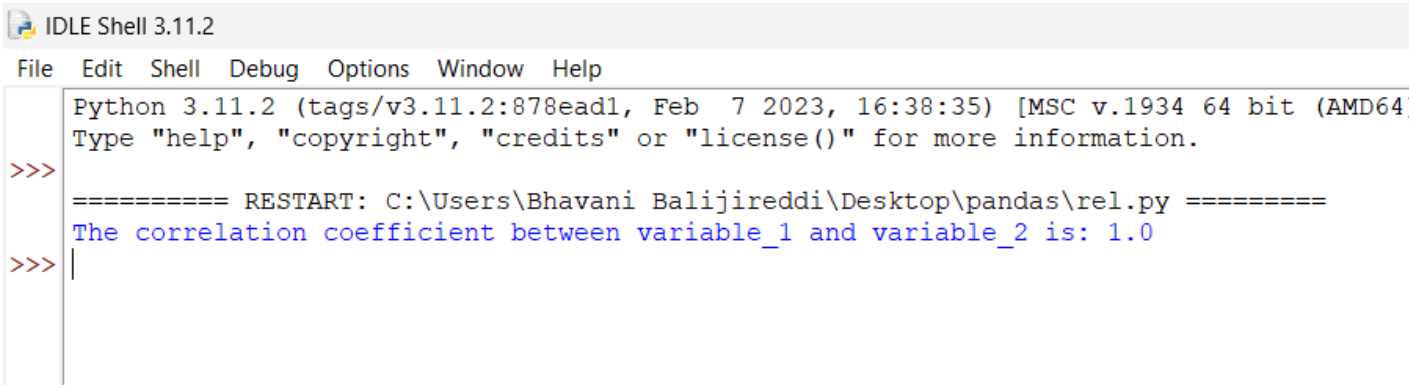


4.3 Relationships between continuous variables

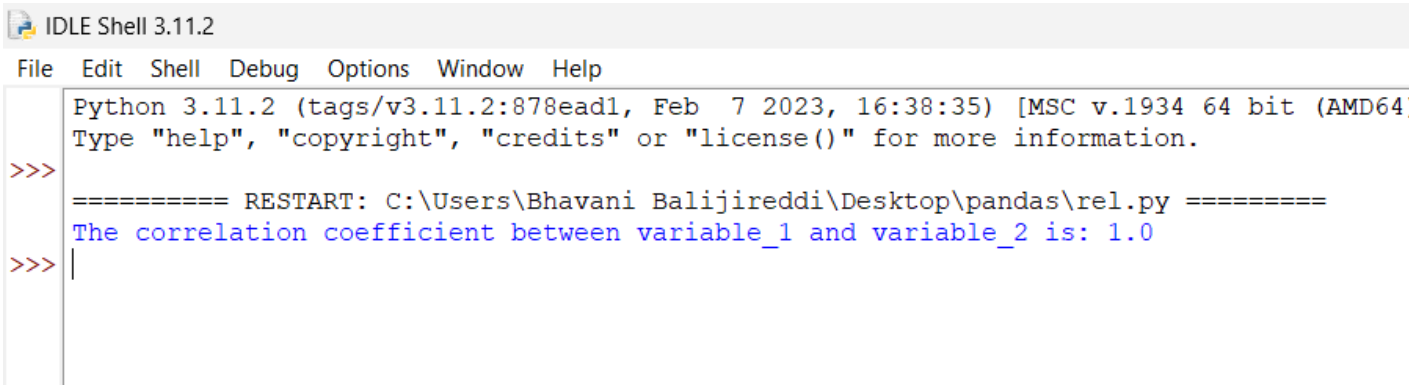
Program:

```
import pandas as pd
df = pd.DataFrame({'variable_1': [1, 2, 3, 4, 5],
'variable_2': [10, 15, 20, 25, 30]})
correlation_coefficient = df['variable_1'].corr(df['variable_2'])
print("The correlation coefficient between variable_1 and variable_2 is:", correlation_coefficient)
```

Expected output:



Observed output:

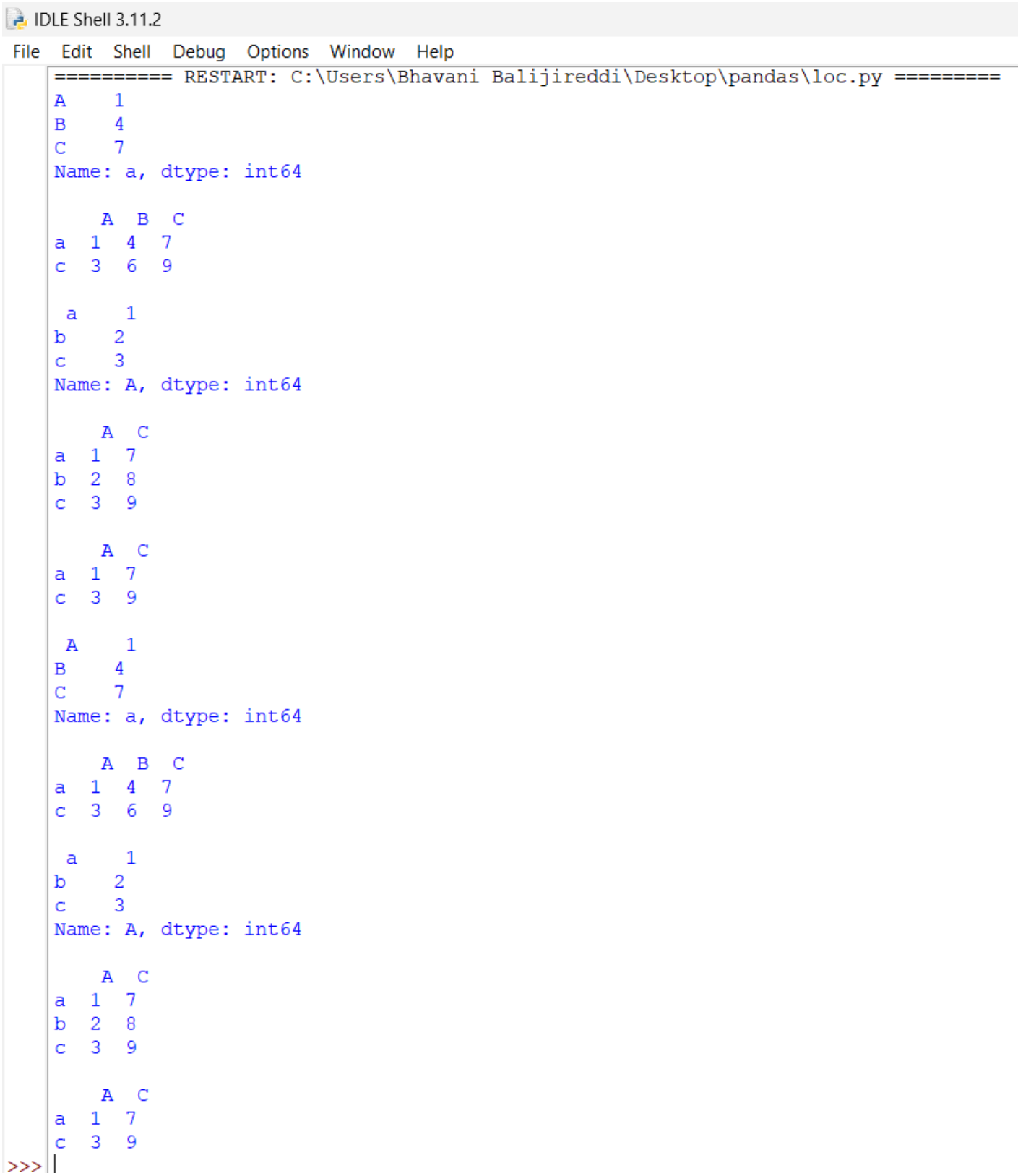


4.4 DataFrame slicing, selecting, extracting

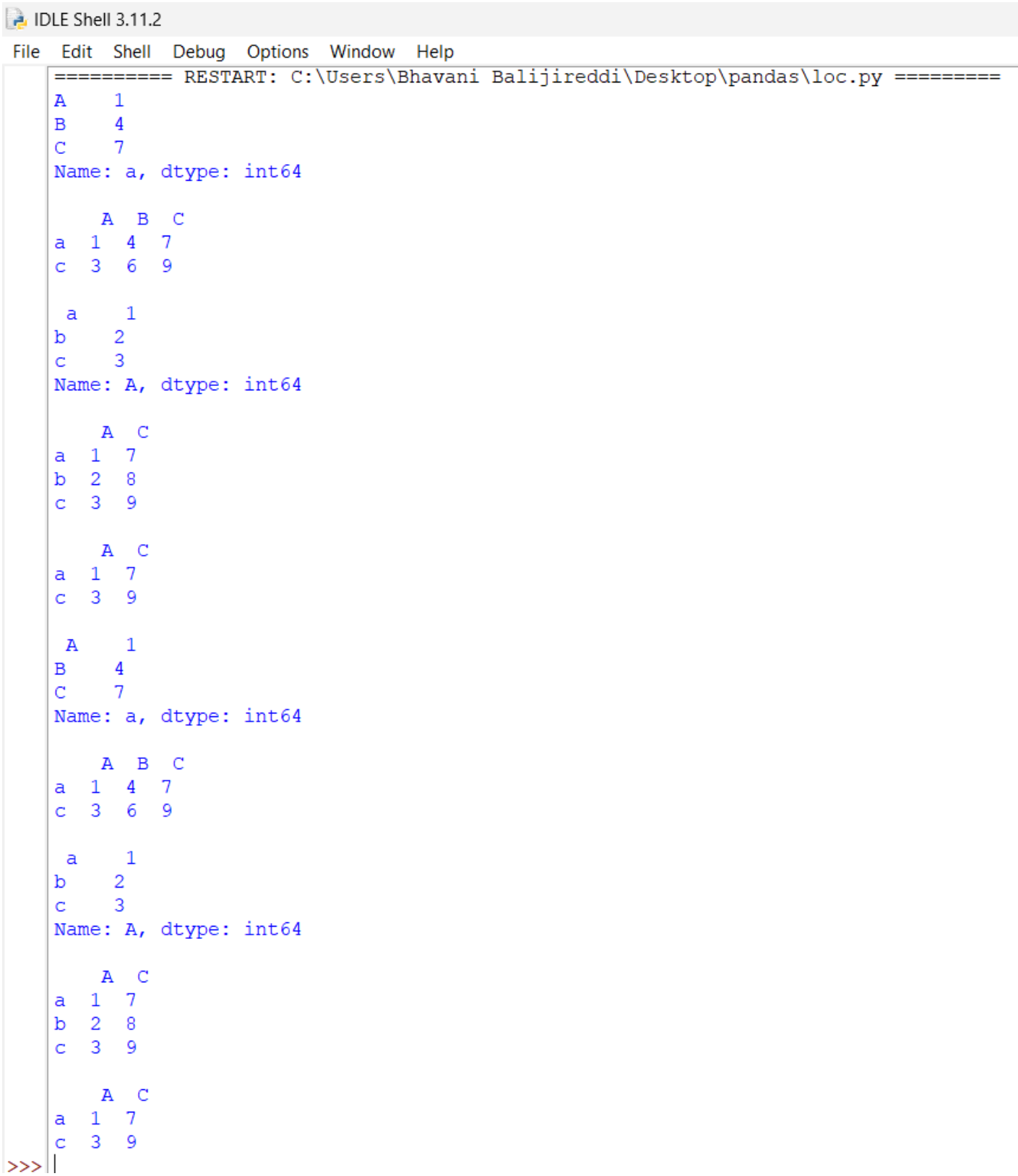
Program:

```
import pandas as pd
df = pd.DataFrame({'A': [1, 2, 3],
'B': [4, 5, 6],
'C': [7, 8, 9]},
index=['a', 'b', 'c'])
print(df.loc['a'])
print('\n',df.loc[['a', 'c']])
print('\n',df.loc[:, 'A'])
print('\n',df.loc[:, ['A', 'C']])
print('\n',df.loc[['a', 'c'], ['A', 'C']])
print('\n',df.iloc[0])
print('\n',df.iloc[[0, 2]])
print('\n',df.iloc[:, 0])
print('\n',df.iloc[:, [0, 2]])
print('\n',df.iloc[[0, 2], [0, 2]])
```

Expected output:



Observed output:

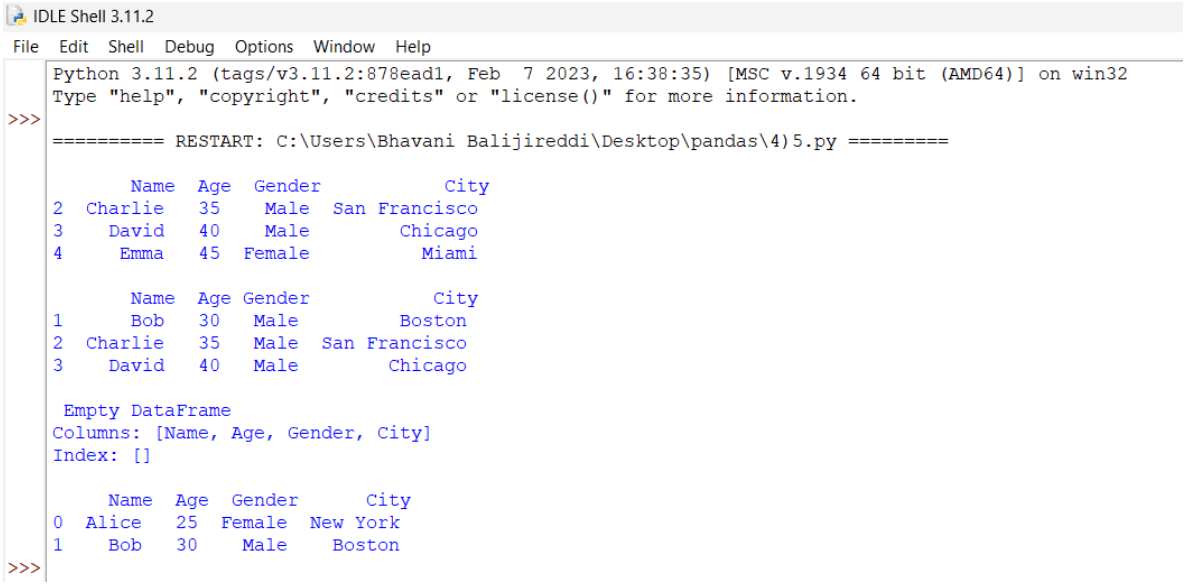


4.5 Conditional selections

Program:

```
import pandas as pd
df = pd.DataFrame({
'Name': ['Alice', 'Bob', 'Charlie', 'David', 'Emma'],
'Age': [25, 30, 35, 40, 45],
'Gender': ['Female', 'Male', 'Male', 'Male', 'Female'],
'City': ['New York', 'Boston', 'San Francisco', 'Chicago', 'Miami']
})
print('\n',df[df['Age'] ¿ 30])
print('\n',df.loc[df['Gender'] == 'Male'])
print('\n',df.query('Age ¿ 30 and City == "Boston"'))
print('\n',df[df['City'].isin(['New York', 'Boston'])])
```

Expected output:



Observed output: