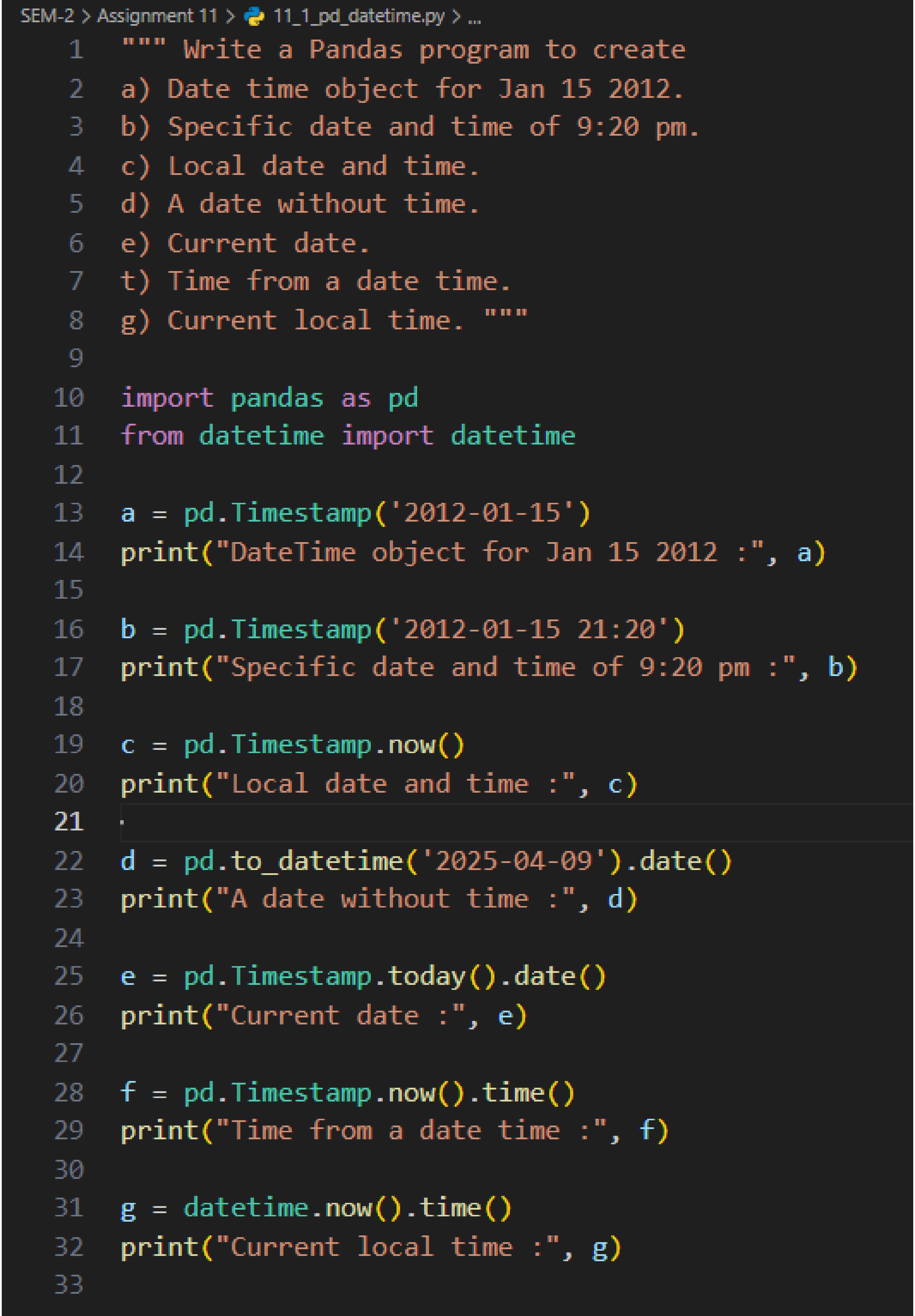
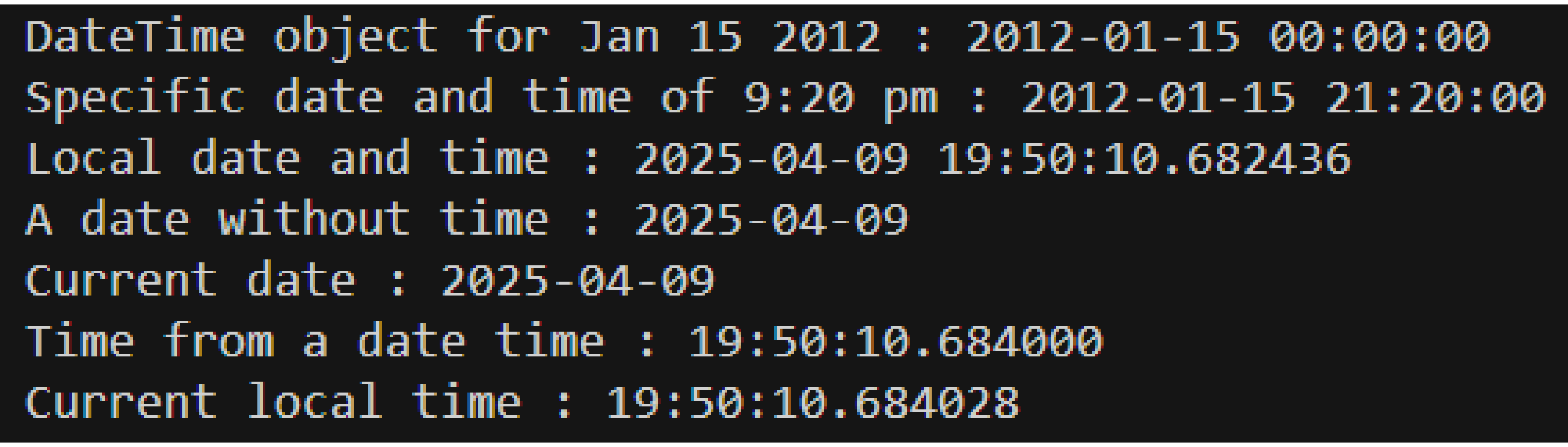
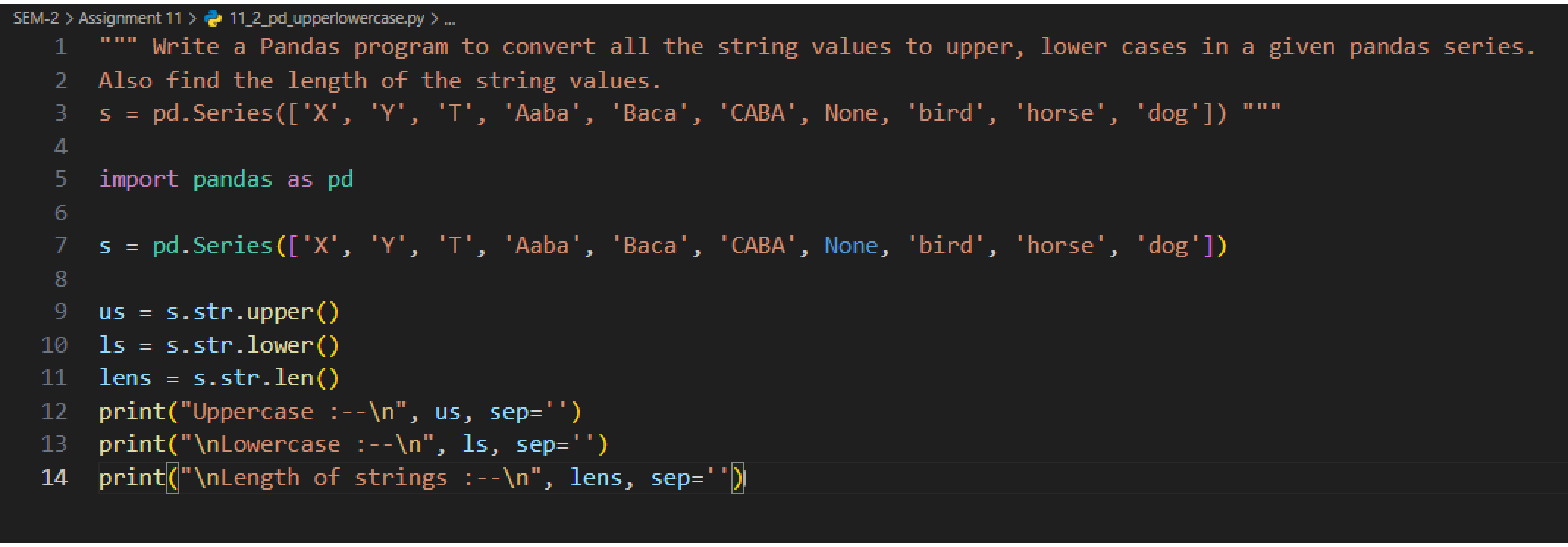


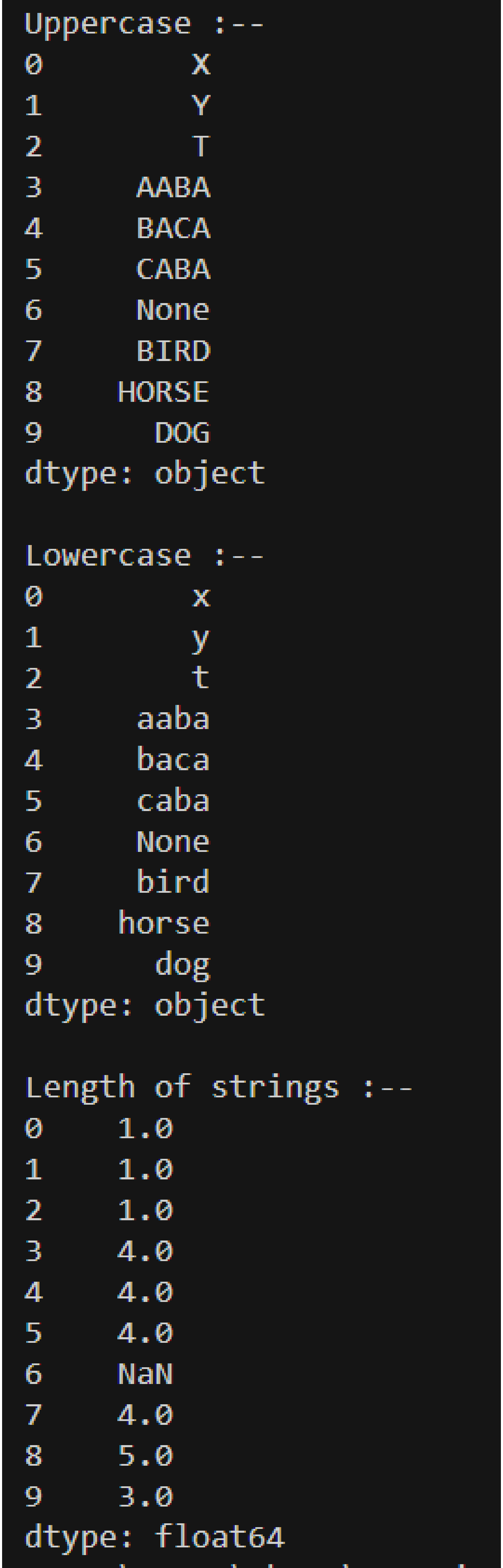
```python
""" Write a Pandas program to create
a) Date time object for Jan 15 2012.
b) Specific date and time of 9:20 pm.
c) Local date and time.
d) A date without time.
e) Current date.
t) Time from a date time.
g) Current local time. """

import pandas as pd
from datetime import datetime

a = pd.Timestamp('2012-01-15')
print("DateTime object for Jan 15 2012 :", a)

b = pd.Timestamp('2012-01-15 21:20')
print("Specific date and time of 9:20 pm :", b)

c = pd.Timestamp.now()
print("Local date and time :", c)

d = pd.to_datetime('2025-04-09').date()
print("A date without time :", d)

e = pd.Timestamp.today().date()
print("Current date :", e)

f = pd.Timestamp.now().time()
print("Time from a date time :", f)

g = datetime.now().time()
print("Current local time :", g)

```



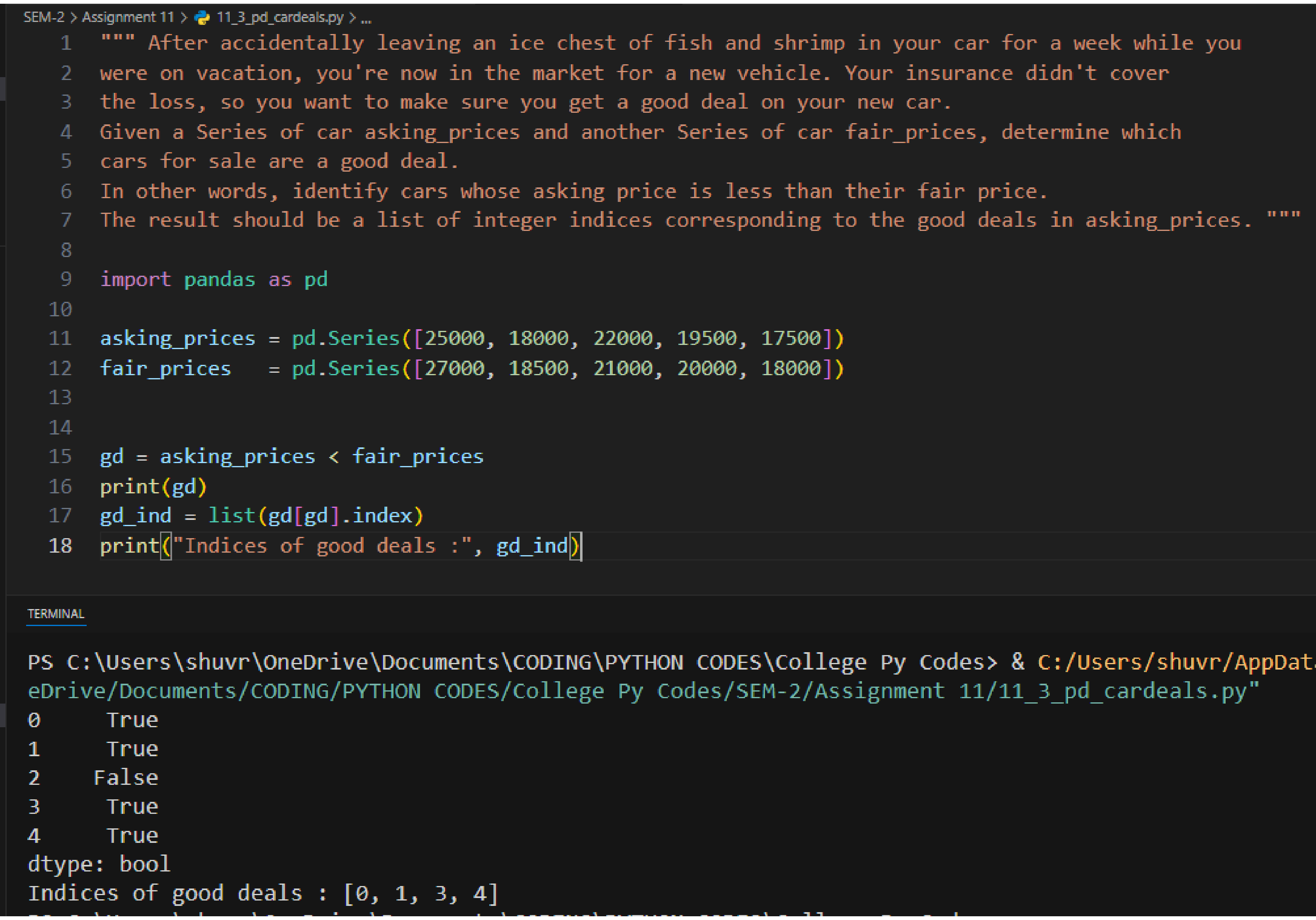
```
DateTime object for Jan 15 2012 : 2012-01-15 00:00:00
Specific date and time of 9:20 pm : 2012-01-15 21:20:00
Local date and time : 2025-04-09 19:50:10.682436
A date without time : 2025-04-09
Current date : 2025-04-09
Time from a date time : 19:50:10.684000
Current local time : 19:50:10.684028
```



```python
""" Write a Pandas program to convert all the string values to upper, lower cases in a given pandas series.
Also find the length of the string values.
s = pd.Series(['X', 'Y', 'T', 'Aaba', 'Baca', 'CABA', None, 'bird', 'horse', 'dog']) """

import pandas as pd

s = pd.Series(['X', 'Y', 'T', 'Aaba', 'Baca', 'CABA', None, 'bird', 'horse', 'dog'])

us = s.str.upper()
ls = s.str.lower()
lens = s.str.len()
print("Uppercase :--\n", us, sep='')
print("\nLowercase :--\n", ls, sep='')
print("\nLength of strings :--\n", lens, sep='')
```

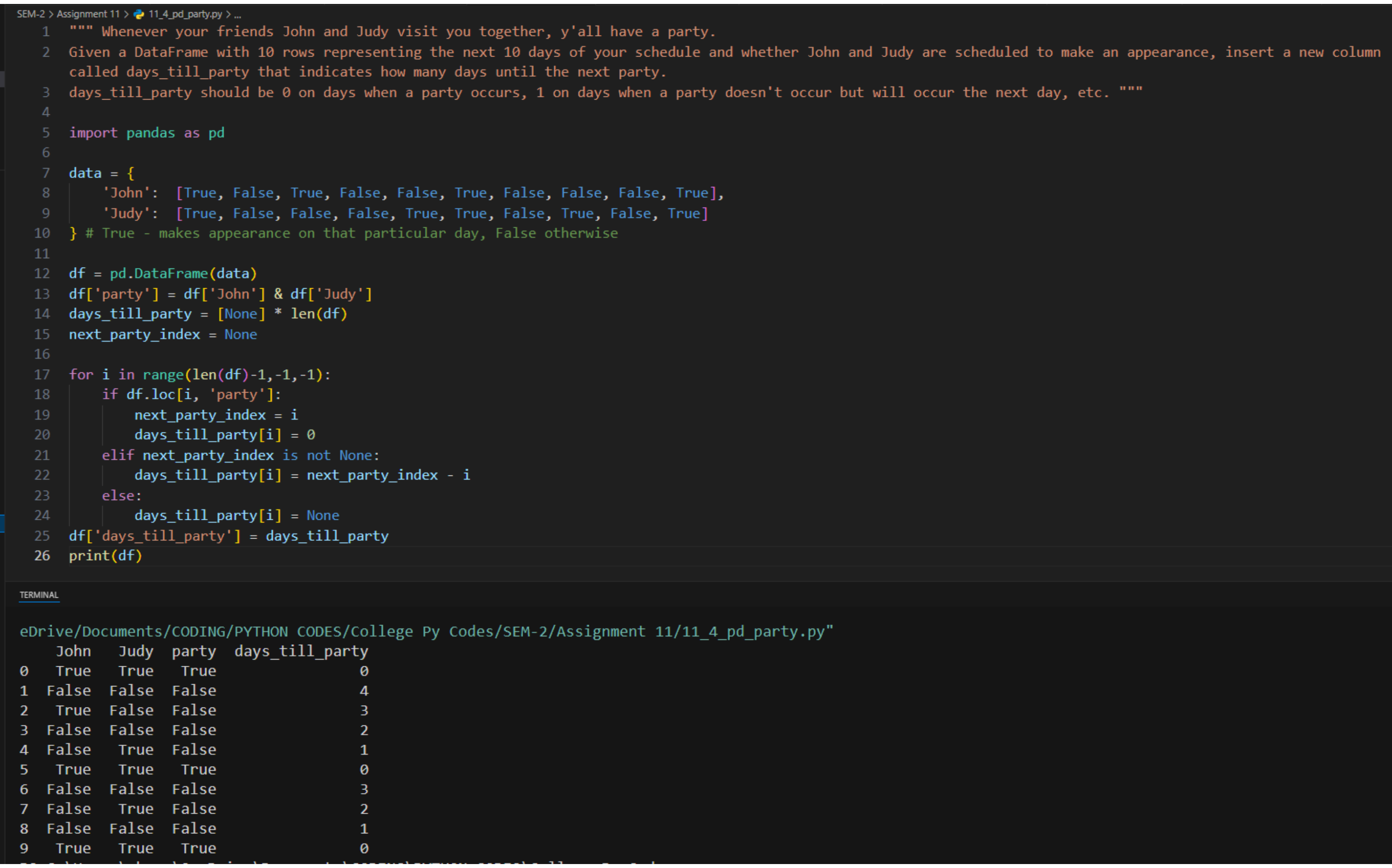


```
Uppercase :--
0         X
1         Y
2         T
3      AABA
4      BACA
5      CABA
6      None
7      BIRD
8     HORSE
9       DOG
dtype: object

Lowercase :--
0         x
1         y
2         t
3      aaba
4      baca
5      caba
6      None
7      bird
8     horse
9       dog
dtype: object

Length of strings :--
0    1.0
1    1.0
2    1.0
3    4.0
4    4.0
5    4.0
6    NaN
7    4.0
8    5.0
9    3.0
dtype: float64
```



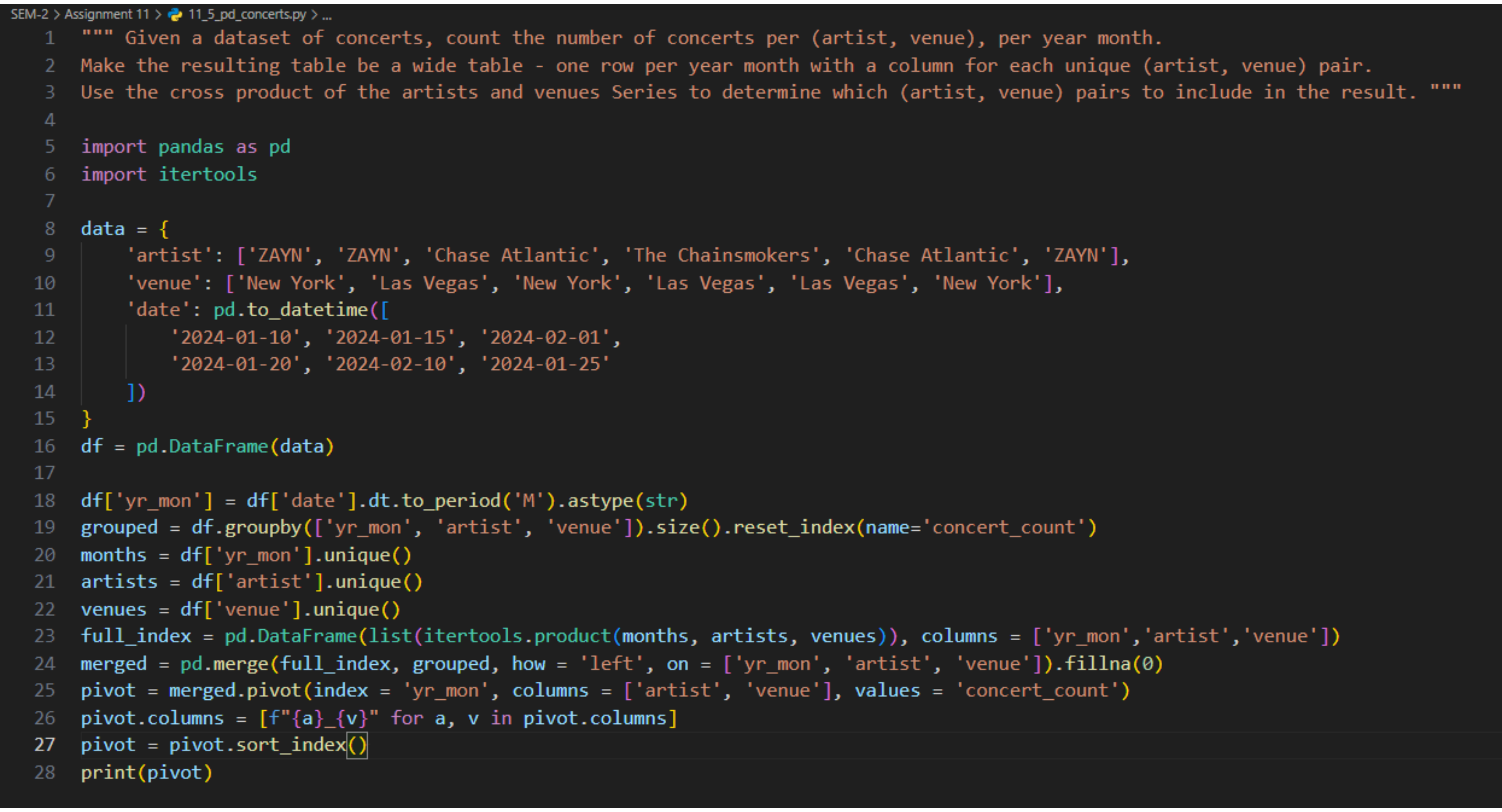
```python
""" After accidentally leaving an ice chest of fish and shrimp in your car for a week while you
were on vacation, you're now in the market for a new vehicle. Your insurance didn't cover
the loss, so you want to make sure you get a good deal on your new car.
Given a Series of car asking_prices and another Series of car fair_prices, determine which
cars for sale are a good deal.
In other words, identify cars whose asking price is less than their fair price.
The result should be a list of integer indices corresponding to the good deals in asking_prices. """

import pandas as pd

asking_prices = pd.Series([25000, 18000, 22000, 19500, 17500])
fair_prices   = pd.Series([27000, 18500, 21000, 20000, 18000])


gd = asking_prices < fair_prices
print(gd)
gd_ind = list(gd[gd].index)
print("Indices of good deals :", gd_ind)
```

```
PS C:\Users\shuvr\OneDrive\Documents\CODING\PYTHON CODES\College Py Codes> & C:/Users/shuvr/AppDat
eDrive/Documents/CODING/PYTHON CODES/College Py Codes/SEM-2/Assignment 11/11_3_pd_cardeals.py"
0     True
1     True
2    False
3     True
4     True
dtype: bool
Indices of good deals : [0, 1, 3, 4]
```



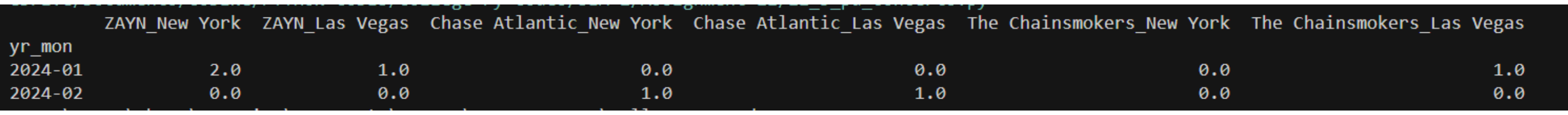
```python
""" Whenever your friends John and Judy visit you together, y'all have a party.
Given a DataFrame with 10 rows representing the next 10 days of your schedule and whether John and Judy are scheduled to make an appearance, insert a new column called days_till_party that indicates how many days until the next party.
days_till_party should be 0 on days when a party occurs, 1 on days when a party doesn't occur but will occur the next day, etc. """

import pandas as pd

data = {
    'John':  [True, False, True, False, False, True, False, False, False, True],
    'Judy':  [True, False, False, False, True, True, False, True, False, True]
} # True - makes appearance on that particular day, False otherwise

df = pd.DataFrame(data)
df['party'] = df['John'] & df['Judy']
days_till_party = [None] * len(df)
next_party_index = None

for i in range(len(df)-1,-1,-1):
    if df.loc[i, 'party']:
        next_party_index = i
        days_till_party[i] = 0
    elif next_party_index is not None:
        days_till_party[i] = next_party_index - i
    else:
        days_till_party[i] = None
df['days_till_party'] = days_till_party
print(df)
```

```
eDrive/Documents/CODING/PYTHON CODES/College Py Codes/SEM-2/Assignment 11/11_4_pd_party.py"
    John   Judy  party  days_till_party
0   True   True   True                0
1  False  False  False                4
2   True  False  False                3
3  False  False  False                2
4  False   True  False                1
5   True   True   True                0
6  False  False  False                3
7  False   True  False                2
8  False  False  False                1
9   True   True   True                0
```



SEM-2 > Assignment 11 > 11_5_pd_concerts.py > ...

```python
""" Given a dataset of concerts, count the number of concerts per (artist, venue), per year month.
Make the resulting table be a wide table - one row per year month with a column for each unique (artist, venue) pair.
Use the cross product of the artists and venues Series to determine which (artist, venue) pairs to include in the result. """

import pandas as pd
import itertools

data = {
    'artist': ['ZAYN', 'ZAYN', 'Chase Atlantic', 'The Chainsmokers', 'Chase Atlantic', 'ZAYN'],
    'venue': ['New York', 'Las Vegas', 'New York', 'Las Vegas', 'Las Vegas', 'New York'],
    'date': pd.to_datetime([
        '2024-01-10', '2024-01-15', '2024-02-01',
        '2024-01-20', '2024-02-10', '2024-01-25'
    ])
}
df = pd.DataFrame(data)

df['yr_mon'] = df['date'].dt.to_period('M').astype(str)
grouped = df.groupby(['yr_mon', 'artist', 'venue']).size().reset_index(name='concert_count')
months = df['yr_mon'].unique()
artists = df['artist'].unique()
venues = df['venue'].unique()
full_index = pd.DataFrame(list(itertools.product(months, artists, venues)), columns = ['yr_mon','artist','venue'])
merged = pd.merge(full_index, grouped, how = 'left', on = ['yr_mon', 'artist', 'venue']).fillna(0)
pivot = merged.pivot(index = 'yr_mon', columns = ['artist', 'venue'], values = 'concert_count')
pivot.columns = [f"{a}_{v}" for a, v in pivot.columns]
pivot = pivot.sort_index()
print(pivot)
```



```
         ZAYN_New York  ZAYN_Las Vegas  Chase Atlantic_New York  Chase Atlantic_Las Vegas  The Chainsmokers_New York  The Chainsmokers_Las Vegas
yr_mon
2024-01            2.0             1.0                      0.0                       0.0                        0.0                         1.0
2024-02            0.0             0.0                      1.0                       1.0                        0.0                         0.0
```