Pandas brain teasers

EXERCISE YOUR MIND

```python
from io import StringIO
import pandas as pd

csv_data = '''\
day,hits
2020-01-01,400
2020-02-02,800
2020-02-03,600
'''

df = pd.read_csv(StringIO(csv_data))
print(df['day'].dt.month.unique())
```

WILL THIS CODE RUN? WHAT WILL IT PRINT?

25 MIND BENDING TEASERS & SOLUTIONS

MIKI TEBEKA

Copyright

Off With Their NaNs

not_nan.py

```
1 import numpy as np
2 import pandas as pd
3
4 s = pd.Series([1, np.nan, 3])
5 print(s[~(s == np.nan)])
```

Try to guess what the output is before moving to the next page.

This code will will print

```
0    1.0
1    NaN
2    3.0
dtype: float64
```

We've covered some of the floating point oddities in [Multiplying]. `NaN` (or `np.nan`) is another oddity. The name `NaN` stands for "not a number", it serves two purposes - illegal computation and missing values.

Here's an example of a bad computation:

```
In [1]: np.float64(0)/np.float64(0)
<ipython-input-50-796728115601>:1: RuntimeWarning: invalid value
encountered in double_scalars
  np.float64(0)/np.float64(0)
Out[1]: nan
```

You see a warning but not an exception and the return value is `nan`.

`nan` does not equal any number, *including itself*.

```
In [2]: np.nan == np.nan
Out[2]: False
```

To check that a value is `nan`, you need to use a special function such as `pandas.isnull`.

```
In [3]: pd.isnull(np.nan)
Out[3]: True
```

You can use `pandas.isnull` to fix this teaser.

not_nan_fixed.py

```
1 import numpy as np
2 import pandas as pd
3
4 s = pd.Series([1, np.nan, 3])
5 print(s[~pd.isnull(s)])
```

`pandas.isnull` work with all of Pandas "missing" values: `None`, `pandas.NaT` (not a time) and the new

`pandas.NA`.

Floating points have several other special "numbers" such as `inf` (infinity), `-inf`, `-0`, `+0` and others. You can learn more about them in the links below.

Further Reading

- pandas.isnull documentation
- Experimental NA scalar to denote missing values in the Pandas documentation
- Floating Point Arithmetic: Issues and Limitations in the Python documentation
- floating point zine by Julia Evans
- What Every Computer Scientist Should Know About Floating-Point Arithmetic