

Extracted from:

Pandas Brain Teasers

Exercise Your Mind

This PDF file contains pages extracted from *Pandas Brain Teasers*, published by the Pragmatic Bookshelf. For more information or to purchase a paperback or PDF copy, please visit <http://www.pragprog.com>.

Note: This extract contains some colored text (particularly in code listing). This is available only in online versions of the books. The printed versions are black and white. Pagination might vary between the online and printed versions; the content is otherwise identical.

Copyright © 2021 The Pragmatic Programmers, LLC.

All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form, or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior consent of the publisher.

The Pragmatic Bookshelf

Raleigh, North Carolina

Pandas Brain Teasers

Exercise Your Mind



Miki Tebeka

edited by Margaret Eldridge

Pandas Brain Teasers

Exercise Your Mind

Miki Tebeka

The Pragmatic Bookshelf

Raleigh, North Carolina



Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and The Pragmatic Programmers, LLC was aware of a trademark claim, the designations have been printed in initial capital letters or in all capitals. The Pragmatic Starter Kit, The Pragmatic Programmer, Pragmatic Programming, Pragmatic Bookshelf, PragProg and the linking *g* device are trademarks of The Pragmatic Programmers, LLC.

Every precaution was taken in the preparation of this book. However, the publisher assumes no responsibility for errors or omissions, or for damages that may result from the use of information (including program listings) contained herein.

For our complete catalog of hands-on, practical, and Pragmatic content for software developers, please visit <https://pragprog.com>.

The team that produced this book includes:

CEO: Dave Rankin

COO: Janet Furlow

Managing Editor: Tammy Coron

Development Editor: Margaret Eldridge

Copy Editor: Jennifer Whipple

Indexing: Potomac Indexing, LLC

Layout: Gilson Graphics

Founders: Andy Hunt and Dave Thomas

For sales, volume licensing, and support, please contact support@pragprog.com.

For international rights, please contact rights@pragprog.com.

Copyright © 2021 The Pragmatic Programmers, LLC.

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form, or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior consent of the publisher.

ISBN-13: 978-1-68050-901-4

Encoded using the finest acid-free high-entropy binary digits.

Book version: P1.0—September 2021

To all the data nerds out there, you rock!

Puzzle 1

Rectified

relu.py

```
import pandas as pd
```

```
def relu(n):  
    if n < 0:  
        return 0  
    return n
```

```
arr = pd.Series([-1, 0, 1])  
print(relu(arr))
```

Guess the Output



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

This code will raise a `ValueError`.

The problematic line is `if n < 0:`. `n` is the result of `arr < 0`, which is a `pandas.Series`.

```
In [1]: import pandas as pd
In [2]: arr = pd.Series([-1, 0, 1])
In [3]: arr < 0
Out[3]:
0      True
1     False
2     False
dtype: bool
```

Once `arr < 0` is computed, you use it in an `if` statement, which brings us to how Boolean values work in Python.

Every Python object, not just `True` and `False`, has a Boolean value. The documentation states the rules:

Everything is `True` except

- 0 numbers: 0, 0.0, 0+0j, ...
- Empty collections: [], {}, "", ...
- `None`
- `False`

You can test the truth value of a Python object using the built-in `bool` function.

In addition to these rules, any object can state its own Boolean value using the `__bool__` special method. The Boolean logic for a `pandas.Series` is different from the one for a list or a tuple; it raises an exception.

```
In [4]: bool(arr < 0)
...
ValueError: The truth value of a Series is ambiguous.
Use a.empty, a.bool(), a.item(), a.any() or a.all().
```

The exception tells you the reasoning. It follows “The Zen of Python,” which states the following:

In the face of ambiguity, refuse the temptation to guess.

So what are your options? You can use `all` or `any` but then you’ll need to check the type of `n` to see if it’s a plain number or a `pandas.Series`.

A function that works both on scalar and a `pandas.Series` (or a `numpy` array) is called a `ufunc`, short for *universal function*. Most of the functions from `numpy` or `Pandas`, such as `min` or `to_datetime`, are `ufuncs`.

numpy has a vectorize decorator for these cases.

```
relu_vec.py
import numpy as np
import pandas as pd
```

```
@np.vectorize
def relu(n):
    if n < 0:
        return 0
    return n

arr = pd.Series([-1, 0, 1])
print(relu(arr))
```

Now, relu will work both on scalars (e.g., 7, 2.18, ...) and vectors (e.g., numpy array, pandas.Series, ...)

Watch Your Types



The output of relu now is numpy.ndarray, not pandas.Series as well.

Further Reading

Truth Value Testing in the Python Documentation

docs.python.org/3/library/stdtypes.html#truth-value-testing

PEP 285

python.org/dev/peps/pep-0285/

bool Type Documentation

docs.python.org/3/reference/datamodel.html#object._bool_

Universal Functions on the numpy Docs

numpy.org/doc/stable/reference/ufuncs.html?highlight=ufunc

“The Zen of Python”

python.org/dev/peps/pep-0020/#the-zen-of-python

numpy.vectorize

numpy.org/doc/stable/reference/generated/numpy.vectorize.html#numpy.vectorize

numba.vectorize

numba.pydata.org/numba-doc/latest/user/vectorize.html