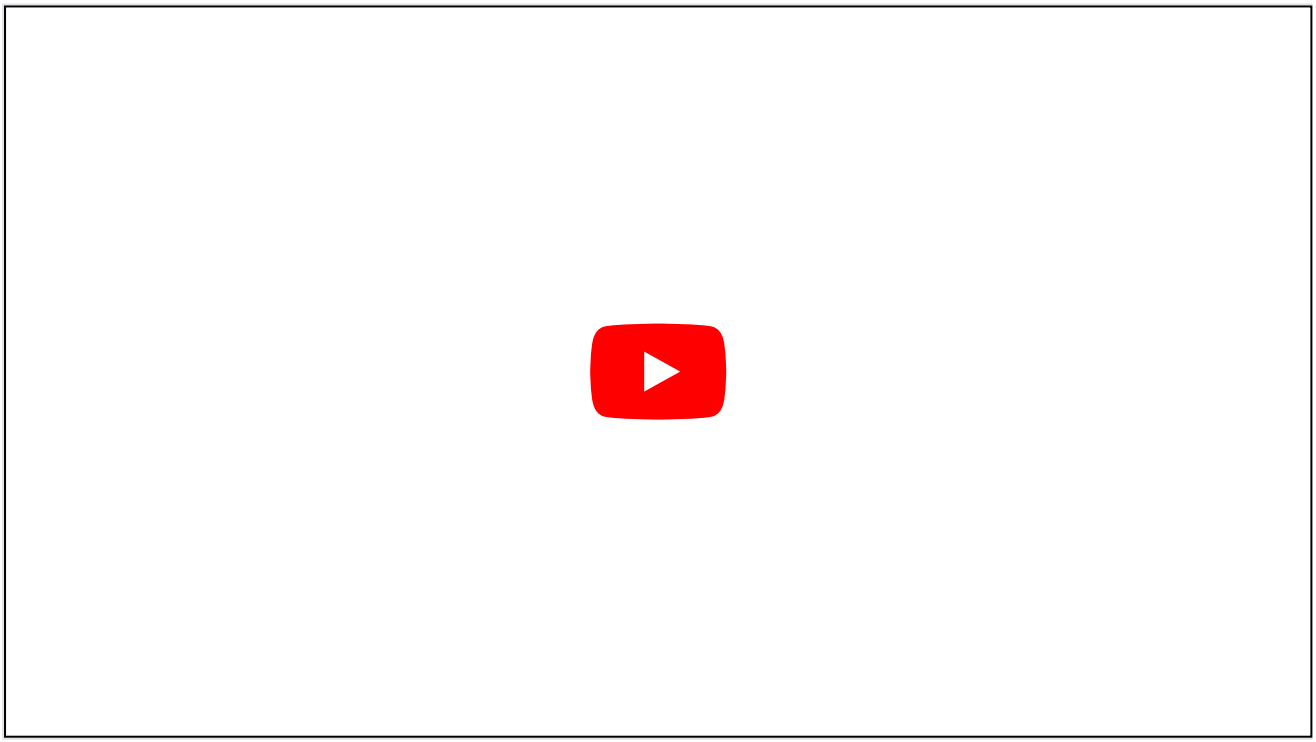
CS50's Introduction to Programming with Python

OpenCourseWare

Donate (https://cs50.harvard.edu/donate)

David J. Malan (https://cs.harvard.edu/malan/)
malan@harvard.edu
(https://www.facebook.com/dmalan) (https://github.com/dmalan)
(https://www.instagram.com/davidjmalan/) (https://www.linkedin.com/in/malan/)
(https://www.reddit.com/user/davidjmalan)
(https://www.threads.net/@davidjmalan) (https://twitter.com/davidjmalan)

Re-requesting a Vanity Plate



In a file called `plates.py`, reimplement Vanity Plates from Problem Set 2, restructuring your code per the below, wherein `is_valid` still expects a `str` as input and returns `True` if that `str` meets all requirements and `False` if it does not, but `main` is only called if the value of `__name__` is `"__main__"`:

```
def main():
    ...


def is_valid(s):
    ...


if __name__ == "__main__":
    main()
```

Then, in a file called `test_plates.py`, implement **four or more** functions that collectively test your implementation of `is_valid` thoroughly, each of whose names should begin with `test_` so that you can execute your tests with:

```
pytest test_plates.py
```

▼ **Hints**

- Be sure to include

  ```
  import plates
  ```

  or

  ```
  from plates import is_valid
  ```

  atop `test_plates.py` so that you can call `is_valid` in your tests.
- Take care to `return`, not `print`, a `bool` in `is_valid`. Only `main` should call `print`.

Before You Begin

Log into cs50.dev (https://cs50.dev/), click on your terminal window, and execute `cd` by itself. You should find that your terminal window's prompt resembles the below:

```
$
```

Next execute

```
mkdir test_plates
```

to make a folder called `test_plates` in your codespace.

Then execute

```
cd test_plates
```

to change directories into that folder. You should now see your terminal prompt as `test_plates/ $`. You can now execute

```
code test_plates.py
```

to make a file called `test_plates.py` where you'll write your tests.

How to Test

To test your tests, run `pytest test_plates.py`. Be sure you have a copy of a `plates.py` file in the same folder. Try to use correct and incorrect versions of `plates.py` to determine how well your tests spot errors:

- Ensure you have a correct version of `plates.py`. Run your tests by executing `pytest test_plates.py`. `pytest` should show that all of your tests have passed.
- Modify the correct version of `plates.py`, perhaps eliminating some of its constraints. Your program might, for example, mistakenly print "Valid" for a license plate of any length! Run your tests by executing `pytest test_plates.py`. `pytest` should show that at least one of your tests has failed.

You can execute the below to check your tests using `check50`, a program CS50 will use to test your code when you submit. (Now there are tests to test your tests!). Be sure to test your tests yourself and determine which tests are needed to ensure `plates.py` is checked thoroughly.

```
check50 cs50/problems/2022/python/tests/plates
```

Green smilies mean your program has passed a test! Red frownies will indicate your program output something unexpected. Visit the URL that `check50` outputs to see the input `check50` handed to your program, what output it expected, and what output your program actually gave.

How to Submit

In your terminal, execute the below to submit your work.

```
submit50 cs50/problems/2022/python/tests/plates
```