








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>)

Vanity Plates



In Massachusetts, home to Harvard University, it's possible to [request a vanity license plate](https://www.mass.gov/how-to/request-a-vanity-license-plate) (<https://www.mass.gov/how-to/request-a-vanity-license-plate>) for your car, with your choice of letters and numbers instead of random ones. Among the requirements, though, are:

- “All vanity plates must start with at least two letters.”
- “... vanity plates may contain a maximum of 6 characters (letters or numbers) and a minimum of 2 characters.”
- “Numbers cannot be used in the middle of a plate; they must come at the end. For example, AAA222 would be an acceptable ... vanity plate; AAA22A would not be acceptable. The first number used cannot be a ‘0’”

- “No periods, spaces, or punctuation marks are allowed.”

In `plates.py`, implement a program that prompts the user for a vanity plate and then output `Valid` if it meets all of the requirements or `Invalid` if it does not. Assume that any letters in the user's input will be uppercase. Structure your program per the below, wherein `is_valid` returns `True` if `s` meets all requirements and `False` if it does not. Assume that `s` will be a `str`. You're welcome to implement additional functions for `is_valid` to call (e.g., one function per requirement).

```
def main():
    plate = input("Plate: ")
    if is_valid(plate):
        print("Valid")
    else:
        print("Invalid")

def is_valid(s):
    ...

main()
```

► Hints

Demo



Recorded with [asciinema](#)

Before You Begin

Log into [cs50.dev \(https://cs50.dev/\)](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 plates
```

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

Then execute

```
cd plates
```

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

```
code plates.py
```

to make a file called `plates.py` where you'll write your program.

How to Test

Here's how to test your code manually:

- Run your program with `python plates.py`. Type `CS50` and press Enter. Your program should output:

```
Valid
```

- Run your program with `python plates.py`. Type `CS05` and press Enter. Your program should output:

```
Invalid
```

- Run your program with `python plates.py`. Type `CS50P` and press Enter. Your program should output

```
Invalid
```

- Run your program with `python plates.py`. Type `PI3.14` and press Enter. Your program should output

```
Invalid
```

- Run your program with `python plates.py`. Type `H` and press Enter. Your program should output

```
Invalid
```

- Run your program with `python plates.py`. Type `OUTATIME` and press Enter. Your program should output

```
Invalid
```

You can execute the below to check your code using `check50`, a program that CS50 will use to test your code when you submit. But be sure to test it yourself as well!

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

