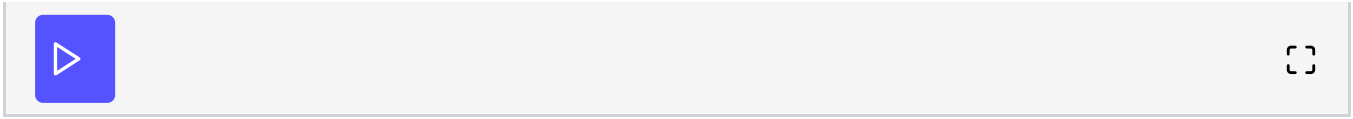
Running Scripts

Python modules are objects and have several useful attributes. You can use this to easily test your modules as you write them, by including a special block of code that executes when you run the Python file on the command line. Take the last few lines of `humansize.py`:

```
from humansize import approximate_size

if __name__ == '__main__':
    print(approximate_size(1000000000000, False))
    print(approximate_size(1000000000000))
```

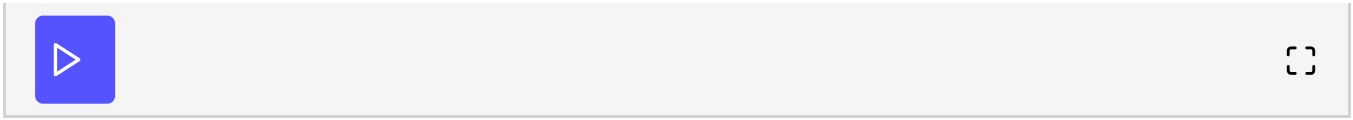


Everything in Python is an object.

>*Like c, Python uses == for comparison and = for assignment. Unlike c, Python does not support in-line assignment, so there's no chance of accidentally assigning the value you thought you were comparing.*

So what makes this `if` statement special? Well, modules are objects, and all modules have a built-in attribute `__name__`. A module's `__name__` depends on how you're using the module. If you `import` the module, then `__name__` is the module's filename, without a directory path or file extension.

```
import humansize
print(humansize.__name__)
#humansize
```



But you can also run the module directly as a standalone program, in which case `__name__` will be a special default value `__main__`. Python will evaluate

case `__name__` will be a special default value, `__main__`. Python will evaluate

this `if` statement, find a true expression, and execute the `if` code block. In this case, to print two values.

```
1.0 TB
931.3 GiB
```

And that's your first Python program!