

Python Program Execution Flow

CPE 101

Winter 2019 @ CP SLO

By

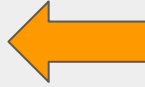
Toshi

Basic Program Execution Flow

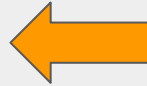
- Top Down
 - Expressions will be evaluated.
 - Statements will be executed.
 - Except for function and class definitions.
- Definitions
 - Will be read and compiled but code will not be evaluated / executed until function/class method calls are made.

Examples

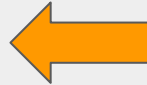
```
print("Let's start!")
```



```
def foo():  
    return "foo"
```



```
def bar():  
    return "bar"
```



```
foo()  
bar()  
print("Let's end here!")
```

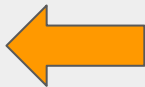


Examples

```
def foo():  
    return "foo"
```

```
def bar():  
    return "bar"
```

```
print("Let's end here!")
```



Examples

```
def foo():  
    return "foo"
```

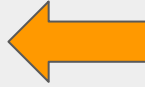
```
def bar():  
    return "bar"
```

`__main__` — Top-level script environment

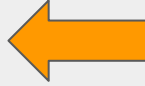
- `'__main__'` is the name of the scope in which top-level code executes.
- A module's `__name__` is set equal to `'__main__'` when read as top-level code (not read as imports from other programs).
- A module can discover whether or not it is running in the main scope by checking its own `__name__`

Examples

```
def foo():  
    return "foo"
```



```
def bar():  
    return "bar"
```



```
if __name__ == "__main__":  
    # execute only if run as a script  
    foo()  
    bar()
```



Examples

test_foo_bar.py

```
from foo_bar import foo
from foo_bar import bar
```

```
assert foo() == "foo"
assert bar() == "bar"
```

foo_bar.py

```
def foo():
    return "foo"
```

```
def bar():
    return "bar"
```

```
if __name__ == "__main__":
    # execute only if run as a script
    foo()
    bar()
```