How Python Works

WARNING: This section covers the current version of the reference implementation of python called CPython. There are other implementations of the python language that do not work this way and are more performant. However, people often refer to the reference implementation when they say "python".

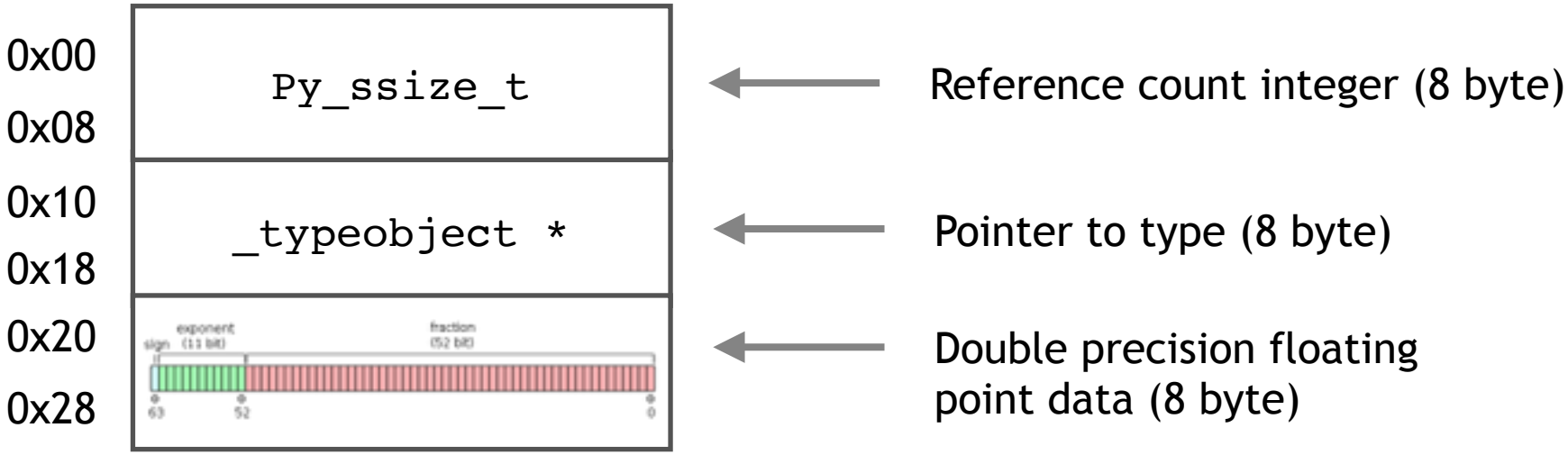
Python Memory Management

ACCRE
Advanced Computing Center for Research & Education

- Each python object has a reference count
- Binding to a name, placement in a list position, increments the reference count
- When the reference count is zero, the object is destroyed and memory freed
- A garbage collector runs periodically to take care of circular references.

Python Objects in Memory

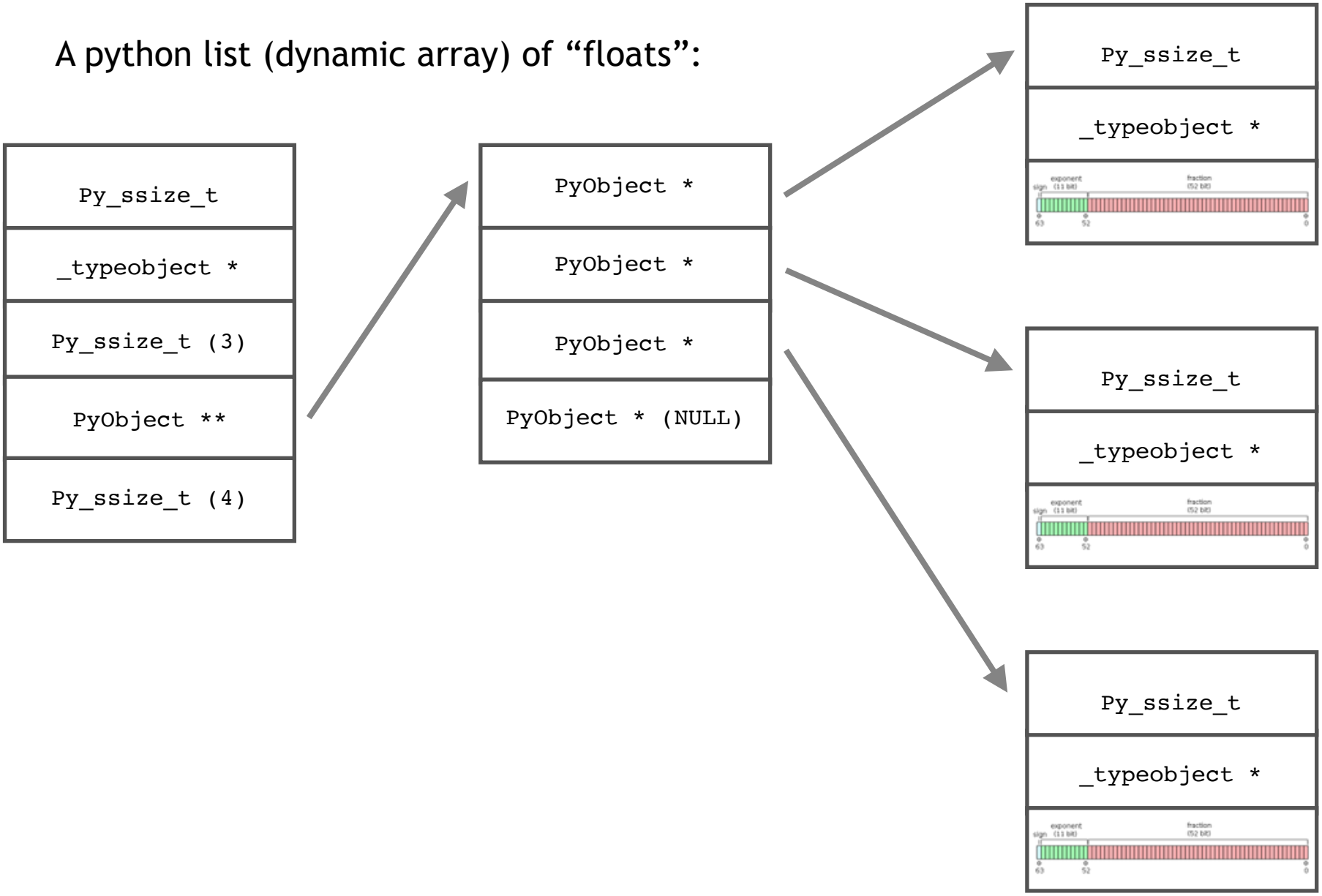
Python dynamic typing and memory management by reference counting incurs memory overhead (64-bit architecture):



A python “float” object.

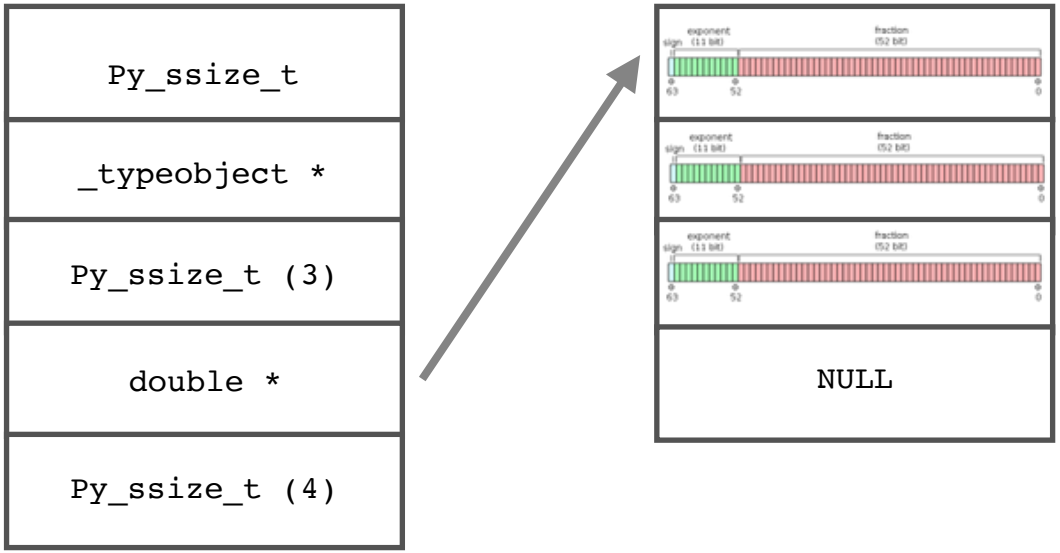
A Python List of Floats

A python list (dynamic array) of “floats”:



Something better for HPC

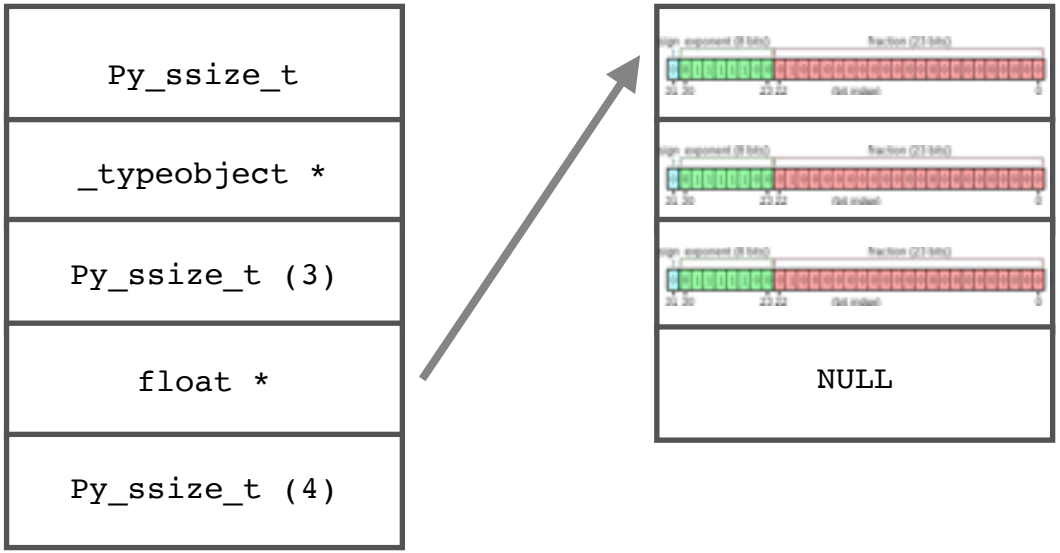
Something like this would be better for performance



8 bytes per double, no dereferencing individual pointers, checking types for each one.

Something better for HPC

Or even better if you don't need double precision!



4 bytes per float, no dereferencing individual pointers, checking types for each one.

Python Bytecode and Interpretation

ACCRE
Advanced Computing Center for Research & Education

- Python source is compiled into a platform-independent bytecode
- (CPython) A simple virtual stack machine, written in C, executes the bytecode
- (CPython) bytecode is NOT just-in-time compiled to native processor instructions
- Other implementations of python, such as pypy, do improve performance by compiling to native processor instructions.

Example Function

```
>>> import dis
>>> def foo(x, y):
...     return x * y + 17
...
>>> foo.__code__.co_varnames
('x', 'y')
>>> foo.__code__.co_consts
(None, 17)
>>> [hex(b) for b in foo.__code__.co_code]
['0x7c', '0x0', '0x7c', '0x1', '0x14', '0x0', '0x64', '0x1', '0x17', '0x0', '0x53', '0x0']
>>> dis.dis(foo)
  2           0 LOAD_FAST                0 (x)
              2 LOAD_FAST                1 (y)
              4 BINARY_MULTIPLY
              6 LOAD_CONST               1 (17)
              8 BINARY_ADD
             10 RETURN_VALUE
>>>
```

Example Function Invocation

ACCRE
Advanced Computing Center for Research & Education

CALLED: foo(3, 5)

function code

```
  0 LOAD_FAST                0 (x)
  2 LOAD_FAST                1 (y)
  4 BINARY_MULTIPLY
  6 LOAD_CONST               1 (17)
  8 BINARY_ADD
 10 RETURN_VALUE
```

co_varnames

PyObject * (3)	PyObject * (5)

co_consts

PyObject * (None)	PyObject * (17)

object stack

Example Function Invocation

ACCRE
Advanced Computing Center for Research & Education

CALLED: foo(3, 5)

function code

```
 0 LOAD_FAST                0 (x)
 2 LOAD_FAST                1 (y)
 4 BINARY_MULTIPLY
 6 LOAD_CONST               1 (17)
 8 BINARY_ADD
10 RETURN_VALUE
```

co_varnames

PyObject * (3)	PyObject * (5)

co_consts

PyObject * (None)	PyObject * (17)

object stack

Example Function Invocation

ACCRE Advanced Computing Center for Research & Education

CALLED: foo(3, 5)

function code

```
 0 LOAD_FAST                0 (x)
 2 LOAD_FAST                1 (y)
 4 BINARY_MULTIPLY
 6 LOAD_CONST               1 (17)
 8 BINARY_ADD
10 RETURN_VALUE
```

co_varnames

PyObject * (3)	PyObject * (5)

co_consts

PyObject * (None)	PyObject * (17)

PUSH!

object stack

PyObject * (3)

Example Function Invocation

ACCRE
Advanced Computing Center for Research & Education

CALLED: foo(3, 5)

function code

```
  0 LOAD_FAST                0 (x)
  2 LOAD_FAST                1 (y)
  4 BINARY_MULTIPLY
  6 LOAD_CONST               1 (17)
  8 BINARY_ADD
 10 RETURN_VALUE
```

co_varnames

PyObject * (3)	PyObject * (5)

co_consts

PyObject * (None)	PyObject * (17)

object stack

PyObject * (3)

Example Function Invocation

ACCRE
Advanced Computing Center for Research & Education

CALLED: foo(3, 5)

function code

```
 0 LOAD_FAST                0 (x)
 2 LOAD_FAST                1 (y)
 4 BINARY_MULTIPLY
 6 LOAD_CONST               1 (17)
 8 BINARY_ADD
10 RETURN_VALUE
```

co_varnames

PyObject * (3)	PyObject * (5)

co_consts

PyObject * (None)	PyObject * (17)

PUSH!

object stack

PyObject * (5)

PyObject * (3)

Example Function Invocation

ACCRE
Advanced Computing Center for Research & Education

CALLED: foo(3, 5)

function code

```
  0 LOAD_FAST                0 (x)
  2 LOAD_FAST                1 (y)
  4 BINARY_MULTIPLY
  6 LOAD_CONST               1 (17)
  8 BINARY_ADD
 10 RETURN_VALUE
```

co_varnames

PyObject * (3)	PyObject * (5)

co_consts

PyObject * (None)	PyObject * (17)

object stack

PyObject * (5)

PyObject * (3)

Example Function Invocation

ACCRE
Advanced Computing Center for Research & Education

CALLED: foo(3, 5)

function code

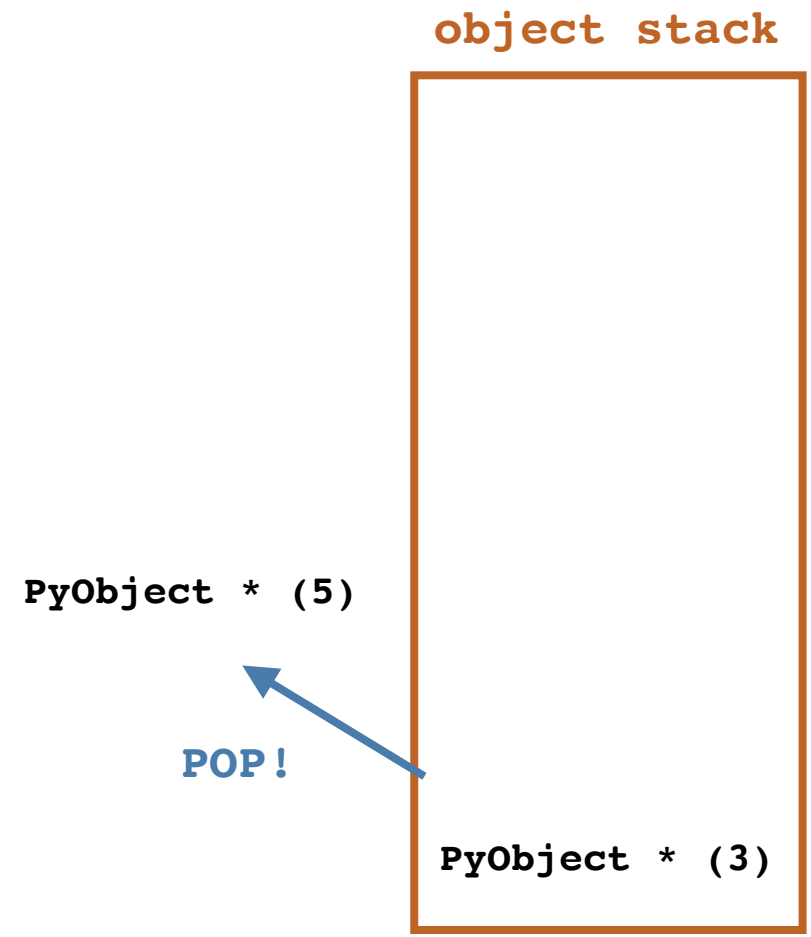
```
  0 LOAD_FAST                0 (x)
  2 LOAD_FAST                1 (y)
  4 BINARY_MULTIPLY
  6 LOAD_CONST               1 (17)
  8 BINARY_ADD
 10 RETURN_VALUE
```

co_varnames

PyObject * (3)	PyObject * (5)

co_consts

PyObject * (None)	PyObject * (17)



Example Function Invocation

ACCRE Advanced Computing Center for Research & Education

CALLED: foo(3, 5)

function code

```
 0 LOAD_FAST              0 (x)
 2 LOAD_FAST              1 (y)
 4 BINARY_MULTIPLY
 6 LOAD_CONST             1 (17)
 8 BINARY_ADD
10 RETURN_VALUE
```

co_varnames

PyObject * (3)	PyObject * (5)

co_consts

PyObject * (None)	PyObject * (17)

object stack

PyObject * (5)

PyObject * (3)

POP!

Example Function Invocation

ACCRE Advanced Computing Center for Research & Education

CALLED: foo(3, 5)

function code

```
 0 LOAD_FAST              0 (x)
 2 LOAD_FAST              1 (y)
 4 BINARY_MULTIPLY
 6 LOAD_CONST             1 (17)
 8 BINARY_ADD
10 RETURN_VALUE
```

co_varnames

PyObject * (3)	PyObject * (5)

co_consts

PyObject * (None)	PyObject * (17)

__mult__

PyObject * (5)

PyObject * (3)

PyObject * (15)

object stack

Example Function Invocation

ACCRE Advanced Computing Center for Research & Education

CALLED: foo(3, 5)

function code

```
 0 LOAD_FAST                0 (x)
 2 LOAD_FAST                1 (y)
 4 BINARY_MULTIPLY
 6 LOAD_CONST               1 (17)
 8 BINARY_ADD
10 RETURN_VALUE
```

co_varnames

PyObject * (3)	PyObject * (5)

co_consts

PyObject * (None)	PyObject * (17)

object stack

PyObject * (15)

PyObject * (15)

PUSH!

Example Function Invocation

ACCRE
Advanced Computing Center for Research & Education

CALLED: foo(3, 5)

function code

```
 0 LOAD_FAST                0 (x)
 2 LOAD_FAST                1 (y)
 4 BINARY_MULTIPLY
 6 LOAD_CONST               1 (17)
 8 BINARY_ADD
10 RETURN_VALUE
```

co_varnames

PyObject * (3)	PyObject * (5)

co_consts

PyObject * (None)	PyObject * (17)

object stack

PyObject * (15)

Example Function Invocation

ACCRE Advanced Computing Center for Research & Education

CALLED: foo(3, 5)

function code

```
 0 LOAD_FAST              0 (x)
 2 LOAD_FAST              1 (y)
 4 BINARY_MULTIPLY
 6 LOAD_CONST             1 (17)
 8 BINARY_ADD
10 RETURN_VALUE
```

co_varnames

PyObject * (3)	PyObject * (5)

co_consts

PyObject * (None)	PyObject * (17)

PUSH!

object stack

PyObject * (17)

PyObject * (15)

Example Function Invocation

ACCRE
Advanced Computing Center for Research & Education

CALLED: foo(3, 5)

function code

```
 0 LOAD_FAST                0 (x)
 2 LOAD_FAST                1 (y)
 4 BINARY_MULTIPLY
 6 LOAD_CONST               1 (17)
 8 BINARY_ADD
10 RETURN_VALUE
```

co_varnames

PyObject * (3)	PyObject * (5)

co_consts

PyObject * (None)	PyObject * (17)

object stack

PyObject * (17)

PyObject * (15)

Example Function Invocation

ACCRE
Advanced Computing Center for Research & Education

CALLED: foo(3, 5)

function code

```
 0 LOAD_FAST                0 (x)
 2 LOAD_FAST                1 (y)
 4 BINARY_MULTIPLY
 6 LOAD_CONST               1 (17)
 8 BINARY_ADD
10 RETURN_VALUE
```

co_varnames

PyObject * (3)	PyObject * (5)

co_consts

PyObject * (None)	PyObject * (17)

object stack

PyObject * (17)

POP!

PyObject * (15)

Example Function Invocation

ACCRE Advanced Computing Center for Research & Education

CALLED: foo(3, 5)

function code

```
 0 LOAD_FAST              0 (x)
 2 LOAD_FAST              1 (y)
 4 BINARY_MULTIPLY
 6 LOAD_CONST             1 (17)
 8 BINARY_ADD
10 RETURN_VALUE
```

co_varnames

PyObject * (3)	PyObject * (5)

co_consts

PyObject * (None)	PyObject * (17)

object stack

PyObject * (17)

PyObject * (15)

POP!

Example Function Invocation

ACCRE Advanced Computing Center for Research & Education

CALLED: foo(3, 5)

function code

```
 0 LOAD_FAST              0 (x)
 2 LOAD_FAST              1 (y)
 4 BINARY_MULTIPLY
 6 LOAD_CONST             1 (17)
 8 BINARY_ADD
10 RETURN_VALUE
```

co_varnames

PyObject * (3)	PyObject * (5)

co_consts

PyObject * (None)	PyObject * (17)

__add__

PyObject * (17)

PyObject * (15)

PyObject * (32)

object stack

Example Function Invocation

ACCRE
Advanced Computing Center for Research & Education

CALLED: foo(3, 5)

function code

```
 0 LOAD_FAST                0 (x)
 2 LOAD_FAST                1 (y)
 4 BINARY_MULTIPLY
 6 LOAD_CONST               1 (17)
 8 BINARY_ADD
10 RETURN_VALUE
```

co_varnames

PyObject * (3)	PyObject * (5)

co_consts

PyObject * (None)	PyObject * (17)

object stack

PyObject * (32)

PyObject * (32)

PUSH!

Example Function Invocation

ACCRE
Advanced Computing Center for Research & Education

CALLED: foo(3, 5)

function code

```
 0 LOAD_FAST                0 (x)
 2 LOAD_FAST                1 (y)
 4 BINARY_MULTIPLY
 6 LOAD_CONST               1 (17)
 8 BINARY_ADD
10 RETURN_VALUE
```

co_varnames

PyObject * (3)	PyObject * (5)

co_consts

PyObject * (None)	PyObject * (17)

object stack

PyObject * (32)

Example Function Invocation

ACCRE Advanced Computing Center for Research & Education

CALLED: foo(3, 5)

function code

```
 0 LOAD_FAST                0 (x)
 2 LOAD_FAST                1 (y)
 4 BINARY_MULTIPLY
 6 LOAD_CONST               1 (17)
 8 BINARY_ADD
10 RETURN_VALUE
```

co_varnames

PyObject * (3)	PyObject * (5)

co_consts

PyObject * (None)	PyObject * (17)

PyObject * (32)

POP!

object stack

PyObject * (32)

Example Function Invocation

ACCRE
Advanced Computing Center for Research & Education

CALLED: foo(3, 5)

function code

```
 0 LOAD_FAST                0 (x)
 2 LOAD_FAST                1 (y)
 4 BINARY_MULTIPLY
 6 LOAD_CONST               1 (17)
 8 BINARY_ADD
10 RETURN_VALUE
```

co_varnames

PyObject * (3)	PyObject * (5)

co_consts

PyObject * (None)	PyObject * (17)

(return to caller)

PyObject * (32)

object stack

Parallelism in Python

Matrix Inversion Example

ACCRE
Advanced Computing Center for Research & Education

- Take some LU-decompositon and inversion example code written in pure python
- Run it serially over a set of random matrices
- Try improving performance by multithreading
- Try improving performance by multiprocessing

Example code in the repo if you (or I) fall behind!

About Concurrency

Concurrency vs Parallelism

ACCRE
Advanced Computing Center for Research & Education

- Parallelism: Do multiple things at the same time
- Concurrency: Deal with multiple things going on at the same time
- Cooking example

Python supports multiple paradigms for achieving concurrent programming including os-level threads and also native coroutines

Python Coroutine Example

ACCRE
Advanced Computing Center for Research & Education

```python
import asyncio

import aiohttp

BASE_URL = 'https://ericappelt.com/animals/'


async def speak(animal, session):

    async with session.get('{}/{}'.format(BASE_URL, animal)) as response:
        response.raise_for_status()
        sound = await response.text()

    return 'The {} says "{}".'.format(animal, sound)
```