Building a JIT compiler

Brandt Bucher (March 13th, 2025)

Building a JIT compiler*

Brandt Bucher (March 13th, 2025)

What they * tell you about

Brandt Bucher (March 13th, 2025)

What they don't tell you about

Brandt Bucher (March 13th, 2025)

What they don't tell you about building a JIT compiler

Brandt Bucher (March 13th, 2025)

Brandt Bucher

- 8 years using Python.
- 6 years contributing to CPython (the reference implementation of Python).
- 4 years on Microsoft's CPython Performance Engineering Team.
- 2 years working on CPython's experimental new JIT compiler.

Brandt Bucher

- 4 years on Microsoft's CPython Performance Engineering Team.

CPython Performance Engineering Team

CPython Performance Engineering Team

Results

CPython Performance Engineering Team

Results (so far...)

- Python 3.11: 25% faster
- Python 3.12: 4% faster
- Python 3.13: 7% faster
- Python 3.14: 8% faster

CPython Performance Engineering Team

Results (so far...)

- In less than 4 years, Python has gotten about 50% faster:
  - 93% of benchmarks have improved.
  - 48% of benchmarks are *over 50%* faster.
  - 14% of benchmarks are *over 100%* faster.

CPython Performance Engineering Team

Results (so far...)

- In less than 4 years, Python has gotten about 50% faster:
  - 93% of benchmarks have improved.
  - 48% of benchmarks are *over 50%* faster.
  - 14% of benchmarks are *over 100%* faster... including Pylint!

CPython Performance Engineering Team

Results (so far...)

Brandt Bucher

- 4 years on Microsoft's CPython Performance Engineering Team.

Brandt Bucher

- 8 years using Python.
- 6 years contributing to CPython (the reference implementation of Python).
- 4 years on Microsoft's CPython Performance Engineering Team.
- 2 years working on CPython's experimental new JIT compiler.

Brandt Bucher

- 8 years using Python.

Brandt Bucher

- 8 years using Python: CounterAttack

CounterAttack

CounterAttack

Using Python to catch card counters!

CounterAttack

Using Python to catch card counters!

```
ONE_CARD = 1 / 52


def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)
```

CounterAttack

Using Python to catch card counters!

```
ONE_CARD = 1 / 52


def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)
```

CounterAttack

Using Python to catch card counters!

```
ONE_CARD = 1 / 52


def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)
```

CounterAttack

Using Python to catch card counters!

```
ONE_CARD = 1 / 52


def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)
```

CounterAttack

Using Python to catch card counters!

```
ONE_CARD = 1 / 52


def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)
```

CounterAttack

Using Python to catch card counters!

```
ONE_CARD = 1 / 52


def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)
```

CounterAttack

Using Python to catch card counters!

```
ONE_CARD = 1 / 52


def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)
```

CounterAttack

Using Python to catch card counters!

```
ONE_CARD = 1 / 52


def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)
```

CounterAttack

Using Python to catch card counters!

```
ONE_CARD = 1 / 52


def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)
```

```
HI = {"10", "J", "Q", "K", "A"}
LO = {"2", "3", "4", "5", "6"}

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0
```

CounterAttack

Using Python to catch card counters!

```
ONE_CARD = 1 / 52


def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

count(get_cards(), 4, system_hi_lo)
```

```
HI = {"10", "J", "Q", "K", "A"}
LO = {"2", "3", "4", "5", "6"}

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0
```

CounterAttack

Using Python to catch card counters!

```
ONE_CARD = 1 / 52


def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

count(get_cards(), 4, system_hi_lo)
```

```
HI = {"10", "J", "Q", "K", "A"}
LO = {"2", "3", "4", "5", "6"}

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

CounterAttack

Using Python to catch card counters!

```
ONE_CARD = 1 / 52


def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

count(get_cards(), 4, system_hi_lo)

count(get_cards(), 6, system_a_5)
```

```
HI = {"10", "J", "Q", "K", "A"}
LO = {"2", "3", "4", "5", "6"}

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

Bytecode

Python 3.10

```
ONE_CARD = 1 / 52


def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)
```

Bytecode

Python 3.10

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

Bytecode

Python 3.10

```
card
decks -= ONE_CARD
count += system card
print count / decks
```

Bytecode
Python 3.10

```
FOR_ITER
STORE_FAST    card
LOAD_FAST     decks
LOAD_GLOBAL   ONE_CARD
BINARY_OP     -=
STORE_FAST    decks
LOAD_FAST     count
LOAD_FAST     system
LOAD_FAST     card
CALL          1
BINARY_OP     +=
STORE_FAST    count
LOAD_GLOBAL   print
LOAD_FAST     count
LOAD_FAST     decks
BINARY_OP     /
CALL          1
POP_TOP
JUMP_BACKWARD
```

Bytecode

Python 3.10

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

```
FOR_ITER
STORE_FAST    card
LOAD_FAST     decks
LOAD_GLOBAL   ONE_CARD
BINARY_OP     -=
STORE_FAST    decks
LOAD_FAST     count
LOAD_FAST     system
LOAD_FAST     card
CALL          1
BINARY_OP     +=
STORE_FAST    count
LOAD_GLOBAL   print
LOAD_FAST     count
LOAD_FAST     decks
BINARY_OP     /
CALL          1
POP_TOP
JUMP_BACKWARD
```

Bytecode
Python 3.10

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

```
FOR_ITER
STORE_FAST    card
LOAD_FAST     decks
LOAD_GLOBAL   ONE_CARD
BINARY_OP     -=
STORE_FAST    decks
LOAD_FAST     count
LOAD_FAST     system
LOAD_FAST     card                                    stack
CALL          1
BINARY_OP     +=
STORE_FAST    count
LOAD_GLOBAL   print
LOAD_FAST     count
LOAD_FAST     decks
BINARY_OP     /
CALL          1
POP_TOP
JUMP_BACKWARD
```

Bytecode
Python 3.10

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

```
FOR_ITER
STORE_FAST    card
LOAD_FAST     decks
LOAD_GLOBAL   ONE_CARD
BINARY_OP     -=
STORE_FAST    decks
LOAD_FAST     count
LOAD_FAST     system
LOAD_FAST     card
CALL          1
BINARY_OP     +=
STORE_FAST    count
LOAD_GLOBAL   print
LOAD_FAST     count
LOAD_FAST     decks
BINARY_OP     /
CALL          1
POP_TOP
JUMP_BACKWARD
```

stack
iterator

Bytecode
Python 3.10

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

```
FOR_ITER
STORE_FAST    card
LOAD_FAST     decks
LOAD_GLOBAL   ONE_CARD
BINARY_OP     -=
STORE_FAST    decks
LOAD_FAST     count
LOAD_FAST     system
LOAD_FAST     card
CALL          1
BINARY_OP     +=
STORE_FAST    count
LOAD_GLOBAL   print
LOAD_FAST     count
LOAD_FAST     decks
BINARY_OP     /
CALL          1
POP_TOP
JUMP_BACKWARD
```

stack

next(iterator)

iterator

Bytecode
Python 3.10

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

```
FOR_ITER
STORE_FAST
LOAD_FAST      decks
LOAD_GLOBAL    ONE_CARD
BINARY_OP      -=
STORE_FAST     decks
LOAD_FAST      count
LOAD_FAST      system
LOAD_FAST      card
CALL           1
BINARY_OP      +=
STORE_FAST     count
LOAD_GLOBAL    print
LOAD_FAST      count
LOAD_FAST      decks
BINARY_OP      /
CALL           1
POP_TOP
JUMP_BACKWARD
```

```
card = next(iterator)
```

stack

iterator

Bytecode
Python 3.10

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

```
FOR_ITER
STORE_FAST
LOAD_FAST
LOAD_GLOBAL    ONE_CARD
BINARY_OP      -=
STORE_FAST     decks
LOAD_FAST      count
LOAD_FAST      system
LOAD_FAST      card
CALL           1
BINARY_OP      +=
STORE_FAST     count
LOAD_GLOBAL    print
LOAD_FAST      count
LOAD_FAST      decks
BINARY_OP      /
CALL           1
POP_TOP
JUMP_BACKWARD
```

```
card = next(iterator)
```

stack
decks
iterator

Bytecode
Python 3.10

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

```
FOR_ITER
STORE_FAST
LOAD_FAST
LOAD_GLOBAL
BINARY_OP      -=
STORE_FAST     decks
LOAD_FAST      count
LOAD_FAST      system
LOAD_FAST      card
CALL           1
BINARY_OP      +=
STORE_FAST     count
LOAD_GLOBAL    print
LOAD_FAST      count
LOAD_FAST      decks
BINARY_OP      /
CALL           1
POP_TOP
JUMP_BACKWARD
```

```
card = next(iterator)
```

stack
ONE_CARD
decks
iterator

Bytecode
Python 3.10

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

```
FOR_ITER
STORE_FAST
LOAD_FAST
LOAD_GLOBAL
BINARY_OP
STORE_FAST     decks
LOAD_FAST      count
LOAD_FAST      system
LOAD_FAST      card
CALL           1
BINARY_OP      +=
STORE_FAST     count
LOAD_GLOBAL    print
LOAD_FAST      count
LOAD_FAST      decks
BINARY_OP      /
CALL           1
POP_TOP
JUMP_BACKWARD
```

```
card = next(iterator)
```

stack

decks - ONE_CARD

iterator

Bytecode

Python 3.10

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

```
FOR_ITER
STORE_FAST
LOAD_FAST
LOAD_GLOBAL
BINARY_OP
STORE_FAST
LOAD_FAST      count
LOAD_FAST      system
LOAD_FAST      card
CALL           1
BINARY_OP      +=
STORE_FAST     count
LOAD_GLOBAL    print
LOAD_FAST      count
LOAD_FAST      decks
BINARY_OP      /
CALL           1
POP_TOP
JUMP_BACKWARD
```

```
card = next(iterator)
decks = decks - ONE_CARD
```

stack

iterator

Bytecode

Python 3.10

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

```
FOR_ITER
STORE_FAST
LOAD_FAST
LOAD_GLOBAL
BINARY_OP
STORE_FAST
LOAD_FAST
LOAD_FAST       system
LOAD_FAST       card
CALL            1
BINARY_OP       +=
STORE_FAST      count
LOAD_GLOBAL     print
LOAD_FAST       count
LOAD_FAST       decks
BINARY_OP       /
CALL            1
POP_TOP
JUMP_BACKWARD
```

```
card = next(iterator)
decks = decks - ONE_CARD
```

stack
count
iterator

Bytecode
Python 3.10

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

```
FOR_ITER
STORE_FAST
LOAD_FAST
LOAD_GLOBAL
BINARY_OP
STORE_FAST
LOAD_FAST
LOAD_FAST      system
LOAD_FAST      card
CALL           1
BINARY_OP      +=
STORE_FAST     count
LOAD_GLOBAL    print
LOAD_FAST      count
LOAD_FAST      decks
BINARY_OP      /
CALL           1
POP_TOP
JUMP_BACKWARD
```

```
card = next(iterator)
decks = decks - ONE_CARD
```

stack
count
iterator

Bytecode
Python 3.10

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

```
FOR_ITER
STORE_FAST
LOAD_FAST
LOAD_GLOBAL
BINARY_OP
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST      card
CALL           1
BINARY_OP      +=
STORE_FAST     count
LOAD_GLOBAL    print
LOAD_FAST      count
LOAD_FAST      decks
BINARY_OP      /
CALL           1
POP_TOP
JUMP_BACKWARD
```

```
card = next(iterator)
decks = decks - ONE_CARD
```

stack
system
count
iterator

Bytecode
Python 3.10

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

```
FOR_ITER
STORE_FAST
LOAD_FAST
LOAD_GLOBAL
BINARY_OP
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST      card
CALL           1
BINARY_OP      +=
STORE_FAST     count
LOAD_GLOBAL    print
LOAD_FAST      count
LOAD_FAST      decks
BINARY_OP      /
CALL           1
POP_TOP
JUMP_BACKWARD
```

```
card = next(iterator)
decks = decks - ONE_CARD
```

stack
system
count
iterator

Bytecode
Python 3.10

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

```
FOR_ITER
STORE_FAST
LOAD_FAST
LOAD_GLOBAL
BINARY_OP
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL            1
BINARY_OP       +=
STORE_FAST      count
LOAD_GLOBAL     print
LOAD_FAST       count
LOAD_FAST       decks
BINARY_OP       /
CALL            1
POP_TOP
JUMP_BACKWARD
```

```
card = next(iterator)
decks = decks - ONE_CARD
```

stack
card
system
count
iterator

Bytecode
Python 3.10

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

```
FOR_ITER
STORE_FAST
LOAD_FAST
LOAD_GLOBAL
BINARY_OP
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL
BINARY_OP      +=
STORE_FAST     count
LOAD_GLOBAL    print
LOAD_FAST      count
LOAD_FAST      decks
BINARY_OP      /
CALL           1
POP_TOP
JUMP_BACKWARD
```

```
card = next(iterator)
decks = decks - ONE_CARD
```

stack

system(card)
count
iterator

Bytecode

Python 3.10

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

```
FOR_ITER
STORE_FAST
LOAD_FAST
LOAD_GLOBAL
BINARY_OP
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL
BINARY_OP
STORE_FAST     count
LOAD_GLOBAL    print
LOAD_FAST      count
LOAD_FAST      decks
BINARY_OP      /
CALL           1
POP_TOP
JUMP_BACKWARD
```

```
card = next(iterator)
decks = decks - ONE_CARD
```

stack

```
count + system(card)
      iterator
```

Bytecode
Python 3.10

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

```
FOR_ITER
STORE_FAST
LOAD_FAST
LOAD_GLOBAL
BINARY_OP
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL
BINARY_OP
STORE_FAST
LOAD_GLOBAL    print
LOAD_FAST      count
LOAD_FAST      decks
BINARY_OP      /
CALL           1
POP_TOP
JUMP_BACKWARD
```

```
card = next(iterator)
decks = decks - ONE_CARD
count = count + system(card)
```

stack

iterator

Bytecode
Python 3.10

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

```
FOR_ITER
STORE_FAST
LOAD_FAST
LOAD_GLOBAL
BINARY_OP
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL
BINARY_OP
STORE_FAST
LOAD_GLOBAL
LOAD_FAST      count
LOAD_FAST      decks
BINARY_OP      /
CALL           1
POP_TOP
JUMP_BACKWARD
```

```
card = next(iterator)
decks = decks - ONE_CARD
count = count + system(card)
```

```
stack
print
iterator
```

Bytecode
Python 3.10

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

```
FOR_ITER
STORE_FAST
LOAD_FAST
LOAD_GLOBAL
BINARY_OP
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL
BINARY_OP
STORE_FAST
LOAD_GLOBAL
LOAD_FAST
LOAD_FAST      decks
BINARY_OP      /
CALL           1
POP_TOP
JUMP_BACKWARD
```

```
card = next(iterator)
decks = decks - ONE_CARD
count = count + system(card)
```

stack
count
print
iterator

Bytecode
Python 3.10

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

```
FOR_ITER
STORE_FAST
LOAD_FAST
LOAD_GLOBAL
BINARY_OP
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL
BINARY_OP
STORE_FAST
LOAD_GLOBAL
LOAD_FAST
LOAD_FAST      decks
BINARY_OP      /
CALL           1
POP_TOP
JUMP_BACKWARD
```

```
card = next(iterator)
decks = decks - ONE_CARD
count = count + system(card)
```

```
stack
count
print
iterator
```

Bytecode
Python 3.10

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

```
FOR_ITER
STORE_FAST
LOAD_FAST
LOAD_GLOBAL
BINARY_OP
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL
BINARY_OP
STORE_FAST
LOAD_GLOBAL
LOAD_FAST
LOAD_FAST
BINARY_OP      /
CALL           1
POP_TOP
JUMP_BACKWARD
```

```
card = next(iterator)
decks = decks - ONE_CARD
count = count + system(card)
```

stack
decks
count
print
iterator

Bytecode
Python 3.10

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

```
FOR_ITER
STORE_FAST
LOAD_FAST
LOAD_GLOBAL
BINARY_OP
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL
BINARY_OP
STORE_FAST
LOAD_GLOBAL
LOAD_FAST
LOAD_FAST
BINARY_OP
CALL           1
POP_TOP
JUMP_BACKWARD
```

```
card = next(iterator)
decks = decks - ONE_CARD
count = count + system(card)
```

stack

count / decks

print

iterator

Bytecode

Python 3.10

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

```
FOR_ITER
STORE_FAST
LOAD_FAST
LOAD_GLOBAL
BINARY_OP
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL
BINARY_OP
STORE_FAST
LOAD_GLOBAL
LOAD_FAST
LOAD_FAST
BINARY_OP
CALL
POP_TOP
JUMP_BACKWARD
```

```
card = next(iterator)
decks = decks - ONE_CARD
count = count + system(card)
```

stack

print(count / decks)

iterator

Bytecode
Python 3.10

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

```
FOR_ITER
STORE_FAST
LOAD_FAST
LOAD_GLOBAL
BINARY_OP
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL
BINARY_OP
STORE_FAST
LOAD_GLOBAL
LOAD_FAST
LOAD_FAST
BINARY_OP
CALL
POP_TOP
JUMP_BACKWARD
```

```
card = next(iterator)
decks = decks - ONE_CARD
count = count + system(card)
```

stack
iterator

Bytecode
Python 3.10

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

```
FOR_ITER
STORE_FAST
LOAD_FAST
LOAD_GLOBAL
BINARY_OP
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL
BINARY_OP
STORE_FAST
LOAD_GLOBAL
LOAD_FAST
LOAD_FAST
BINARY_OP
CALL
POP_TOP
JUMP_BACKWARD
```

```
card = next(iterator)
decks = decks - ONE_CARD
count = count + system(card)
```

stack

iterator

Bytecode
Python 3.10

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

```
FOR_ITER
STORE_FAST
LOAD_FAST
LOAD_GLOBAL
BINARY_OP
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL
BINARY_OP
STORE_FAST
LOAD_GLOBAL
LOAD_FAST
LOAD_FAST
BINARY_OP
CALL
POP_TOP
JUMP_BACKWARD
```

```
card = next(iterator)
decks = decks - ONE_CARD
count = count + system(card)
```

stack
iterator

Specialized Bytecode

Python 3.11

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

```
FOR_ITER
STORE_FAST
LOAD_FAST
LOAD_GLOBAL
BINARY_OP
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL
BINARY_OP
STORE_FAST
LOAD_GLOBAL
LOAD_FAST
LOAD_FAST
BINARY_OP
CALL
POP_TOP
JUMP_BACKWARD
```

Specialized Bytecode

Python 3.11

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

```
FOR_ITER
STORE_FAST
LOAD_FAST
LOAD_GLOBAL
BINARY_OP
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL
BINARY_OP
STORE_FAST
LOAD_GLOBAL
LOAD_FAST
LOAD_FAST
BINARY_OP
CALL
POP_TOP
JUMP_BACKWARD
```

Specialized Bytecode

Python 3.11

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

```
FOR_ITER_LIST
STORE_FAST
LOAD_FAST
LOAD_GLOBAL
BINARY_OP
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL
BINARY_OP
STORE_FAST
LOAD_GLOBAL
LOAD_FAST
LOAD_FAST
BINARY_OP
CALL
POP_TOP
JUMP_BACKWARD
```

Specialized Bytecode

Python 3.11

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

```
FOR_ITER_LIST
STORE_FAST
LOAD_FAST
LOAD_GLOBAL
BINARY_OP
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL
BINARY_OP
STORE_FAST
LOAD_GLOBAL
LOAD_FAST
LOAD_FAST
BINARY_OP
CALL
POP_TOP
JUMP_BACKWARD
```

Specialized Bytecode

Python 3.11

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

```
FOR_ITER_LIST
STORE_FAST
LOAD_FAST
LOAD_GLOBAL_MODULE
BINARY_OP
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL
BINARY_OP
STORE_FAST
LOAD_GLOBAL
LOAD_FAST
LOAD_FAST
BINARY_OP
CALL
POP_TOP
JUMP_BACKWARD
```

Specialized Bytecode

Python 3.11

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

```
FOR_ITER_LIST
STORE_FAST
LOAD_FAST
LOAD_GLOBAL_MODULE
BINARY_OP
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL
BINARY_OP
STORE_FAST
LOAD_GLOBAL
LOAD_FAST
LOAD_FAST
BINARY_OP
CALL
POP_TOP
JUMP_BACKWARD
```

Specialized Bytecode

Python 3.11

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

```
FOR_ITER_LIST
STORE_FAST
LOAD_FAST
LOAD_GLOBAL_MODULE
BINARY_OP_SUBTRACT_FLOAT
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL
BINARY_OP
STORE_FAST
LOAD_GLOBAL
LOAD_FAST
LOAD_FAST
BINARY_OP
CALL
POP_TOP
JUMP_BACKWARD
```

Specialized Bytecode

Python 3.11

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

```
FOR_ITER_LIST
STORE_FAST
LOAD_FAST
LOAD_GLOBAL_MODULE
BINARY_OP_SUBTRACT_FLOAT
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL
BINARY_OP
STORE_FAST
LOAD_GLOBAL
LOAD_FAST
LOAD_FAST
BINARY_OP
CALL
POP_TOP
JUMP_BACKWARD
```

Specialized Bytecode

Python 3.11

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

```
FOR_ITER_LIST
STORE_FAST
LOAD_FAST
LOAD_GLOBAL_MODULE
BINARY_OP_SUBTRACT_FLOAT
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL_PY_EXACT_ARGS
BINARY_OP
STORE_FAST
LOAD_GLOBAL
LOAD_FAST
LOAD_FAST
BINARY_OP
CALL
POP_TOP
JUMP_BACKWARD
```

Specialized Bytecode

Python 3.11

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

```
FOR_ITER_LIST
STORE_FAST
LOAD_FAST
LOAD_GLOBAL_MODULE
BINARY_OP_SUBTRACT_FLOAT
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL_PY_EXACT_ARGS
BINARY_OP
STORE_FAST
LOAD_GLOBAL
LOAD_FAST
LOAD_FAST
BINARY_OP
CALL
POP_TOP
JUMP_BACKWARD
```

Specialized Bytecode

Python 3.11

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

```
FOR_ITER_LIST
STORE_FAST
LOAD_FAST
LOAD_GLOBAL_MODULE
BINARY_OP_SUBTRACT_FLOAT
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL_PY_EXACT_ARGS
BINARY_OP_ADD_INT
STORE_FAST
LOAD_GLOBAL
LOAD_FAST
LOAD_FAST
BINARY_OP
CALL
POP_TOP
JUMP_BACKWARD
```

Specialized Bytecode

Python 3.11

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

```
FOR_ITER_LIST
STORE_FAST
LOAD_FAST
LOAD_GLOBAL_MODULE
BINARY_OP_SUBTRACT_FLOAT
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL_PY_EXACT_ARGS
BINARY_OP_ADD_INT
STORE_FAST
LOAD_GLOBAL
LOAD_FAST
LOAD_FAST
BINARY_OP
CALL
POP_TOP
JUMP_BACKWARD
```

Specialized Bytecode

Python 3.11

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

```
FOR_ITER_LIST
STORE_FAST
LOAD_FAST
LOAD_GLOBAL_MODULE
BINARY_OP_SUBTRACT_FLOAT
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL_PY_EXACT_ARGS
BINARY_OP_ADD_INT
STORE_FAST
LOAD_GLOBAL_BUILTIN
LOAD_FAST
LOAD_FAST
BINARY_OP
CALL
POP_TOP
JUMP_BACKWARD
```

Specialized Bytecode

Python 3.11

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

```
FOR_ITER_LIST
STORE_FAST
LOAD_FAST
LOAD_GLOBAL_MODULE
BINARY_OP_SUBTRACT_FLOAT
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL_PY_EXACT_ARGS
BINARY_OP_ADD_INT
STORE_FAST
LOAD_GLOBAL_BUILTIN
LOAD_FAST
LOAD_FAST
BINARY_OP
CALL
POP_TOP
JUMP_BACKWARD
```

Specialized Bytecode

Python 3.11

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

```
FOR_ITER_LIST
STORE_FAST
LOAD_FAST
LOAD_GLOBAL_MODULE
BINARY_OP_SUBTRACT_FLOAT
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL_PY_EXACT_ARGS
BINARY_OP_ADD_INT
STORE_FAST
LOAD_GLOBAL_BUILTIN
LOAD_FAST
LOAD_FAST
BINARY_OP_TRUE_DIVIDE_INT_FLOAT
CALL
POP_TOP
JUMP_BACKWARD
```

Specialized Bytecode

Python 3.11

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

```
FOR_ITER_LIST
STORE_FAST
LOAD_FAST
LOAD_GLOBAL_MODULE
BINARY_OP_SUBTRACT_FLOAT
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL_PY_EXACT_ARGS
BINARY_OP_ADD_INT
STORE_FAST
LOAD_GLOBAL_BUILTIN
LOAD_FAST
LOAD_FAST
BINARY_OP_TRUE_DIVIDE_INT_FLOAT
CALL
POP_TOP
JUMP_BACKWARD
```

Specialized Bytecode

Python 3.11

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

```
FOR_ITER_LIST
STORE_FAST
LOAD_FAST
LOAD_GLOBAL_MODULE
BINARY_OP_SUBTRACT_FLOAT
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL_PY_EXACT_ARGS
BINARY_OP_ADD_INT
STORE_FAST
LOAD_GLOBAL_BUILTIN
LOAD_FAST
LOAD_FAST
BINARY_OP_TRUE_DIVIDE_INT_FLOAT
CALL_BUILTIN_FAST_WITH_KEYWORDS
POP_TOP
JUMP_BACKWARD
```

Specialized Bytecode

Python 3.11

```
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

```
FOR_ITER_LIST
STORE_FAST
LOAD_FAST
LOAD_GLOBAL_MODULE
BINARY_OP_SUBTRACT_FLOAT
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL_PY_EXACT_ARGS
BINARY_OP_ADD_INT
STORE_FAST
LOAD_GLOBAL_BUILTIN
LOAD_FAST
LOAD_FAST
BINARY_OP_TRUE_DIVIDE_INT_FLOAT
CALL_BUILTIN_FAST_WITH_KEYWORDS
POP_TOP
JUMP_BACKWARD
```

Specialized Bytecode

Python 3.11

```
FOR_ITER_LIST
STORE_FAST
LOAD_FAST
LOAD_GLOBAL_MODULE
BINARY_OP_SUBTRACT_FLOAT
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL_PY_EXACT_ARGS
BINARY_OP_ADD_INT
STORE_FAST
LOAD_GLOBAL_BUILTIN
LOAD_FAST
LOAD_FAST
BINARY_OP_TRUE_DIVIDE_INT_FLOAT
CALL_BUILTIN_FAST_WITH_KEYWORDS
POP_TOP
JUMP_BACKWARD
```

Micro-Ops

Python 3.12

```
FOR_ITER_LIST
STORE_FAST
LOAD_FAST
LOAD_GLOBAL_MODULE
BINARY_OP_SUBTRACT_FLOAT
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL_PY_EXACT_ARGS
BINARY_OP_ADD_INT
STORE_FAST
LOAD_GLOBAL_BUILTIN
LOAD_FAST
LOAD_FAST
BINARY_OP_TRUE_DIVIDE_INT_FLOAT
CALL_BUILTIN_FAST_WITH_KEYWORDS
POP_TOP
JUMP_BACKWARD
```

Micro-Ops

Python 3.12

```
_CHECK_VALIDITY_AND_SET_IP
_ITER_CHECK_LIST
_GUARD_NOT_EXHAUSTED_LIST
_ITER_NEXT_LIST
```

```
STORE_FAST
LOAD_FAST
LOAD_GLOBAL_MODULE
BINARY_OP_SUBTRACT_FLOAT
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL_PY_EXACT_ARGS
BINARY_OP_ADD_INT
STORE_FAST
LOAD_GLOBAL_BUILTIN
LOAD_FAST
LOAD_FAST
BINARY_OP_TRUE_DIVIDE_INT_FLOAT
CALL_BUILTIN_FAST_WITH_KEYWORDS
POP_TOP
JUMP_BACKWARD
```

Micro-Ops

Python 3.12

```
_CHECK_VALIDITY_AND_SET_IP
_ITER_CHECK_LIST
_GUARD_NOT_EXHAUSTED_LIST
_ITER_NEXT_LIST

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST
```

```
LOAD_FAST
LOAD_GLOBAL_MODULE
BINARY_OP_SUBTRACT_FLOAT
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL_PY_EXACT_ARGS
BINARY_OP_ADD_INT
STORE_FAST
LOAD_GLOBAL_BUILTIN
LOAD_FAST
LOAD_FAST
BINARY_OP_TRUE_DIVIDE_INT_FLOAT
CALL_BUILTIN_FAST_WITH_KEYWORDS
POP_TOP
JUMP_BACKWARD
```

Micro-Ops

Python 3.12

```
_CHECK_VALIDITY_AND_SET_IP
_ITER_CHECK_LIST
_GUARD_NOT_EXHAUSTED_LIST
_ITER_NEXT_LIST

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST
```

```
LOAD_GLOBAL_MODULE
BINARY_OP_SUBTRACT_FLOAT
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL_PY_EXACT_ARGS
BINARY_OP_ADD_INT
STORE_FAST
LOAD_GLOBAL_BUILTIN
LOAD_FAST
LOAD_FAST
BINARY_OP_TRUE_DIVIDE_INT_FLOAT
CALL_BUILTIN_FAST_WITH_KEYWORDS
POP_TOP
JUMP_BACKWARD
```

Micro-Ops

Python 3.12

```
_CHECK_VALIDITY_AND_SET_IP
_ITER_CHECK_LIST
_GUARD_NOT_EXHAUSTED_LIST
_ITER_NEXT_LIST

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_GLOBAL_MODULE
```

```
BINARY_OP_SUBTRACT_FLOAT
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL_PY_EXACT_ARGS
BINARY_OP_ADD_INT
STORE_FAST
LOAD_GLOBAL_BUILTIN
LOAD_FAST
LOAD_FAST
BINARY_OP_TRUE_DIVIDE_INT_FLOAT
CALL_BUILTIN_FAST_WITH_KEYWORDS
POP_TOP
JUMP_BACKWARD
```

Micro-Ops

Python 3.12

```
_CHECK_VALIDITY_AND_SET_IP
_ITER_CHECK_LIST
_GUARD_NOT_EXHAUSTED_LIST
_ITER_NEXT_LIST

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_GLOBAL_MODULE

_CHECK_VALIDITY_AND_SET_IP
_GUARD_BOTH_FLOAT
_BINARY_OP_SUBTRACT_FLOAT
```

```
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL_PY_EXACT_ARGS
BINARY_OP_ADD_INT
STORE_FAST
LOAD_GLOBAL_BUILTIN
LOAD_FAST
LOAD_FAST
BINARY_OP_TRUE_DIVIDE_INT_FLOAT
CALL_BUILTIN_FAST_WITH_KEYWORDS
POP_TOP
JUMP_BACKWARD
```

Micro-Ops

Python 3.12

```
_CHECK_VALIDITY_AND_SET_IP
_ITER_CHECK_LIST
_GUARD_NOT_EXHAUSTED_LIST
_ITER_NEXT_LIST

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_GLOBAL_MODULE

_CHECK_VALIDITY_AND_SET_IP
_GUARD_BOTH_FLOAT
_BINARY_OP_SUBTRACT_FLOAT

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST
```

```
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL_PY_EXACT_ARGS
BINARY_OP_ADD_INT
STORE_FAST
LOAD_GLOBAL_BUILTIN
LOAD_FAST
LOAD_FAST
BINARY_OP_TRUE_DIVIDE_INT_FLOAT
CALL_BUILTIN_FAST_WITH_KEYWORDS
POP_TOP
JUMP_BACKWARD
```

Micro-Ops

Python 3.12

```
_CHECK_VALIDITY_AND_SET_IP
_ITER_CHECK_LIST
_GUARD_NOT_EXHAUSTED_LIST
_ITER_NEXT_LIST

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_GLOBAL_MODULE

_CHECK_VALIDITY_AND_SET_IP
_GUARD_BOTH_FLOAT
_BINARY_OP_SUBTRACT_FLOAT

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST
```

```
LOAD_FAST
LOAD_FAST
CALL_PY_EXACT_ARGS
BINARY_OP_ADD_INT
STORE_FAST
LOAD_GLOBAL_BUILTIN
LOAD_FAST
LOAD_FAST
BINARY_OP_TRUE_DIVIDE_INT_FLOAT
CALL_BUILTIN_FAST_WITH_KEYWORDS
POP_TOP
JUMP_BACKWARD
```

Micro-Ops

Python 3.12

```
_CHECK_VALIDITY_AND_SET_IP
_ITER_CHECK_LIST
_GUARD_NOT_EXHAUSTED_LIST
_ITER_NEXT_LIST

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_GLOBAL_MODULE

_CHECK_VALIDITY_AND_SET_IP
_GUARD_BOTH_FLOAT
_BINARY_OP_SUBTRACT_FLOAT

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST
```

```
_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST
```

```
LOAD_FAST
CALL_PY_EXACT_ARGS
BINARY_OP_ADD_INT
STORE_FAST
LOAD_GLOBAL_BUILTIN
LOAD_FAST
LOAD_FAST
BINARY_OP_TRUE_DIVIDE_INT_FLOAT
CALL_BUILTIN_FAST_WITH_KEYWORDS
POP_TOP
JUMP_BACKWARD
```

Micro-Ops

Python 3.12

```
_CHECK_VALIDITY_AND_SET_IP
_ITER_CHECK_LIST
_GUARD_NOT_EXHAUSTED_LIST
_ITER_NEXT_LIST

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_GLOBAL_MODULE

_CHECK_VALIDITY_AND_SET_IP
_GUARD_BOTH_FLOAT
_BINARY_OP_SUBTRACT_FLOAT

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST
```

```
_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST
```

```
CALL_PY_EXACT_ARGS
BINARY_OP_ADD_INT
STORE_FAST
LOAD_GLOBAL_BUILTIN
LOAD_FAST
LOAD_FAST
BINARY_OP_TRUE_DIVIDE_INT_FLOAT
CALL_BUILTIN_FAST_WITH_KEYWORDS
POP_TOP
JUMP_BACKWARD
```

Micro-Ops

Python 3.12

```
_CHECK_VALIDITY_AND_SET_IP
_ITER_CHECK_LIST
_GUARD_NOT_EXHAUSTED_LIST
_ITER_NEXT_LIST

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_GLOBAL_MODULE

_CHECK_VALIDITY_AND_SET_IP
_GUARD_BOTH_FLOAT
_BINARY_OP_SUBTRACT_FLOAT

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST
```

```
_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_CHECK_PEP_523
_CHECK_FUNCTION_VERSION
_CHECK_FUNCTION_EXACT_ARGS
_CHECK_STACK_SPACE
_INIT_CALL_PY_EXACT_ARGS
_SAVE_RETURN_OFFSET
_PUSH_FRAME

...
```

```
BINARY_OP_ADD_INT
STORE_FAST
LOAD_GLOBAL_BUILTIN
LOAD_FAST
LOAD_FAST
BINARY_OP_TRUE_DIVIDE_INT_FLOAT
CALL_BUILTIN_FAST_WITH_KEYWORDS
POP_TOP
JUMP_BACKWARD
```

Micro-Ops

Python 3.12

```
_CHECK_VALIDITY_AND_SET_IP
_ITER_CHECK_LIST
_GUARD_NOT_EXHAUSTED_LIST
_ITER_NEXT_LIST

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_GLOBAL_MODULE

_CHECK_VALIDITY_AND_SET_IP
_GUARD_BOTH_FLOAT
_BINARY_OP_SUBTRACT_FLOAT

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST
```

```
_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_CHECK_PEP_523
_CHECK_FUNCTION_VERSION
_CHECK_FUNCTION_EXACT_ARGS
_CHECK_STACK_SPACE
_INIT_CALL_PY_EXACT_ARGS
_SAVE_RETURN_OFFSET
_PUSH_FRAME

...

_CHECK_VALIDITY_AND_SET_IP
_GUARD_BOTH_INT
_BINARY_OP_ADD_INT
```

```
STORE_FAST
LOAD_GLOBAL_BUILTIN
LOAD_FAST
LOAD_FAST
BINARY_OP_TRUE_DIVIDE_INT_FLOAT
CALL_BUILTIN_FAST_WITH_KEYWORDS
POP_TOP
JUMP_BACKWARD
```

Micro-Ops

Python 3.12

```
_CHECK_VALIDITY_AND_SET_IP
_ITER_CHECK_LIST
_GUARD_NOT_EXHAUSTED_LIST
_ITER_NEXT_LIST

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_GLOBAL_MODULE

_CHECK_VALIDITY_AND_SET_IP
_GUARD_BOTH_FLOAT
_BINARY_OP_SUBTRACT_FLOAT

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST
```

```
_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_CHECK_PEP_523
_CHECK_FUNCTION_VERSION
_CHECK_FUNCTION_EXACT_ARGS
_CHECK_STACK_SPACE
_INIT_CALL_PY_EXACT_ARGS
_SAVE_RETURN_OFFSET
_PUSH_FRAME

...

_CHECK_VALIDITY_AND_SET_IP
_GUARD_BOTH_INT
_BINARY_OP_ADD_INT

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST
```

```
LOAD_GLOBAL_BUILTIN
LOAD_FAST
LOAD_FAST
BINARY_OP_TRUE_DIVIDE_INT_FLOAT
CALL_BUILTIN_FAST_WITH_KEYWORDS
POP_TOP
JUMP_BACKWARD
```

Micro-Ops

Python 3.12

```
_CHECK_VALIDITY_AND_SET_IP
_ITER_CHECK_LIST
_GUARD_NOT_EXHAUSTED_LIST
_ITER_NEXT_LIST

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_GLOBAL_MODULE

_CHECK_VALIDITY_AND_SET_IP
_GUARD_BOTH_FLOAT
_BINARY_OP_SUBTRACT_FLOAT

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST
```

```
_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_CHECK_PEP_523
_CHECK_FUNCTION_VERSION
_CHECK_FUNCTION_EXACT_ARGS
_CHECK_STACK_SPACE
_INIT_CALL_PY_EXACT_ARGS
_SAVE_RETURN_OFFSET
_PUSH_FRAME

...

_CHECK_VALIDITY_AND_SET_IP
_GUARD_BOTH_INT
_BINARY_OP_ADD_INT

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST
```

```
_CHECK_VALIDITY_AND_SET_IP
_GUARD_GLOBALS_VERSION
_LOAD_GLOBAL_BUILTINS
```

```
LOAD_FAST
LOAD_FAST
BINARY_OP_TRUE_DIVIDE_INT_FLOAT
CALL_BUILTIN_FAST_WITH_KEYWORDS
POP_TOP
JUMP_BACKWARD
```

Micro-Ops

Python 3.12

```
_CHECK_VALIDITY_AND_SET_IP
_ITER_CHECK_LIST
_GUARD_NOT_EXHAUSTED_LIST
_ITER_NEXT_LIST

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_GLOBAL_MODULE

_CHECK_VALIDITY_AND_SET_IP
_GUARD_BOTH_FLOAT
_BINARY_OP_SUBTRACT_FLOAT

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST
```

```
_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_CHECK_PEP_523
_CHECK_FUNCTION_VERSION
_CHECK_FUNCTION_EXACT_ARGS
_CHECK_STACK_SPACE
_INIT_CALL_PY_EXACT_ARGS
_SAVE_RETURN_OFFSET
_PUSH_FRAME

...

_CHECK_VALIDITY_AND_SET_IP
_GUARD_BOTH_INT
_BINARY_OP_ADD_INT

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST
```

```
_CHECK_VALIDITY_AND_SET_IP
_GUARD_GLOBALS_VERSION
_LOAD_GLOBAL_BUILTINS

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

LOAD_FAST
BINARY_OP_TRUE_DIVIDE_INT_FLOAT
CALL_BUILTIN_FAST_WITH_KEYWORDS
POP_TOP
JUMP_BACKWARD
```

Micro-Ops

Python 3.12

```
_CHECK_VALIDITY_AND_SET_IP
_ITER_CHECK_LIST
_GUARD_NOT_EXHAUSTED_LIST
_ITER_NEXT_LIST

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_GLOBAL_MODULE

_CHECK_VALIDITY_AND_SET_IP
_GUARD_BOTH_FLOAT
_BINARY_OP_SUBTRACT_FLOAT

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST
```

```
_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_CHECK_PEP_523
_CHECK_FUNCTION_VERSION
_CHECK_FUNCTION_EXACT_ARGS
_CHECK_STACK_SPACE
_INIT_CALL_PY_EXACT_ARGS
_SAVE_RETURN_OFFSET
_PUSH_FRAME

...

_CHECK_VALIDITY_AND_SET_IP
_GUARD_BOTH_INT
_BINARY_OP_ADD_INT

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST
```

```
_CHECK_VALIDITY_AND_SET_IP
_GUARD_GLOBALS_VERSION
_LOAD_GLOBAL_BUILTINS

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

BINARY_OP_TRUE_DIVIDE_INT_FLOAT
CALL_BUILTIN_FAST_WITH_KEYWORDS
POP_TOP
JUMP_BACKWARD
```

Micro-Ops

Python 3.12

```
_CHECK_VALIDITY_AND_SET_IP
_ITER_CHECK_LIST
_GUARD_NOT_EXHAUSTED_LIST
_ITER_NEXT_LIST

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_GLOBAL_MODULE

_CHECK_VALIDITY_AND_SET_IP
_GUARD_BOTH_FLOAT
_BINARY_OP_SUBTRACT_FLOAT

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST
```

```
_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_CHECK_PEP_523
_CHECK_FUNCTION_VERSION
_CHECK_FUNCTION_EXACT_ARGS
_CHECK_STACK_SPACE
_INIT_CALL_PY_EXACT_ARGS
_SAVE_RETURN_OFFSET
_PUSH_FRAME

...

_CHECK_VALIDITY_AND_SET_IP
_GUARD_BOTH_INT
_BINARY_OP_ADD_INT

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST
```

```
_CHECK_VALIDITY_AND_SET_IP
_GUARD_GLOBALS_VERSION
_LOAD_GLOBAL_BUILTINS

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_GUARD_BINARY_OP_TRUE_DIVIDE_INT_FLOAT
_BINARY_OP_TRUE_DIVIDE_INT_FLOAT

CALL_BUILTIN_FAST_WITH_KEYWORDS
POP_TOP
JUMP_BACKWARD
```

Micro-Ops

Python 3.12

```
_CHECK_VALIDITY_AND_SET_IP
_ITER_CHECK_LIST
_GUARD_NOT_EXHAUSTED_LIST
_ITER_NEXT_LIST

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_GLOBAL_MODULE

_CHECK_VALIDITY_AND_SET_IP
_GUARD_BOTH_FLOAT
_BINARY_OP_SUBTRACT_FLOAT

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST
```

```
_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_CHECK_PEP_523
_CHECK_FUNCTION_VERSION
_CHECK_FUNCTION_EXACT_ARGS
_CHECK_STACK_SPACE
_INIT_CALL_PY_EXACT_ARGS
_SAVE_RETURN_OFFSET
_PUSH_FRAME

...

_CHECK_VALIDITY_AND_SET_IP
_GUARD_BOTH_INT
_BINARY_OP_ADD_INT

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST
```

```
_CHECK_VALIDITY_AND_SET_IP
_GUARD_GLOBALS_VERSION
_LOAD_GLOBAL_BUILTINS

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_GUARD_BINARY_OP_TRUE_DIVIDE_INT_FLOAT
_BINARY_OP_TRUE_DIVIDE_INT_FLOAT

_CHECK_VALIDITY_AND_SET_IP
_CALL_BUILTIN_FAST_WITH_KEYWORDS
_CHECK_PERIODIC



POP_TOP
JUMP_BACKWARD
```

Micro-Ops

Python 3.12

```
_CHECK_VALIDITY_AND_SET_IP
_ITER_CHECK_LIST
_GUARD_NOT_EXHAUSTED_LIST
_ITER_NEXT_LIST

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_GLOBAL_MODULE

_CHECK_VALIDITY_AND_SET_IP
_GUARD_BOTH_FLOAT
_BINARY_OP_SUBTRACT_FLOAT

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST
```

```
_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_CHECK_PEP_523
_CHECK_FUNCTION_VERSION
_CHECK_FUNCTION_EXACT_ARGS
_CHECK_STACK_SPACE
_INIT_CALL_PY_EXACT_ARGS
_SAVE_RETURN_OFFSET
_PUSH_FRAME

...

_CHECK_VALIDITY_AND_SET_IP
_GUARD_BOTH_INT
_BINARY_OP_ADD_INT

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST
```

```
_CHECK_VALIDITY_AND_SET_IP
_GUARD_GLOBALS_VERSION
_LOAD_GLOBAL_BUILTINS

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_GUARD_BINARY_OP_TRUE_DIVIDE_INT_FLOAT
_BINARY_OP_TRUE_DIVIDE_INT_FLOAT

_CHECK_VALIDITY_AND_SET_IP
_CALL_BUILTIN_FAST_WITH_KEYWORDS
_CHECK_PERIODIC

_CHECK_VALIDITY_AND_SET_IP
_POP_TOP



JUMP_BACKWARD
```

Micro-Ops

Python 3.12

```
_CHECK_VALIDITY_AND_SET_IP
_ITER_CHECK_LIST
_GUARD_NOT_EXHAUSTED_LIST
_ITER_NEXT_LIST

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_GLOBAL_MODULE

_CHECK_VALIDITY_AND_SET_IP
_GUARD_BOTH_FLOAT
_BINARY_OP_SUBTRACT_FLOAT

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST
```

```
_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_CHECK_PEP_523
_CHECK_FUNCTION_VERSION
_CHECK_FUNCTION_EXACT_ARGS
_CHECK_STACK_SPACE
_INIT_CALL_PY_EXACT_ARGS
_SAVE_RETURN_OFFSET
_PUSH_FRAME

...

_CHECK_VALIDITY_AND_SET_IP
_GUARD_BOTH_INT
_BINARY_OP_ADD_INT

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST
```

```
_CHECK_VALIDITY_AND_SET_IP
_GUARD_GLOBALS_VERSION
_LOAD_GLOBAL_BUILTINS

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_GUARD_BINARY_OP_TRUE_DIVIDE_INT_FLOAT
_BINARY_OP_TRUE_DIVIDE_INT_FLOAT

_CHECK_VALIDITY_AND_SET_IP
_CALL_BUILTIN_FAST_WITH_KEYWORDS
_CHECK_PERIODIC

_CHECK_VALIDITY_AND_SET_IP
_POP_TOP

_CHECK_VALIDITY_AND_SET_IP
_CHECK_PERIODIC
_JUMP_TO_TOP
```

Optimized Micro-Ops

Python 3.13

```
_CHECK_VALIDITY_AND_SET_IP
_ITER_CHECK_LIST
_GUARD_NOT_EXHAUSTED_LIST
_ITER_NEXT_LIST

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_GLOBAL_MODULE

_CHECK_VALIDITY_AND_SET_IP
_GUARD_BOTH_FLOAT
_BINARY_OP_SUBTRACT_FLOAT

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST
```

```
_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_CHECK_PEP_523
_CHECK_FUNCTION_VERSION
_CHECK_FUNCTION_EXACT_ARGS
_CHECK_STACK_SPACE
_INIT_CALL_PY_EXACT_ARGS
_SAVE_RETURN_OFFSET
_PUSH_FRAME

...

_CHECK_VALIDITY_AND_SET_IP
_GUARD_BOTH_INT
_BINARY_OP_ADD_INT

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST
```

```
_CHECK_VALIDITY_AND_SET_IP
_GUARD_GLOBALS_VERSION
_LOAD_GLOBAL_BUILTINS

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_GUARD_BINARY_OP_TRUE_DIVIDE_INT_FLOAT
_BINARY_OP_TRUE_DIVIDE_INT_FLOAT

_CHECK_VALIDITY_AND_SET_IP
_CALL_BUILTIN_FAST_WITH_KEYWORDS
_CHECK_PERIODIC

_CHECK_VALIDITY_AND_SET_IP
_POP_TOP

_CHECK_VALIDITY_AND_SET_IP
_CHECK_PERIODIC
_JUMP_TO_TOP
```

Optimized Micro-Ops

Python 3.13

```
_CHECK_VALIDITY_AND_SET_IP
_ITER_CHECK_LIST
_GUARD_NOT_EXHAUSTED_LIST
_ITER_NEXT_LIST

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_GLOBAL_MODULE

_CHECK_VALIDITY_AND_SET_IP
_GUARD_BOTH_FLOAT
_BINARY_OP_SUBTRACT_FLOAT

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST
```

```
_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_CHECK_PEP_523
_CHECK_FUNCTION_VERSION
_CHECK_FUNCTION_EXACT_ARGS
_CHECK_STACK_SPACE
_INIT_CALL_PY_EXACT_ARGS
_SAVE_RETURN_OFFSET
_PUSH_FRAME

...

_CHECK_VALIDITY_AND_SET_IP
_GUARD_BOTH_INT
_BINARY_OP_ADD_INT

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST
```

```
_CHECK_VALIDITY_AND_SET_IP
_GUARD_GLOBALS_VERSION
_LOAD_GLOBAL_BUILTINS

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_LOAD_FAST

_CHECK_VALIDITY_AND_SET_IP
_GUARD_BINARY_OP_TRUE_DIVIDE_INT_FLOAT
_BINARY_OP_TRUE_DIVIDE_INT_FLOAT

_CHECK_VALIDITY_AND_SET_IP
_CALL_BUILTIN_FAST_WITH_KEYWORDS
_CHECK_PERIODIC

_CHECK_VALIDITY_AND_SET_IP
_POP_TOP

_CHECK_VALIDITY_AND_SET_IP
_CHECK_PERIODIC
_JUMP_TO_TOP
```

Optimized Micro-Ops

Python 3.13

```
_CHECK_VALIDITY
_ITER_CHECK_LIST
_GUARD_NOT_EXHAUSTED_LIST
_ITER_NEXT_LIST

_SET_IP
_STORE_FAST

_CHECK_VALIDITY
_LOAD_FAST

_CHECK_FUNCTION
_LOAD_CONST_INLINE


_GUARD_NOS_FLOAT
_BINARY_OP_SUBTRACT_FLOAT

_SET_IP
_STORE_FAST

_CHECK_VALIDITY
_LOAD_FAST
```

```
_LOAD_FAST


_LOAD_FAST

_SET_IP

_CHECK_FUNCTION_VERSION
_CHECK_FUNCTION_EXACT_ARGS
_CHECK_STACK_SPACE
_INIT_CALL_PY_EXACT_ARGS
_SAVE_RETURN_OFFSET
_PUSH_FRAME

...

_CHECK_VALIDITY_AND_SET_IP
_GUARD_NOS_INT
_BINARY_OP_ADD_INT

_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST
```

```
_CHECK_VALIDITY

_LOAD_CONST_INLINE


_LOAD_FAST


_LOAD_FAST

_SET_IP

_BINARY_OP_TRUE_DIVIDE_INT_FLOAT

_CHECK_VALIDITY_AND_SET_IP
_CALL_BUILTIN_FAST_WITH_KEYWORDS
_CHECK_PERIODIC

_CHECK_VALIDITY
_POP_TOP

_CHECK_VALIDITY_AND_SET_IP
_CHECK_PERIODIC
_JUMP_TO_TOP
```

Optimized Micro-Ops

Python 3.13

```
_CHECK_VALIDITY
_ITER_CHECK_LIST
_GUARD_NOT_EXHAUSTED_LIST
_ITER_NEXT_LIST
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
_CHECK_FUNCTION
_LOAD_CONST_INLINE
_GUARD_NOS_FLOAT
_BINARY_OP_SUBTRACT_FLOAT
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
```

Machine Code

Python 3.14

```
_CHECK_VALIDITY
_ITER_CHECK_LIST
_GUARD_NOT_EXHAUSTED_LIST
_ITER_NEXT_LIST
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
_CHECK_FUNCTION
_LOAD_CONST_INLINE
_GUARD_NOS_FLOAT
_BINARY_OP_SUBTRACT_FLOAT
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
```

Machine Code

Python 3.14

```
                                                _CHECK_VALIDITY
_ITER_CHECK_LIST
_GUARD_NOT_EXHAUSTED_LIST
_ITER_NEXT_LIST
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
_CHECK_FUNCTION
_LOAD_CONST_INLINE
_GUARD_NOS_FLOAT
_BINARY_OP_SUBTRACT_FLOAT
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
```

Machine Code

Python 3.14

```
                                                08 00 00 90 08 01 40 f9 08 89 40 39 48 00 00 37 00 00 00 14 00 00 00 14
_ITER_CHECK_LIST
_GUARD_NOT_EXHAUSTED_LIST
_ITER_NEXT_LIST
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
_CHECK_FUNCTION
_LOAD_CONST_INLINE
_GUARD_NOS_FLOAT
_BINARY_OP_SUBTRACT_FLOAT
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
```

Machine Code

Python 3.14

```
08 00 00 90 08 01 40 f9 08 89 40 39 48 00 00 37 00 00 00 14 00 00 00 14
_ITER_CHECK_LIST
```

```
_GUARD_NOT_EXHAUSTED_LIST
_ITER_NEXT_LIST
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
_CHECK_FUNCTION
_LOAD_CONST_INLINE
_GUARD_NOS_FLOAT
_BINARY_OP_SUBTRACT_FLOAT
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
```

Machine Code

Python 3.14

```
08 00 00 90 08 01 40 f9 08 89 40 39 48 00 00 37 00 00 00 14 00 00 00 14 a8 82 5f f8 08
05 40 f9 09 00 00 90 29 01 40 f9 1f 01 09 eb 40 00 00 54 00 00 00 14 00 00 00 14
```

```
_GUARD_NOT_EXHAUSTED_LIST
_ITER_NEXT_LIST
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
_CHECK_FUNCTION
_LOAD_CONST_INLINE
_GUARD_NOS_FLOAT
_BINARY_OP_SUBTRACT_FLOAT
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
```

Machine Code

Python 3.14

```
08 00 00 90 08 01 40 f9 08 89 40 39 48 00 00 37 00 00 00 14 00 00 00 14 a8 82 5f f8 08
05 40 f9 09 00 00 90 29 01 40 f9 1f 01 09 eb 40 00 00 54 00 00 00 14 00 00 00 14
_GUARD_NOT_EXHAUSTED_LIST
```

```
_ITER_NEXT_LIST
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
_CHECK_FUNCTION
_LOAD_CONST_INLINE
_GUARD_NOS_FLOAT
_BINARY_OP_SUBTRACT_FLOAT
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
```

Machine Code

Python 3.14

```
08 00 00 90 08 01 40 f9 08 89 40 39 48 00 00 37 00 00 00 14 00 00 00 14 a8 82 5f f8 08
05 40 f9 09 00 00 90 29 01 40 f9 1f 01 09 eb 40 00 00 54 00 00 00 14 00 00 00 14 a8 82
5f f8 09 0d 40 f9 c9 00 00 b4 0a 09 40 f9 29 09 40 f9 5f 01 09 eb 62 00 00 54 00 00 00
14 00 00 00 14 09 00 80 92 09 09 00 f9 00 00 00 14
```

```
_ITER_NEXT_LIST
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
_CHECK_FUNCTION
_LOAD_CONST_INLINE
_GUARD_NOS_FLOAT
_BINARY_OP_SUBTRACT_FLOAT
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
```

Machine Code

Python 3.14

```
08 00 00 90 08 01 40 f9 08 89 40 39 48 00 00 37 00 00 00 14 00 00 00 14 a8 82 5f f8 08
05 40 f9 09 00 00 90 29 01 40 f9 1f 01 09 eb 40 00 00 54 00 00 00 14 00 00 00 14 a8 82
5f f8 09 0d 40 f9 c9 00 00 b4 0a 09 40 f9 29 09 40 f9 5f 01 09 eb 62 00 00 54 00 00 00
14 00 00 00 14 09 00 80 92 09 09 00 f9 00 00 00 14 _ITER_NEXT_LIST
```

```
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
_CHECK_FUNCTION
_LOAD_CONST_INLINE
_GUARD_NOS_FLOAT
_BINARY_OP_SUBTRACT_FLOAT
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
```

Machine Code

Python 3.14

```
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
_CHECK_FUNCTION
_LOAD_CONST_INLINE
_GUARD_NOS_FLOAT
_BINARY_OP_SUBTRACT_FLOAT
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
```

```
08 00 00 90 08 01 40 f9 08 89 40 39 48 00 00 37 00 00 00 14 00 00 00 14 a8 82 5f f8 08
05 40 f9 09 00 00 90 29 01 40 f9 1f 01 09 eb 40 00 00 54 00 00 00 14 00 00 00 14 a8 82
5f f8 09 0d 40 f9 c9 00 00 b4 0a 09 40 f9 29 09 40 f9 5f 01 09 eb 62 00 00 54 00 00 00
14 00 00 00 14 09 00 80 92 09 09 00 f9 00 00 00 14 a8 82 5f f8 0a 25 41 a9 29 0d 40 f9
4b 05 00 91 0b 09 00 f9 28 79 6a f8 09 01 40 b9 69 00 f8 37 29 05 00 11 09 01 00 b9 a8
86 00 f8 00 00 00 14
```

Machine Code

Python 3.14

```
08 00 00 90 08 01 40 f9 08 89 40 39 48 00 00 37 00 00 00 14 00 00 00 14 a8 82 5f f8 08
05 40 f9 09 00 00 90 29 01 40 f9 1f 01 09 eb 40 00 00 54 00 00 00 14 00 00 00 14 a8 82
5f f8 09 0d 40 f9 c9 00 00 b4 0a 09 40 f9 29 09 40 f9 5f 01 09 eb 62 00 00 54 00 00 00
14 00 00 00 14 09 00 80 92 09 09 00 f9 00 00 00 14 a8 82 5f f8 0a 25 41 a9 29 0d 40 f9
4b 05 00 91 0b 09 00 f9 28 79 6a f8 09 01 40 b9 69 00 f8 37 29 05 00 11 09 01 00 b9 a8
86 00 f8 00 00 00 14 _SET_IP
```

```
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
_CHECK_FUNCTION
_LOAD_CONST_INLINE
_GUARD_NOS_FLOAT
_BINARY_OP_SUBTRACT_FLOAT
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
```

Machine Code

Python 3.14

```
08 00 00 90 08 01 40 f9 08 89 40 39 48 00 00 37 00 00 00 14 00 00 00 14 a8 82 5f f8 08
05 40 f9 09 00 00 90 29 01 40 f9 1f 01 09 eb 40 00 00 54 00 00 00 14 00 00 00 14 a8 82
5f f8 09 0d 40 f9 c9 00 00 b4 0a 09 40 f9 29 09 40 f9 5f 01 09 eb 62 00 00 54 00 00 00
14 00 00 00 14 09 00 80 92 09 09 00 f9 00 00 00 14 a8 82 5f f8 0a 25 41 a9 29 0d 40 f9
4b 05 00 91 0b 09 00 f9 28 79 6a f8 09 01 40 b9 69 00 f8 37 29 05 00 11 09 01 00 b9 a8
86 00 f8 00 00 00 14 08 00 00 90 08 01 40 f9 88 1e 00 f9 00 00 00 14
```

```
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
_CHECK_FUNCTION
_LOAD_CONST_INLINE
_GUARD_NOS_FLOAT
_BINARY_OP_SUBTRACT_FLOAT
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
```

Machine Code

Python 3.14

```
08 00 00 90 08 01 40 f9 08 89 40 39 48 00 00 37 00 00 00 14 00 00 00 14 a8 82 5f f8 08
05 40 f9 09 00 00 90 29 01 40 f9 1f 01 09 eb 40 00 00 54 00 00 00 14 00 00 00 14 a8 82
5f f8 09 0d 40 f9 c9 00 00 b4 0a 09 40 f9 29 09 40 f9 5f 01 09 eb 62 00 00 54 00 00 00
14 00 00 00 14 09 00 80 92 09 09 00 f9 00 00 00 14 a8 82 5f f8 0a 25 41 a9 29 0d 40 f9
4b 05 00 91 0b 09 00 f9 28 79 6a f8 09 01 40 b9 69 00 f8 37 29 05 00 11 09 01 00 b9 a8
86 00 f8 00 00 00 14 08 00 00 90 08 01 40 f9 88 1e 00 f9 00 00 00 14 _STORE_FAST
```

```
_CHECK_VALIDITY
_LOAD_FAST
_CHECK_FUNCTION
_LOAD_CONST_INLINE
_GUARD_NOS_FLOAT
_BINARY_OP_SUBTRACT_FLOAT
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
```

Machine Code

Python 3.14

```
_CHECK_VALIDITY
_LOAD_FAST
_CHECK_FUNCTION
_LOAD_CONST_INLINE
_GUARD_NOS_FLOAT
_BINARY_OP_SUBTRACT_FLOAT
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
```

```
08 00 00 90 08 01 40 f9 08 89 40 39 48 00 00 37 00 00 00 14 00 00 00 14 a8 82 5f f8 08
05 40 f9 09 00 00 90 29 01 40 f9 1f 01 09 eb 40 00 00 54 00 00 00 14 00 00 00 14 a8 82
5f f8 09 0d 40 f9 c9 00 00 b4 0a 09 40 f9 29 09 40 f9 5f 01 09 eb 62 00 00 54 00 00 00
14 00 00 00 14 09 00 80 92 09 09 00 f9 00 00 00 14 a8 82 5f f8 0a 25 41 a9 29 0d 40 f9
4b 05 00 91 0b 09 00 f9 28 79 6a f8 09 01 40 b9 69 00 f8 37 29 05 00 11 09 01 00 b9 a8
86 00 f8 00 00 00 14 08 00 00 90 08 01 40 f9 88 1e 00 f9 00 00 00 14 a8 8e 5f f8 09 00
00 90 29 01 40 f9 89 2e 29 8b 20 29 40 f9 28 29 00 f9 95 22 00 f9 c0 00 00 b4 08 00 40
b9 88 00 f8 37 08 05 00 71 08 00 00 b9 80 00 00 54 95 22 40 f9 9f 22 00 f9 00 00 00 14
fd 7b bf a9 fd 03 00 91 00 00 00 94 fd 7b c1 a8 95 22 40 f9 9f 22 00 f9 00 00 00 14
```

Machine Code

Python 3.14

```
_LOAD_FAST
_CHECK_FUNCTION
_LOAD_CONST_INLINE
_GUARD_NOS_FLOAT
_BINARY_OP_SUBTRACT_FLOAT
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
```

```
08 00 00 90 08 01 40 f9 08 89 40 39 48 00 00 37 00 00 00 14 00 00 00 14 a8 82 5f f8 08
05 40 f9 09 00 00 90 29 01 40 f9 1f 01 09 eb 40 00 00 54 00 00 00 14 00 00 00 14 a8 82
5f f8 09 0d 40 f9 c9 00 00 b4 0a 09 40 f9 29 09 40 f9 5f 01 09 eb 62 00 00 54 00 00 00
14 00 00 00 14 09 00 80 92 09 09 00 f9 00 00 00 14 a8 82 5f f8 0a 25 41 a9 29 0d 40 f9
4b 05 00 91 0b 09 00 f9 28 79 6a f8 09 01 40 b9 69 00 f8 37 29 05 00 11 09 01 00 b9 a8
86 00 f8 00 00 00 14 08 00 00 90 08 01 40 f9 88 1e 00 f9 00 00 00 14 a8 8e 5f f8 09 00
00 90 29 01 40 f9 89 2e 29 8b 20 29 40 f9 28 29 00 f9 95 22 00 f9 c0 00 00 b4 08 00 40
b9 88 00 f8 37 08 05 00 71 08 00 00 b9 80 00 00 54 95 22 40 f9 9f 22 00 f9 00 00 00 14
fd 7b bf a9 fd 03 00 91 00 00 00 94 fd 7b c1 a8 95 22 40 f9 9f 22 00 f9 00 00 00 14
_CHECK_VALIDITY
```

Machine Code

Python 3.14

```
_LOAD_FAST
_CHECK_FUNCTION
_LOAD_CONST_INLINE
_GUARD_NOS_FLOAT
_BINARY_OP_SUBTRACT_FLOAT
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
```

```
08 00 00 90 08 01 40 f9 08 89 40 39 48 00 00 37 00 00 00 14 00 00 00 14 a8 82 5f f8 08
05 40 f9 09 00 00 90 29 01 40 f9 1f 01 09 eb 40 00 00 54 00 00 00 14 00 00 00 14 a8 82
5f f8 09 0d 40 f9 c9 00 00 b4 0a 09 40 f9 29 09 40 f9 5f 01 09 eb 62 00 00 54 00 00 00
14 00 00 00 14 09 00 80 92 09 09 00 f9 00 00 00 14 a8 82 5f f8 0a 25 41 a9 29 0d 40 f9
4b 05 00 91 0b 09 00 f9 28 79 6a f8 09 01 40 b9 69 00 f8 37 29 05 00 11 09 01 00 b9 a8
86 00 f8 00 00 00 14 08 00 00 90 08 01 40 f9 88 1e 00 f9 00 00 00 14 a8 8e 5f f8 09 00
00 90 29 01 40 f9 89 2e 29 8b 20 29 40 f9 28 29 00 f9 95 22 00 f9 c0 00 00 b4 08 00 40
b9 88 00 f8 37 08 05 00 71 08 00 00 b9 80 00 00 54 95 22 40 f9 9f 22 00 f9 00 00 00 14
fd 7b bf a9 fd 03 00 91 00 00 00 94 fd 7b c1 a8 95 22 40 f9 9f 22 00 f9 00 00 00 14 08
00 00 90 08 01 40 f9 08 89 40 39 48 00 00 37 00 00 00 14 00 00 00 14
```

Machine Code

Python 3.14

```
08 00 00 90 08 01 40 f9 08 89 40 39 48 00 00 37 00 00 00 14 00 00 00 14 a8 82 5f f8 08
05 40 f9 09 00 00 90 29 01 40 f9 1f 01 09 eb 40 00 00 54 00 00 00 14 00 00 00 14 a8 82
5f f8 09 0d 40 f9 c9 00 00 b4 0a 09 40 f9 29 09 40 f9 5f 01 09 eb 62 00 00 54 00 00 00
14 00 00 00 14 09 00 80 92 09 09 00 f9 00 00 00 14 a8 82 5f f8 0a 25 41 a9 29 0d 40 f9
4b 05 00 91 0b 09 00 f9 28 79 6a f8 09 01 40 b9 69 00 f8 37 29 05 00 11 09 01 00 b9 a8
86 00 f8 00 00 00 14 08 00 00 90 08 01 40 f9 88 1e 00 f9 00 00 00 14 a8 8e 5f f8 09 00
00 90 29 01 40 f9 89 2e 29 8b 20 29 40 f9 28 29 00 f9 95 22 00 f9 c0 00 00 b4 08 00 40
b9 88 00 f8 37 08 05 00 71 08 00 00 b9 80 00 00 54 95 22 40 f9 9f 22 00 f9 00 00 00 14
fd 7b bf a9 fd 03 00 91 00 00 00 94 fd 7b c1 a8 95 22 40 f9 9f 22 00 f9 00 00 00 14 08
00 00 90 08 01 40 f9 08 89 40 39 48 00 00 37 00 00 00 14 00 00 00 14 _LOAD_FAST
```

```
_CHECK_FUNCTION
_LOAD_CONST_INLINE
_GUARD_NOS_FLOAT
_BINARY_OP_SUBTRACT_FLOAT
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
```

Machine Code

Python 3.14

```
_CHECK_FUNCTION
_LOAD_CONST_INLINE
_GUARD_NOS_FLOAT
_BINARY_OP_SUBTRACT_FLOAT
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
```

```
08 00 00 90 08 01 40 f9 08 89 40 39 48 00 00 37 00 00 00 14 00 00 00 14 a8 82 5f f8 08
05 40 f9 09 00 00 90 29 01 40 f9 1f 01 09 eb 40 00 00 54 00 00 00 14 00 00 00 14 a8 82
5f f8 09 0d 40 f9 c9 00 00 b4 0a 09 40 f9 29 09 40 f9 5f 01 09 eb 62 00 00 54 00 00 00
14 00 00 00 14 09 00 80 92 09 09 00 f9 00 00 00 14 a8 82 5f f8 0a 25 41 a9 29 0d 40 f9
4b 05 00 91 0b 09 00 f9 28 79 6a f8 09 01 40 b9 69 00 f8 37 29 05 00 11 09 01 00 b9 a8
86 00 f8 00 00 00 14 08 00 00 90 08 01 40 f9 88 1e 00 f9 00 00 00 14 a8 8e 5f f8 09 00
00 90 29 01 40 f9 89 2e 29 8b 20 29 40 f9 28 29 00 f9 95 22 00 f9 c0 00 00 b4 08 00 40
b9 88 00 f8 37 08 05 00 71 08 00 00 b9 80 00 00 54 95 22 40 f9 9f 22 00 f9 00 00 00 14
fd 7b bf a9 fd 03 00 91 00 00 00 94 fd 7b c1 a8 95 22 40 f9 9f 22 00 f9 00 00 00 14 08
00 00 90 08 01 40 f9 08 89 40 39 48 00 00 37 00 00 00 14 00 00 00 14 08 00 00 90 08 01
40 f9 88 2e 28 8b 08 29 40 f9 09 01 40 b9 69 00 f8 37 29 05 00 11 09 01 00 b9 a8 86 00
f8 00 00 00 14
```

Machine Code

Python 3.14

```
_LOAD_CONST_INLINE
_GUARD_NOS_FLOAT
_BINARY_OP_SUBTRACT_FLOAT
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
```

```
08 00 00 90 08 01 40 f9 08 89 40 39 48 00 00 37 00 00 00 14 00 00 00 14 a8 82 5f f8 08
05 40 f9 09 00 00 90 29 01 40 f9 1f 01 09 eb 40 00 00 54 00 00 00 14 00 00 00 14 a8 82
5f f8 09 0d 40 f9 c9 00 00 b4 0a 09 40 f9 29 09 40 f9 5f 01 09 eb 62 00 00 54 00 00 00
14 00 00 00 14 09 00 80 92 09 09 00 f9 00 00 00 14 a8 82 5f f8 0a 25 41 a9 29 0d 40 f9
4b 05 00 91 0b 09 00 f9 28 79 6a f8 09 01 40 b9 69 00 f8 37 29 05 00 11 09 01 00 b9 a8
86 00 f8 00 00 00 14 08 00 00 90 08 01 40 f9 88 1e 00 f9 00 00 00 14 a8 8e 5f f8 09 00
00 90 29 01 40 f9 89 2e 29 8b 20 29 40 f9 28 29 00 f9 95 22 00 f9 c0 00 00 b4 08 00 40
b9 88 00 f8 37 08 05 00 71 08 00 00 b9 80 00 00 54 95 22 40 f9 9f 22 00 f9 00 00 00 14
fd 7b bf a9 fd 03 00 91 00 00 00 94 fd 7b c1 a8 95 22 40 f9 9f 22 00 f9 00 00 00 14 08
00 00 90 08 01 40 f9 08 89 40 39 48 00 00 37 00 00 00 14 00 00 00 14 08 00 00 90 08 01
40 f9 88 2e 28 8b 08 29 40 f9 09 01 40 b9 69 00 f8 37 29 05 00 11 09 01 00 b9 a8 86 00
f8 00 00 00 14 _CHECK_FUNCTION
```

Machine Code

Python 3.14

```
_LOAD_CONST_INLINE
_GUARD_NOS_FLOAT
_BINARY_OP_SUBTRACT_FLOAT
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
```

```
08 00 00 90 08 01 40 f9 08 89 40 39 48 00 00 37 00 00 00 14 00 00 00 14 a8 82 5f f8 08
05 40 f9 09 00 00 90 29 01 40 f9 1f 01 09 eb 40 00 00 54 00 00 00 14 00 00 00 14 a8 82
5f f8 09 0d 40 f9 c9 00 00 b4 0a 09 40 f9 29 09 40 f9 5f 01 09 eb 62 00 00 54 00 00 00
14 00 00 00 14 09 00 80 92 09 09 00 f9 00 00 00 14 a8 82 5f f8 0a 25 41 a9 29 0d 40 f9
4b 05 00 91 0b 09 00 f9 28 79 6a f8 09 01 40 b9 69 00 f8 37 29 05 00 11 09 01 00 b9 a8
86 00 f8 00 00 00 14 08 00 00 90 08 01 40 f9 88 1e 00 f9 00 00 00 14 a8 8e 5f f8 09 00
00 90 29 01 40 f9 89 2e 29 8b 20 29 40 f9 28 29 00 f9 95 22 00 f9 c0 00 00 b4 08 00 40
b9 88 00 f8 37 08 05 00 71 08 00 00 b9 80 00 00 54 95 22 40 f9 9f 22 00 f9 00 00 00 14
fd 7b bf a9 fd 03 00 91 00 00 00 94 fd 7b c1 a8 95 22 40 f9 9f 22 00 f9 00 00 00 14 08
00 00 90 08 01 40 f9 08 89 40 39 48 00 00 37 00 00 00 14 00 00 00 14 08 00 00 90 08 01
40 f9 88 2e 28 8b 08 29 40 f9 09 01 40 b9 69 00 f8 37 29 05 00 11 09 01 00 b9 a8 86 00
f8 00 00 00 14 88 0a 40 f9 08 91 40 b9 09 00 00 90 29 01 40 f9 1f 01 09 6b 41 00 00 54
00 00 00 14 00 00 00 14
```

Machine Code

Python 3.14

```
_GUARD_NOS_FLOAT
_BINARY_OP_SUBTRACT_FLOAT
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
```

```
08 00 00 90 08 01 40 f9 08 89 40 39 48 00 00 37 00 00 00 14 00 00 00 14 a8 82 5f f8 08
05 40 f9 09 00 00 90 29 01 40 f9 1f 01 09 eb 40 00 00 54 00 00 00 14 00 00 00 14 a8 82
5f f8 09 0d 40 f9 c9 00 00 b4 0a 09 40 f9 29 09 40 f9 5f 01 09 eb 62 00 00 54 00 00 00
14 00 00 00 14 09 00 80 92 09 09 00 f9 00 00 00 14 a8 82 5f f8 0a 25 41 a9 29 0d 40 f9
4b 05 00 91 0b 09 00 f9 28 79 6a f8 09 01 40 b9 69 00 f8 37 29 05 00 11 09 01 00 b9 a8
86 00 f8 00 00 00 14 08 00 00 90 08 01 40 f9 88 1e 00 f9 00 00 00 14 a8 8e 5f f8 09 00
00 90 29 01 40 f9 89 2e 29 8b 20 29 40 f9 28 29 00 f9 95 22 00 f9 c0 00 00 b4 08 00 40
b9 88 00 f8 37 08 05 00 71 08 00 00 b9 80 00 00 54 95 22 40 f9 9f 22 00 f9 00 00 00 14
fd 7b bf a9 fd 03 00 91 00 00 00 94 fd 7b c1 a8 95 22 40 f9 9f 22 00 f9 00 00 00 14 08
00 00 90 08 01 40 f9 08 89 40 39 48 00 00 37 00 00 00 14 00 00 00 14 08 00 00 90 08 01
40 f9 88 2e 28 8b 08 29 40 f9 09 01 40 b9 69 00 f8 37 29 05 00 11 09 01 00 b9 a8 86 00
f8 00 00 00 14 88 0a 40 f9 08 91 40 b9 09 00 00 90 29 01 40 f9 1f 01 09 6b 41 00 00 54
00 00 00 14 00 00 00 14 _LOAD_CONST_INLINE
```

Machine Code

Python 3.14

```
_GUARD_NOS_FLOAT
_BINARY_OP_SUBTRACT_FLOAT
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
```

```
08 00 00 90 08 01 40 f9 08 89 40 39 48 00 00 37 00 00 00 14 00 00 00 14 a8 82 5f f8 08
05 40 f9 09 00 00 90 29 01 40 f9 1f 01 09 eb 40 00 00 54 00 00 00 14 00 00 00 14 a8 82
5f f8 09 0d 40 f9 c9 00 00 b4 0a 09 40 f9 29 09 40 f9 5f 01 09 eb 62 00 00 54 00 00 00
14 00 00 00 14 09 00 80 92 09 09 00 f9 00 00 00 14 a8 82 5f f8 0a 25 41 a9 29 0d 40 f9
4b 05 00 91 0b 09 00 f9 28 79 6a f8 09 01 40 b9 69 00 f8 37 29 05 00 11 09 01 00 b9 a8
86 00 f8 00 00 00 14 08 00 00 90 08 01 40 f9 88 1e 00 f9 00 00 00 14 a8 8e 5f f8 09 00
00 90 29 01 40 f9 89 2e 29 8b 20 29 40 f9 28 29 00 f9 95 22 00 f9 c0 00 00 b4 08 00 40
b9 88 00 f8 37 08 05 00 71 08 00 00 b9 80 00 00 54 95 22 40 f9 9f 22 00 f9 00 00 00 14
fd 7b bf a9 fd 03 00 91 00 00 00 94 fd 7b c1 a8 95 22 40 f9 9f 22 00 f9 00 00 00 14 08
00 00 90 08 01 40 f9 08 89 40 39 48 00 00 37 00 00 00 14 00 00 00 14 08 00 00 90 08 01
40 f9 88 2e 28 8b 08 29 40 f9 09 01 40 b9 69 00 f8 37 29 05 00 11 09 01 00 b9 a8 86 00
f8 00 00 00 14 88 0a 40 f9 08 91 40 b9 09 00 00 90 29 01 40 f9 1f 01 09 6b 41 00 00 54
00 00 00 14 00 00 00 14 08 00 00 90 08 01 40 f9 09 01 40 b9 69 00 f8 37 29 05 00 11 09
01 00 b9 a8 86 00 f8 00 00 00 14
```

Machine Code

Python 3.14

```
_BINARY_OP_SUBTRACT_FLOAT
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
```

```
08 00 00 90 08 01 40 f9 08 89 40 39 48 00 00 37 00 00 00 14 00 00 00 14 a8 82 5f f8 08
05 40 f9 09 00 00 90 29 01 40 f9 1f 01 09 eb 40 00 00 54 00 00 00 14 00 00 00 14 a8 82
5f f8 09 0d 40 f9 c9 00 00 b4 0a 09 40 f9 29 09 40 f9 5f 01 09 eb 62 00 00 54 00 00 00
14 00 00 00 14 09 00 80 92 09 09 00 f9 00 00 00 14 a8 82 5f f8 0a 25 41 a9 29 0d 40 f9
4b 05 00 91 0b 09 00 f9 28 79 6a f8 09 01 40 b9 69 00 f8 37 29 05 00 11 09 01 00 b9 a8
86 00 f8 00 00 00 14 08 00 00 90 08 01 40 f9 88 1e 00 f9 00 00 00 14 a8 8e 5f f8 09 00
00 90 29 01 40 f9 89 2e 29 8b 20 29 40 f9 28 29 00 f9 95 22 00 f9 c0 00 00 b4 08 00 40
b9 88 00 f8 37 08 05 00 71 08 00 00 b9 80 00 00 54 95 22 40 f9 9f 22 00 f9 00 00 00 14
fd 7b bf a9 fd 03 00 91 00 00 00 94 fd 7b c1 a8 95 22 40 f9 9f 22 00 f9 00 00 00 14 08
00 00 90 08 01 40 f9 08 89 40 39 48 00 00 37 00 00 00 14 00 00 00 14 08 00 00 90 08 01
40 f9 88 2e 28 8b 08 29 40 f9 09 01 40 b9 69 00 f8 37 29 05 00 11 09 01 00 b9 a8 86 00
f8 00 00 00 14 88 0a 40 f9 08 91 40 b9 09 00 00 90 29 01 40 f9 1f 01 09 6b 41 00 00 54
00 00 00 14 00 00 00 14 08 00 00 90 08 01 40 f9 09 01 40 b9 69 00 f8 37 29 05 00 11 09
01 00 b9 a8 86 00 f8 00 00 00 14 _GUARD_NOS_FLOAT
```

Machine Code

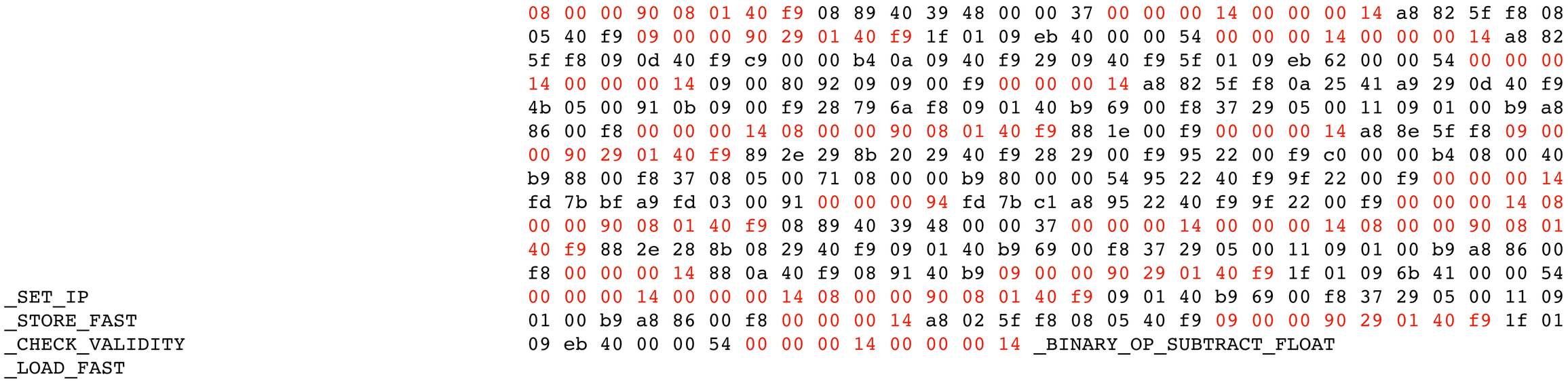
Python 3.14

```
_BINARY_OP_SUBTRACT_FLOAT
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
```

```
08 00 00 90 08 01 40 f9 08 89 40 39 48 00 00 37 00 00 00 14 00 00 00 14 a8 82 5f f8 08
05 40 f9 09 00 00 90 29 01 40 f9 1f 01 09 eb 40 00 00 54 00 00 00 14 00 00 00 14 a8 82
5f f8 09 0d 40 f9 c9 00 00 b4 0a 09 40 f9 29 09 40 f9 5f 01 09 eb 62 00 00 54 00 00 00
14 00 00 00 14 09 00 80 92 09 09 00 f9 00 00 00 14 a8 82 5f f8 0a 25 41 a9 29 0d 40 f9
4b 05 00 91 0b 09 00 f9 28 79 6a f8 09 01 40 b9 69 00 f8 37 29 05 00 11 09 01 00 b9 a8
86 00 f8 00 00 00 14 08 00 00 90 08 01 40 f9 88 1e 00 f9 00 00 00 14 a8 8e 5f f8 09 00
00 90 29 01 40 f9 89 2e 29 8b 20 29 40 f9 28 29 00 f9 95 22 00 f9 c0 00 00 b4 08 00 40
b9 88 00 f8 37 08 05 00 71 08 00 00 b9 80 00 00 54 95 22 40 f9 9f 22 00 f9 00 00 00 14
fd 7b bf a9 fd 03 00 91 00 00 00 94 fd 7b c1 a8 95 22 40 f9 9f 22 00 f9 00 00 00 14 08
00 00 90 08 01 40 f9 08 89 40 39 48 00 00 37 00 00 00 14 00 00 00 14 08 00 00 90 08 01
40 f9 88 2e 28 8b 08 29 40 f9 09 01 40 b9 69 00 f8 37 29 05 00 11 09 01 00 b9 a8 86 00
f8 00 00 00 14 88 0a 40 f9 08 91 40 b9 09 00 00 90 29 01 40 f9 1f 01 09 6b 41 00 00 54
00 00 00 14 00 00 00 14 08 00 00 90 08 01 40 f9 09 01 40 b9 69 00 f8 37 29 05 00 11 09
01 00 b9 a8 86 00 f8 00 00 00 14 a8 02 5f f8 08 05 40 f9 09 00 00 90 29 01 40 f9 1f 01
09 eb 40 00 00 54 00 00 00 14 00 00 00 14
```

Machine Code

Python 3.14



Machine Code

Python 3.14

```
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
```

```
08 00 00 90 08 01 40 f9 08 89 40 39 48 00 00 37 00 00 00 14 00 00 00 14 a8 82 5f f8 08
05 40 f9 09 00 00 90 29 01 40 f9 1f 01 09 eb 40 00 00 54 00 00 00 14 00 00 00 14 a8 82
5f f8 09 0d 40 f9 c9 00 00 b4 0a 09 40 f9 29 09 40 f9 5f 01 09 eb 62 00 00 54 00 00 00
14 00 00 00 14 09 00 80 92 09 09 00 f9 00 00 00 14 a8 82 5f f8 0a 25 41 a9 29 0d 40 f9
4b 05 00 91 0b 09 00 f9 28 79 6a f8 09 01 40 b9 69 00 f8 37 29 05 00 11 09 01 00 b9 a8
86 00 f8 00 00 00 14 08 00 00 90 08 01 40 f9 88 1e 00 f9 00 00 00 14 a8 8e 5f f8 09 00
00 90 29 01 40 f9 89 2e 29 8b 20 29 40 f9 28 29 00 f9 95 22 00 f9 c0 00 00 b4 08 00 40
b9 88 00 f8 37 08 05 00 71 08 00 00 b9 80 00 00 54 95 22 40 f9 9f 22 00 f9 00 00 00 14
fd 7b bf a9 fd 03 00 91 00 00 00 94 fd 7b c1 a8 95 22 40 f9 9f 22 00 f9 00 00 00 14 08
00 00 90 08 01 40 f9 08 89 40 39 48 00 00 37 00 00 00 14 00 00 00 14 08 00 00 90 08 01
40 f9 88 2e 28 8b 08 29 40 f9 09 01 40 b9 69 00 f8 37 29 05 00 11 09 01 00 b9 a8 86 00
f8 00 00 00 14 88 0a 40 f9 08 91 40 b9 09 00 00 90 29 01 40 f9 1f 01 09 6b 41 00 00 54
00 00 00 14 00 00 00 14 08 00 00 90 08 01 40 f9 09 01 40 b9 69 00 f8 37 29 05 00 11 09
01 00 b9 a8 86 00 f8 00 00 00 14 a8 02 5f f8 08 05 40 f9 09 00 00 90 29 01 40 f9 1f 01
09 eb 40 00 00 54 00 00 00 14 00 00 00 14 fd 7b bf a9 fd 03 00 91 a1 8e 5f f8 f3 03 15
aa 60 8e 5f f8 00 08 40 fd 21 08 40 fd 00 38 61 1e 00 00 00 94 80 00 00 b4 60 02 00 f9
fd 7b c1 a8 00 00 00 14 f5 03 13 aa fd 7b c1 a8 00 00 00 14
```

Machine Code

Python 3.14

```
08 00 00 90 08 01 40 f9 08 89 40 39 48 00 00 37 00 00 00 14 00 00 00 14 a8 82 5f f8 08
05 40 f9 09 00 00 90 29 01 40 f9 1f 01 09 eb 40 00 00 54 00 00 00 14 00 00 00 14 a8 82
5f f8 09 0d 40 f9 c9 00 00 b4 0a 09 40 f9 29 09 40 f9 5f 01 09 eb 62 00 00 54 00 00 00
14 00 00 00 14 09 00 80 92 09 09 00 f9 00 00 00 14 a8 82 5f f8 0a 25 41 a9 29 0d 40 f9
4b 05 00 91 0b 09 00 f9 28 79 6a f8 09 01 40 b9 69 00 f8 37 29 05 00 11 09 01 00 b9 a8
86 00 f8 00 00 00 14 08 00 00 90 08 01 40 f9 88 1e 00 f9 00 00 00 14 a8 8e 5f f8 09 00
00 90 29 01 40 f9 89 2e 29 8b 20 29 40 f9 28 29 00 f9 95 22 00 f9 c0 00 00 b4 08 00 40
b9 88 00 f8 37 08 05 00 71 08 00 00 b9 80 00 00 54 95 22 40 f9 9f 22 00 f9 00 00 00 14
fd 7b bf a9 fd 03 00 91 00 00 00 94 fd 7b c1 a8 95 22 40 f9 9f 22 00 f9 00 00 00 14 08
00 00 90 08 01 40 f9 08 89 40 39 48 00 00 37 00 00 00 14 00 00 00 14 08 00 00 90 08 01
40 f9 88 2e 28 8b 08 29 40 f9 09 01 40 b9 69 00 f8 37 29 05 00 11 09 01 00 b9 a8 86 00
f8 00 00 00 14 88 0a 40 f9 08 91 40 b9 09 00 00 90 29 01 40 f9 1f 01 09 6b 41 00 00 54
00 00 00 14 00 00 00 14 08 00 00 90 08 01 40 f9 09 01 40 b9 69 00 f8 37 29 05 00 11 09
01 00 b9 a8 86 00 f8 00 00 00 14 a8 02 5f f8 08 05 40 f9 09 00 00 90 29 01 40 f9 1f 01
09 eb 40 00 00 54 00 00 00 14 00 00 00 14 fd 7b bf a9 fd 03 00 91 a1 8e 5f f8 f3 03 15
aa 60 8e 5f f8 00 08 40 fd 21 08 40 fd 00 38 61 1e 00 00 00 94 80 00 00 b4 60 02 00 f9
fd 7b c1 a8 00 00 00 14 f5 03 13 aa fd 7b c1 a8 00 00 00 14 _SET_IP
```

```
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
```

Machine Code

Python 3.14

```
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
```

```
08 00 00 90 08 01 40 f9 08 89 40 39 48 00 00 37 00 00 00 14 00 00 00 14 a8 82 5f f8 08
05 40 f9 09 00 00 90 29 01 40 f9 1f 01 09 eb 40 00 00 54 00 00 00 14 00 00 00 14 a8 82
5f f8 09 0d 40 f9 c9 00 00 b4 0a 09 40 f9 29 09 40 f9 5f 01 09 eb 62 00 00 54 00 00 00
14 00 00 00 14 09 00 80 92 09 09 00 f9 00 00 00 14 a8 82 5f f8 0a 25 41 a9 29 0d 40 f9
4b 05 00 91 0b 09 00 f9 28 79 6a f8 09 01 40 b9 69 00 f8 37 29 05 00 11 09 01 00 b9 a8
86 00 f8 00 00 00 14 08 00 00 90 08 01 40 f9 88 1e 00 f9 00 00 00 14 a8 8e 5f f8 09 00
00 90 29 01 40 f9 89 2e 29 8b 20 29 40 f9 28 29 00 f9 95 22 00 f9 c0 00 00 b4 08 00 40
b9 88 00 f8 37 08 05 00 71 08 00 00 b9 80 00 00 54 95 22 40 f9 9f 22 00 f9 00 00 00 14
fd 7b bf a9 fd 03 00 91 00 00 00 94 fd 7b c1 a8 95 22 40 f9 9f 22 00 f9 00 00 00 14 08
00 00 90 08 01 40 f9 08 89 40 39 48 00 00 37 00 00 00 14 00 00 00 14 08 00 00 90 08 01
40 f9 88 2e 28 8b 08 29 40 f9 09 01 40 b9 69 00 f8 37 29 05 00 11 09 01 00 b9 a8 86 00
f8 00 00 00 14 88 0a 40 f9 08 91 40 b9 09 00 00 90 29 01 40 f9 1f 01 09 6b 41 00 00 54
00 00 00 14 00 00 00 14 08 00 00 90 08 01 40 f9 09 01 40 b9 69 00 f8 37 29 05 00 11 09
01 00 b9 a8 86 00 f8 00 00 00 14 a8 02 5f f8 08 05 40 f9 09 00 00 90 29 01 40 f9 1f 01
09 eb 40 00 00 54 00 00 00 14 00 00 00 14 fd 7b bf a9 fd 03 00 91 a1 8e 5f f8 f3 03 15
aa 60 8e 5f f8 00 08 40 fd 21 08 40 fd 00 38 61 1e 00 00 00 94 80 00 00 b4 60 02 00 f9
fd 7b c1 a8 00 00 00 14 f5 03 13 aa fd 7b c1 a8 00 00 00 14 08 00 00 90 08 01 40 f9 88
1e 00 f9 00 00 00 14
```

Machine Code

Python 3.14

```
                                                08 00 00 90 08 01 40 f9 08 89 40 39 48 00 00 37 00 00 00 14 00 00 00 14 a8 82 5f f8 08
                                                05 40 f9 09 00 00 90 29 01 40 f9 1f 01 09 eb 40 00 00 54 00 00 00 14 00 00 00 14 a8 82
                                                5f f8 09 0d 40 f9 c9 00 00 b4 0a 09 40 f9 29 09 40 f9 5f 01 09 eb 62 00 00 54 00 00 00
                                                14 00 00 00 14 09 00 80 92 09 09 00 f9 00 00 00 14 a8 82 5f f8 0a 25 41 a9 29 0d 40 f9
                                                4b 05 00 91 0b 09 00 f9 28 79 6a f8 09 01 40 b9 69 00 f8 37 29 05 00 11 09 01 00 b9 a8
                                                86 00 f8 00 00 00 14 08 00 00 90 08 01 40 f9 88 1e 00 f9 00 00 00 14 a8 8e 5f f8 09 00
                                                00 90 29 01 40 f9 89 2e 29 8b 20 29 40 f9 28 29 00 f9 95 22 00 f9 c0 00 00 b4 08 00 40
                                                b9 88 00 f8 37 08 05 00 71 08 00 00 b9 80 00 00 54 95 22 40 f9 9f 22 00 f9 00 00 00 14
                                                fd 7b bf a9 fd 03 00 91 00 00 00 94 fd 7b c1 a8 95 22 40 f9 9f 22 00 f9 00 00 00 14 08
                                                00 00 90 08 01 40 f9 08 89 40 39 48 00 00 37 00 00 00 14 00 00 00 14 08 00 00 90 08 01
                                                40 f9 88 2e 28 8b 08 29 40 f9 09 01 40 b9 69 00 f8 37 29 05 00 11 09 01 00 b9 a8 86 00
                                                f8 00 00 00 14 88 0a 40 f9 08 91 40 b9 09 00 00 90 29 01 40 f9 1f 01 09 6b 41 00 00 54
                                                00 00 00 14 00 00 00 14 08 00 00 90 08 01 40 f9 09 01 40 b9 69 00 f8 37 29 05 00 11 09
                                                01 00 b9 a8 86 00 f8 00 00 00 14 a8 02 5f f8 08 05 40 f9 09 00 00 90 29 01 40 f9 1f 01
_CHECK_VALIDITY                                 09 eb 40 00 00 54 00 00 00 14 00 00 00 14 fd 7b bf a9 fd 03 00 91 a1 8e 5f f8 f3 03 15
_LOAD_FAST                                      aa 60 8e 5f f8 00 08 40 fd 21 08 40 fd 00 38 61 1e 00 00 00 94 80 00 00 b4 60 02 00 f9
                                                fd 7b c1 a8 00 00 00 14 f5 03 13 aa fd 7b c1 a8 00 00 00 14 08 00 00 90 08 01 40 f9 88
                                                1e 00 f9 00 00 00 14 _STORE_FAST
```

Machine Code

Python 3.14

```
_CHECK_VALIDITY
_LOAD_FAST
```

```
08 00 00 90 08 01 40 f9 08 89 40 39 48 00 00 37 00 00 00 14 00 00 00 14 a8 82 5f f8 08
05 40 f9 09 00 00 90 29 01 40 f9 1f 01 09 eb 40 00 00 54 00 00 00 14 00 00 00 14 a8 82
5f f8 09 0d 40 f9 c9 00 00 b4 0a 09 40 f9 29 09 40 f9 5f 01 09 eb 62 00 00 54 00 00 00
14 00 00 00 14 09 00 80 92 09 09 00 f9 00 00 00 14 a8 82 5f f8 0a 25 41 a9 29 0d 40 f9
4b 05 00 91 0b 09 00 f9 28 79 6a f8 09 01 40 b9 69 00 f8 37 29 05 00 11 09 01 00 b9 a8
86 00 f8 00 00 00 14 08 00 00 90 08 01 40 f9 88 1e 00 f9 00 00 00 14 a8 8e 5f f8 09 00
00 90 29 01 40 f9 89 2e 29 8b 20 29 40 f9 28 29 00 f9 95 22 00 f9 c0 00 00 b4 08 00 40
b9 88 00 f8 37 08 05 00 71 08 00 00 b9 80 00 00 54 95 22 40 f9 9f 22 00 f9 00 00 00 14
fd 7b bf a9 fd 03 00 91 00 00 00 94 fd 7b c1 a8 95 22 40 f9 9f 22 00 f9 00 00 00 14 08
00 00 90 08 01 40 f9 08 89 40 39 48 00 00 37 00 00 00 14 00 00 00 14 08 00 00 90 08 01
40 f9 88 2e 28 8b 08 29 40 f9 09 01 40 b9 69 00 f8 37 29 05 00 11 09 01 00 b9 a8 86 00
f8 00 00 00 14 88 0a 40 f9 08 91 40 b9 09 00 00 90 29 01 40 f9 1f 01 09 6b 41 00 00 54
00 00 00 14 00 00 00 14 08 00 00 90 08 01 40 f9 09 01 40 b9 69 00 f8 37 29 05 00 11 09
01 00 b9 a8 86 00 f8 00 00 00 14 a8 02 5f f8 08 05 40 f9 09 00 00 90 29 01 40 f9 1f 01
09 eb 40 00 00 54 00 00 00 14 00 00 00 14 fd 7b bf a9 fd 03 00 91 a1 8e 5f f8 f3 03 15
aa 60 8e 5f f8 00 08 40 fd 21 08 40 fd 00 38 61 1e 00 00 00 94 80 00 00 b4 60 02 00 f9
fd 7b c1 a8 00 00 00 14 f5 03 13 aa fd 7b c1 a8 00 00 00 14 08 00 00 90 08 01 40 f9 88
1e 00 f9 00 00 00 14 a8 8e 5f f8 09 00 00 90 29 01 40 f9 89 2e 29 8b 20 29 40 f9 28 29
00 f9 95 22 00 f9 c0 00 00 b4 08 00 40 b9 88 00 f8 37 08 05 00 71 08 00 00 b9 80 00 00
54 95 22 40 f9 9f 22 00 f9 00 00 00 14 fd 7b bf a9 fd 03 00 91 00 00 00 94 fd 7b c1 a8
95 22 40 f9 9f 22 00 f9 00 00 00 14
```

Machine Code

Python 3.14

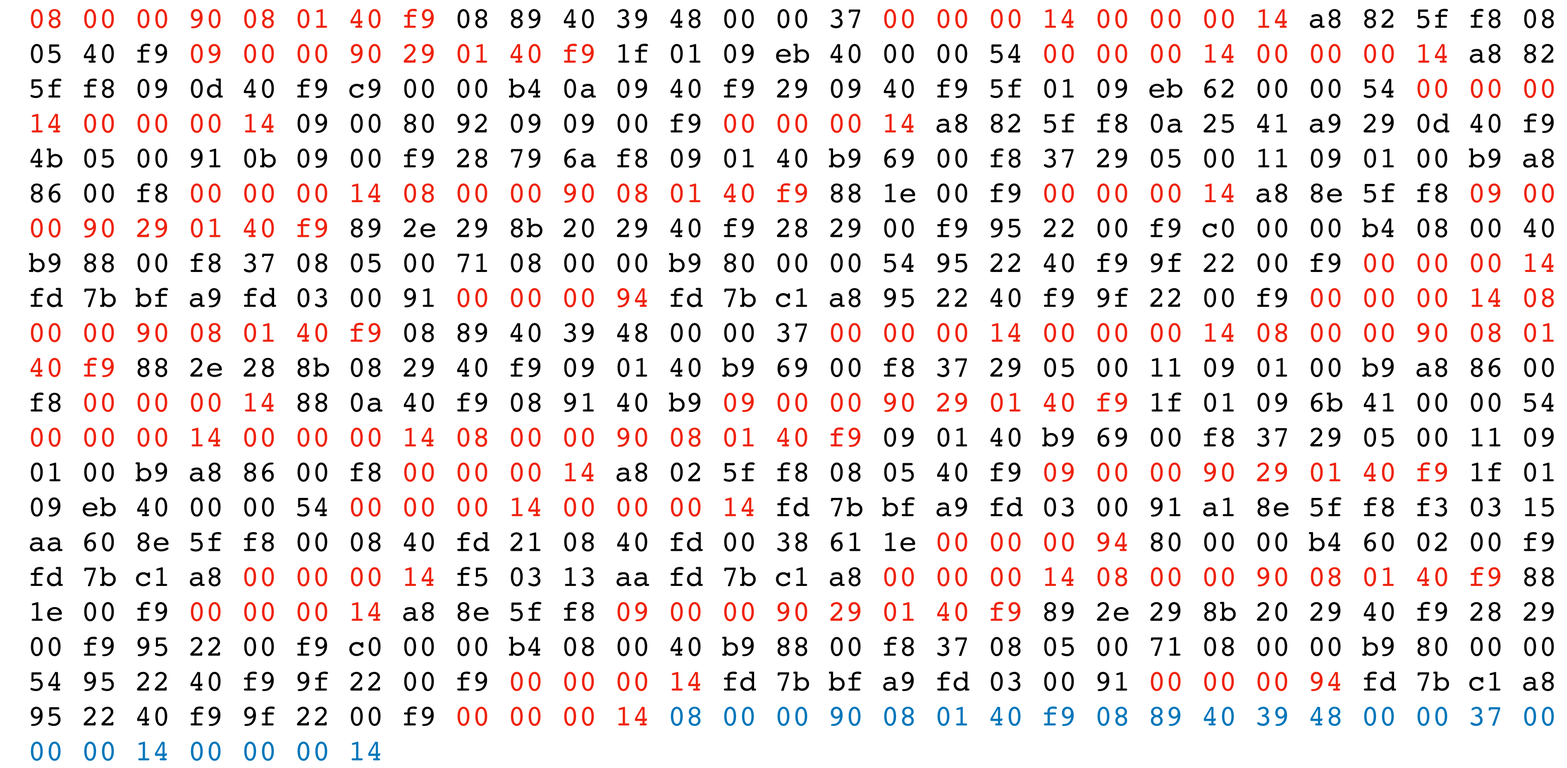
_LOAD_FAST

```
08 00 00 90 08 01 40 f9 08 89 40 39 48 00 00 37 00 00 00 14 00 00 00 14 a8 82 5f f8 08
05 40 f9 09 00 00 90 29 01 40 f9 1f 01 09 eb 40 00 00 54 00 00 00 14 00 00 00 14 a8 82
5f f8 09 0d 40 f9 c9 00 00 b4 0a 09 40 f9 29 09 40 f9 5f 01 09 eb 62 00 00 54 00 00 00
14 00 00 00 14 09 00 80 92 09 09 00 f9 00 00 00 14 a8 82 5f f8 0a 25 41 a9 29 0d 40 f9
4b 05 00 91 0b 09 00 f9 28 79 6a f8 09 01 40 b9 69 00 f8 37 29 05 00 11 09 01 00 b9 a8
86 00 f8 00 00 00 14 08 00 00 90 08 01 40 f9 88 1e 00 f9 00 00 00 14 a8 8e 5f f8 09 00
00 90 29 01 40 f9 89 2e 29 8b 20 29 40 f9 28 29 00 f9 95 22 00 f9 c0 00 00 b4 08 00 40
b9 88 00 f8 37 08 05 00 71 08 00 00 b9 80 00 00 54 95 22 40 f9 9f 22 00 f9 00 00 00 14
fd 7b bf a9 fd 03 00 91 00 00 00 94 fd 7b c1 a8 95 22 40 f9 9f 22 00 f9 00 00 00 14 08
00 00 90 08 01 40 f9 08 89 40 39 48 00 00 37 00 00 00 14 00 00 00 14 08 00 00 90 08 01
40 f9 88 2e 28 8b 08 29 40 f9 09 01 40 b9 69 00 f8 37 29 05 00 11 09 01 00 b9 a8 86 00
f8 00 00 00 14 88 0a 40 f9 08 91 40 b9 09 00 00 90 29 01 40 f9 1f 01 09 6b 41 00 00 54
00 00 00 14 00 00 00 14 08 00 00 90 08 01 40 f9 09 01 40 b9 69 00 f8 37 29 05 00 11 09
01 00 b9 a8 86 00 f8 00 00 00 14 a8 02 5f f8 08 05 40 f9 09 00 00 90 29 01 40 f9 1f 01
09 eb 40 00 00 54 00 00 00 14 00 00 00 14 fd 7b bf a9 fd 03 00 91 a1 8e 5f f8 f3 03 15
aa 60 8e 5f f8 00 08 40 fd 21 08 40 fd 00 38 61 1e 00 00 00 94 80 00 00 b4 60 02 00 f9
fd 7b c1 a8 00 00 00 14 f5 03 13 aa fd 7b c1 a8 00 00 00 14 08 00 00 90 08 01 40 f9 88
1e 00 f9 00 00 00 14 a8 8e 5f f8 09 00 00 90 29 01 40 f9 89 2e 29 8b 20 29 40 f9 28 29
00 f9 95 22 00 f9 c0 00 00 b4 08 00 40 b9 88 00 f8 37 08 05 00 71 08 00 00 b9 80 00 00
54 95 22 40 f9 9f 22 00 f9 00 00 00 14 fd 7b bf a9 fd 03 00 91 00 00 00 94 fd 7b c1 a8
95 22 40 f9 9f 22 00 f9 00 00 00 14 _CHECK_VALIDITY
```

Machine Code

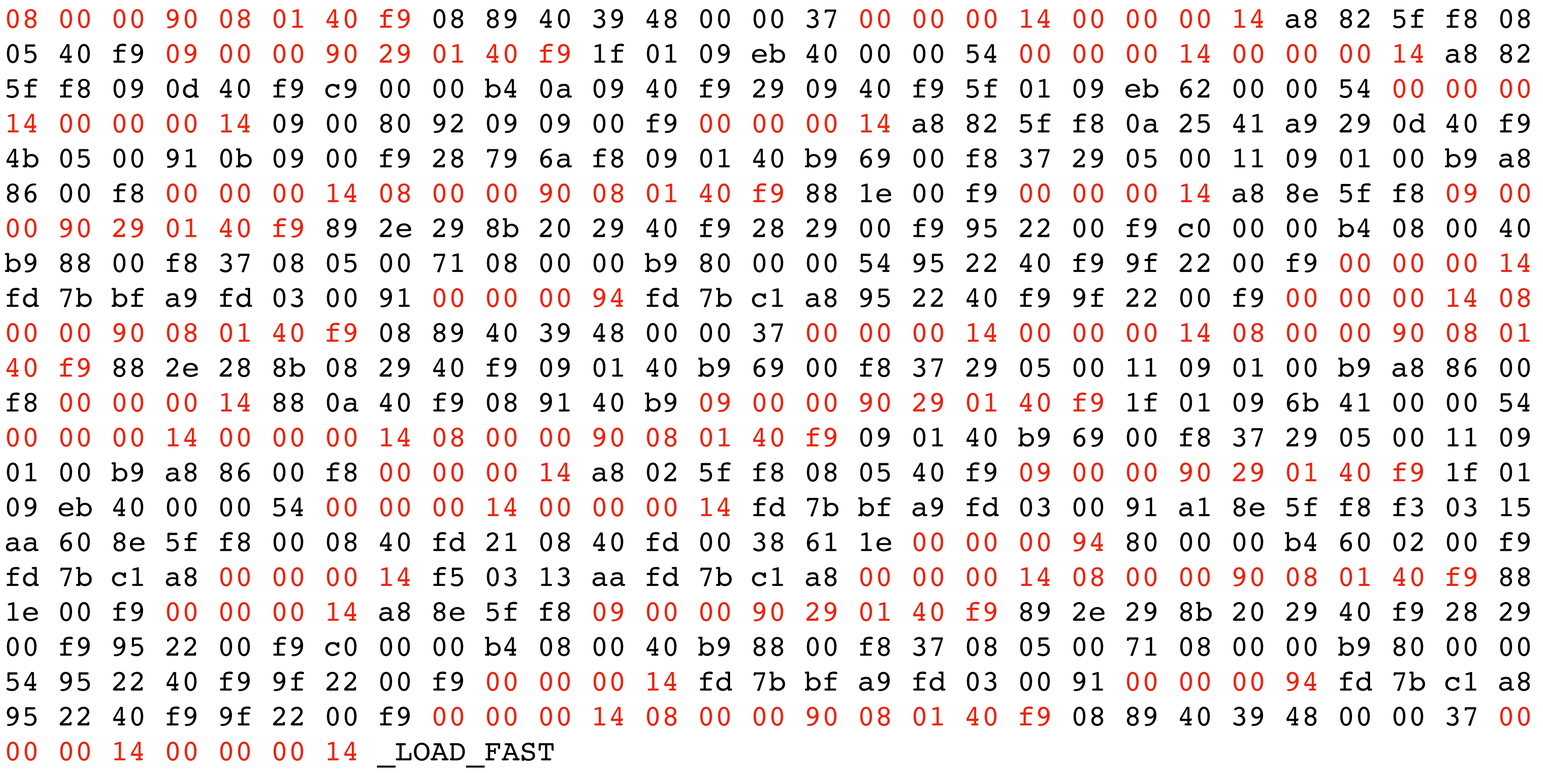
Python 3.14

_LOAD_FAST



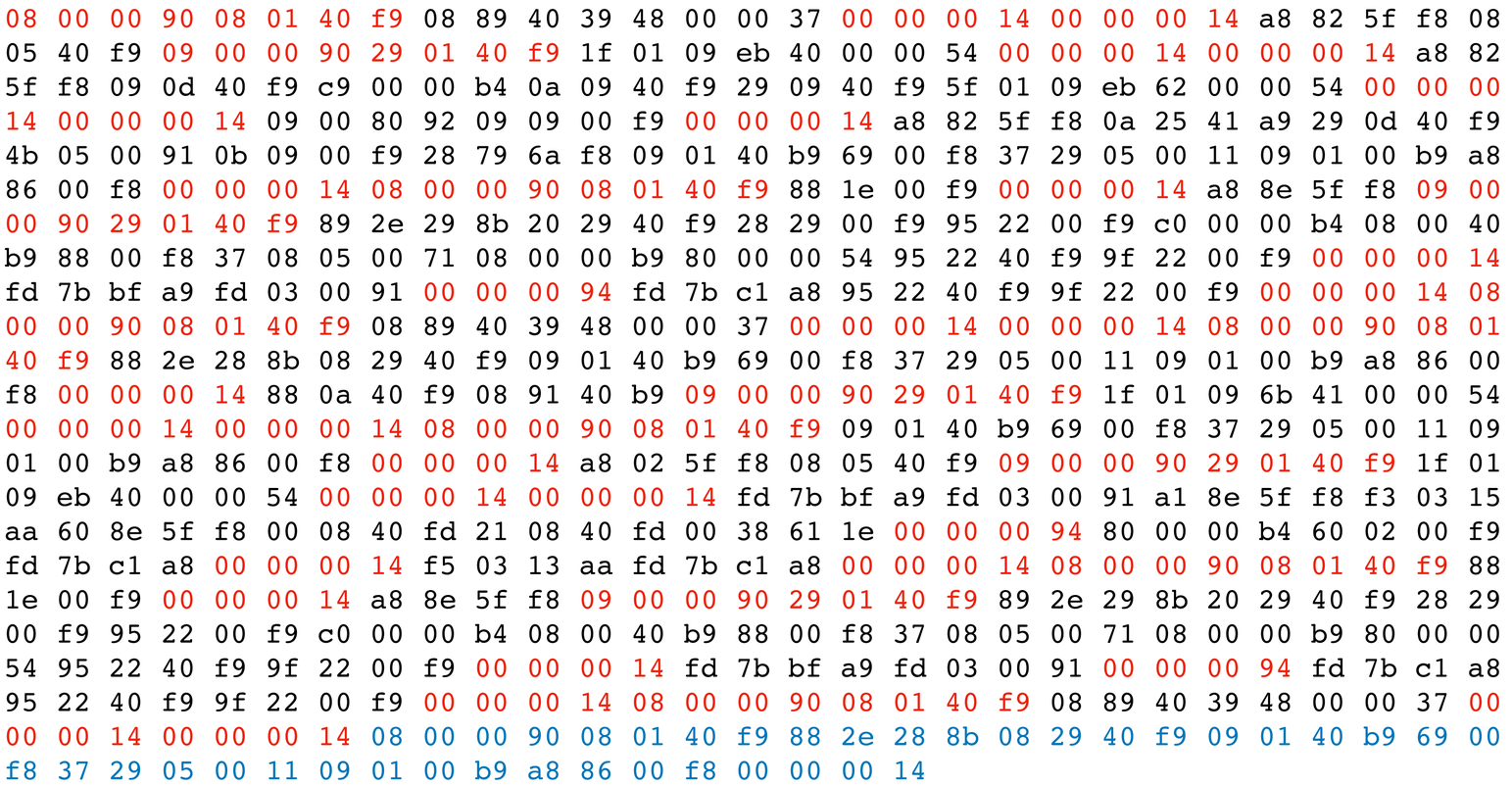
Machine Code

Python 3.14



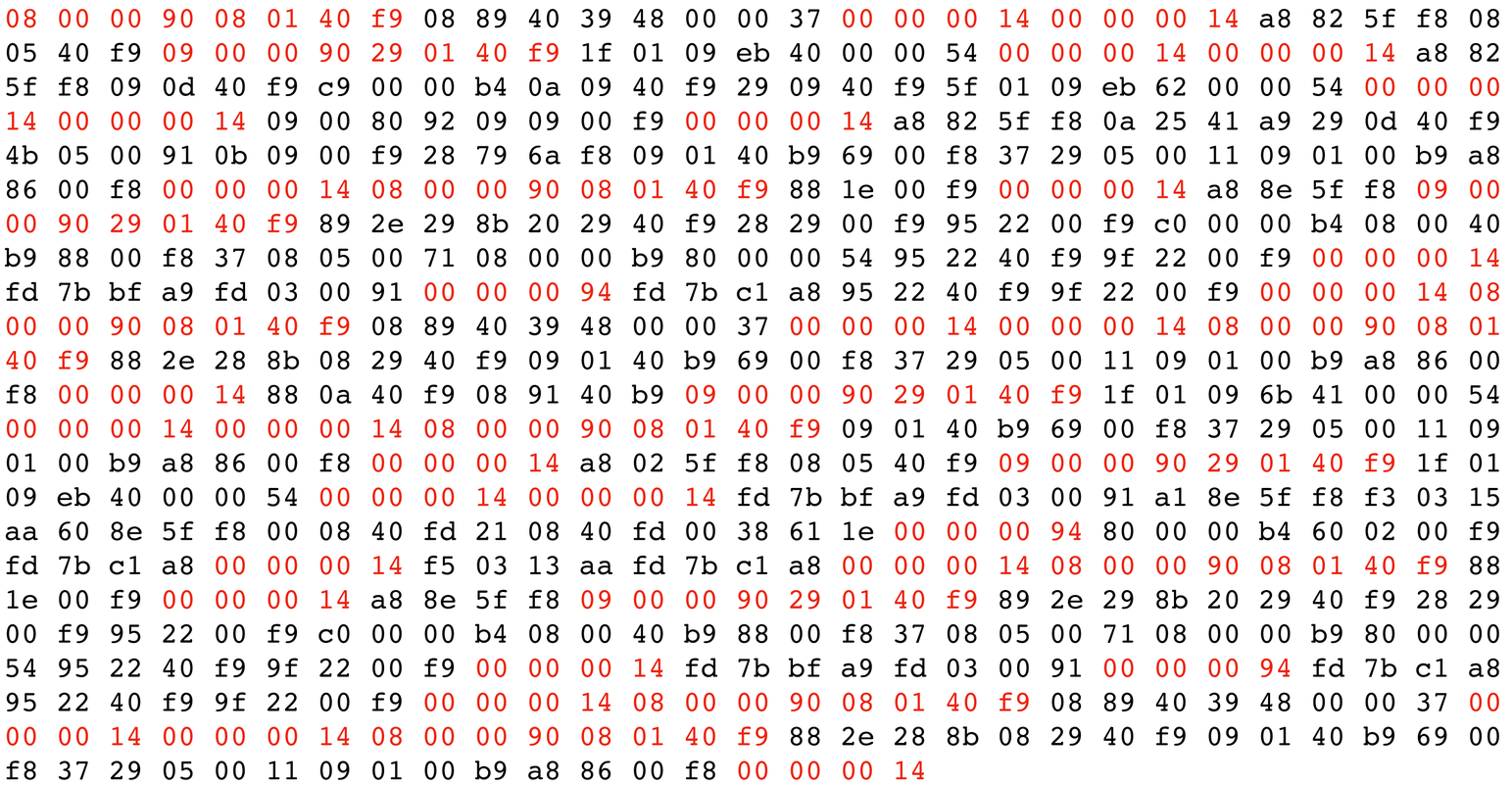
Machine Code

Python 3.14



Machine Code

Python 3.14



Machine Code

Python 3.14

```
08 12 34 95 68 71 48 f9 08 89 40 39 48 00 00 37 9a bc de 14 f0 12 ba 14 a8 82 5f f8 08
05 40 f9 39 45 67 98 29 91 4a f9 1f 01 09 eb 40 00 00 54 bc de f0 14 12 34 56 14 a8 82
5f f8 09 0d 40 f9 c9 00 00 b4 0a 09 40 f9 29 09 40 f9 5f 01 09 eb 62 00 00 54 78 9a bc
14 de f0 12 14 09 00 80 92 09 09 00 f9 34 56 78 14 a8 82 5f f8 0a 25 41 a9 29 0d 40 f9
4b 05 00 91 0b 09 00 f9 28 79 6a f8 09 01 40 b9 69 00 f8 37 29 05 00 11 09 01 00 b9 a8
86 00 f8 9a bc de 14 f8 01 23 94 58 61 47 f9 88 1e 00 f9 89 ab cd 14 a8 8e 5f f8 e9 f0
12 93 29 41 45 f9 89 2e 29 8b 20 29 40 f9 28 29 00 f9 95 22 00 f9 c0 00 00 b4 08 00 40
b9 88 00 f8 37 08 05 00 71 08 00 00 b9 80 00 00 54 95 22 40 f9 9f 22 00 f9 67 89 ab 14
fd 7b bf a9 fd 03 00 91 cd ef 01 94 fd 7b c1 a8 95 22 40 f9 9f 22 00 f9 23 45 67 14 88
9a bc 9d e8 f1 40 f9 08 89 40 39 48 00 00 37 12 34 56 14 78 9a bc 14 d8 ef 01 92 38 41
45 f9 88 2e 28 8b 08 29 40 f9 09 01 40 b9 69 00 f8 37 29 05 00 11 09 01 00 b9 a8 86 00
f8 67 89 db 14 88 0a 40 f9 08 91 40 b9 a9 bc de 9f 29 g1 40 f9 1f 01 09 6b 41 00 00 54
12 34 56 14 78 9a bc 14 d8 ef 01 92 38 41 45 f9 09 01 40 b9 69 00 f8 37 29 05 00 11 09
01 00 b9 a8 86 00 f8 67 89 ab 14 a8 02 5f f8 08 05 40 f9 c9 de f0 91 29 21 43 f9 1f 01
09 eb 40 00 00 54 45 67 89 14 ab cd ef 14 fd 7b bf a9 fd 03 00 91 a1 8e 5f f8 f3 03 15
aa 60 8e 5f f8 00 08 40 fd 21 08 40 fd 00 38 61 1e 01 23 45 94 80 00 00 b4 60 02 00 f9
fd 7b c1 a8 67 89 ab 14 f5 03 13 aa fd 7b c1 a8 cd ef 01 14 28 34 56 97 88 91 4a f9 88
1e 00 f9 bc de fg 14 a8 8e 5f f8 09 12 34 95 29 61 47 f9 89 2e 29 8b 20 29 40 f9 28 29
00 f9 95 22 00 f9 c0 00 00 b4 08 00 40 b9 88 00 f8 37 08 05 00 71 08 00 00 b9 80 00 00
54 95 22 40 f9 9f 22 00 f9 89 ab cd 14 fd 7b bf a9 fd 03 00 91 ef 01 23 94 fd 7b c1 a8
95 22 40 f9 9f 22 00 f9 45 67 89 14 a8 bc de 9f 08 11 42 f9 08 89 40 39 48 00 00 37 34
56 78 14 9a bc de 14 f8 01 23 94 58 61 47 f9 88 2e 28 8b 08 29 40 f9 09 01 40 b9 69 00
f8 37 29 05 00 11 09 01 00 b9 a8 86 00 f8 89 ab cd 14
```

Machine Code

Python 3.14

```
08 12 34 95 68 71 48 f9 08 89 40 39 48 00 00 37 9a bc de 14 f0 12 ba 14 a8 82 5f f8 08
05 40 f9 39 45 67 98 29 91 4a f9 1f 01 09 eb 40 00 00 54 bc de f0 14 12 34 56 14 a8 82
5f f8 09 0d 40 f9 c9 00 00 b4 0a 09 40 f9 29 09 40 f9 5f 01 09 eb 62 00 00 54 78 9a bc
14 de f0 12 14 09 00 80 92 09 09 00 f9 34 56 78 14 a8 82 5f f8 0a 25 41 a9 29 0d 40 f9
4b 05 00 91 0b 09 00 f9 28 79 6a f8 09 01 40 b9 69 00 f8 37 29 05 00 11 09 01 00 b9 a8
86 00 f8 9a bc de 14 f8 01 23 94 58 61 47 f9 88 1e 00 f9 89 ab cd 14 a8 8e 5f f8 e9 f0
12 93 29 41 45 f9 89 2e 29 8b 20 29 40 f9 28 29 00 f9 95 22 00 f9 c0 00 00 b4 08 00 40
b9 88 00 f8 37 08 05 00 71 08 00 00 b9 80 00 00 54 95 22 40 f9 9f 22 00 f9 67 89 ab 14
fd 7b bf a9 fd 03 00 91 cd ef 01 94 fd 7b c1 a8 95 22 40 f9 9f 22 00 f9 23 45 67 14 88
9a bc 9d e8 f1 40 f9 08 89 40 39 48 00 00 37 12 34 56 14 78 9a bc 14 d8 ef 01 92 38 41
45 f9 88 2e 28 8b 08 29 40 f9 09 01 40 b9 69 00 f8 37 29 05 00 11 09 01 00 b9 a8 86 00
f8 67 89 db 14 88 0a 40 f9 08 91 40 b9 a9 bc de 9f 29 g1 40 f9 1f 01 09 6b 41 00 00 54
12 34 56 14 78 9a bc 14 d8 ef 01 92 38 41 45 f9 09 01 40 b9 69 00 f8 37 29 05 00 11 09
01 00 b9 a8 86 00 f8 67 89 ab 14 a8 02 5f f8 08 05 40 f9 c9 de f0 91 29 21 43 f9 1f 01
09 eb 40 00 00 54 45 67 89 14 ab cd ef 14 fd 7b bf a9 fd 03 00 91 a1 8e 5f f8 f3 03 15
aa 60 8e 5f f8 00 08 40 fd 21 08 40 fd 00 38 61 1e 01 23 45 94 80 00 00 b4 60 02 00 f9
fd 7b c1 a8 67 89 ab 14 f5 03 13 aa fd 7b c1 a8 cd ef 01 14 28 34 56 97 88 91 4a f9 88
1e 00 f9 bc de fg 14 a8 8e 5f f8 09 12 34 95 29 61 47 f9 89 2e 29 8b 20 29 40 f9 28 29
00 f9 95 22 00 f9 c0 00 00 b4 08 00 40 b9 88 00 f8 37 08 05 00 71 08 00 00 b9 80 00 00
54 95 22 40 f9 9f 22 00 f9 89 ab cd 14 fd 7b bf a9 fd 03 00 91 ef 01 23 94 fd 7b c1 a8
95 22 40 f9 9f 22 00 f9 45 67 89 14 a8 bc de 9f 08 11 42 f9 08 89 40 39 48 00 00 37 34
56 78 14 9a bc de 14 f8 01 23 94 58 61 47 f9 88 2e 28 8b 08 29 40 f9 09 01 40 b9 69 00
f8 37 29 05 00 11 09 01 00 b9 a8 86 00 f8 89 ab cd 14
```

Machine Code

Python 3.14

```
FOR_ITER_LIST
STORE_FAST
LOAD_FAST
LOAD_GLOBAL_MODULE
BINARY_OP_SUBTRACT_FLOAT
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL_PY_EXACT_ARGS
BINARY_OP_ADD_INT
STORE_FAST
LOAD_GLOBAL_BUILTIN
LOAD_FAST
LOAD_FAST
BINARY_OP_TRUE_DIVIDE_INT_FLOAT
CALL_BUILTIN_FAST_WITH_KEYWORDS
POP_TOP
JUMP_BACKWARD
```

Region Selection

```
FOR_ITER_LIST
STORE_FAST
LOAD_FAST
LOAD_GLOBAL_MODULE
BINARY_OP_SUBTRACT_FLOAT
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL_PY_EXACT_ARGS
BINARY_OP_ADD_INT
STORE_FAST
LOAD_GLOBAL_BUILTIN
LOAD_FAST
LOAD_FAST
BINARY_OP_TRUE_DIVIDE_INT_FLOAT
CALL_BUILTIN_FAST_WITH_KEYWORDS
POP_TOP
JUMP_BACKWARD
```

Region Selection

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
FOR_ITER_LIST
STORE_FAST
LOAD_FAST
LOAD_GLOBAL_MODULE
BINARY_OP_SUBTRACT_FLOAT
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL_PY_EXACT_ARGS
BINARY_OP_ADD_INT
STORE_FAST
LOAD_GLOBAL_BUILTIN
LOAD_FAST
LOAD_FAST
BINARY_OP_TRUE_DIVIDE_INT_FLOAT
CALL_BUILTIN_FAST_WITH_KEYWORDS
POP_TOP
JUMP_BACKWARD
```

Region Selection

What?

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
FOR_ITER_LIST
STORE_FAST
LOAD_FAST
LOAD_GLOBAL_MODULE
BINARY_OP_SUBTRACT_FLOAT
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL_PY_EXACT_ARGS
BINARY_OP_ADD_INT
STORE_FAST
LOAD_GLOBAL_BUILTIN
LOAD_FAST
LOAD_FAST
BINARY_OP_TRUE_DIVIDE_INT_FLOAT
CALL_BUILTIN_FAST_WITH_KEYWORDS
POP_TOP
JUMP_BACKWARD
```

Region Selection

What? When?

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
FOR_ITER_LIST
STORE_FAST
LOAD_FAST
LOAD_GLOBAL_MODULE
BINARY_OP_SUBTRACT_FLOAT
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL_PY_EXACT_ARGS
BINARY_OP_ADD_INT
STORE_FAST
LOAD_GLOBAL_BUILTIN
LOAD_FAST
LOAD_FAST
BINARY_OP_TRUE_DIVIDE_INT_FLOAT
CALL_BUILTIN_FAST_WITH_KEYWORDS
POP_TOP
JUMP_BACKWARD
```

Region Selection

What? When? Where?

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
FOR_ITER_LIST
STORE_FAST
LOAD_FAST
LOAD_GLOBAL_MODULE
BINARY_OP_SUBTRACT_FLOAT
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL_PY_EXACT_ARGS
BINARY_OP_ADD_INT
STORE_FAST
LOAD_GLOBAL_BUILTIN
LOAD_FAST
LOAD_FAST
BINARY_OP_TRUE_DIVIDE_INT_FLOAT
CALL_BUILTIN_FAST_WITH_KEYWORDS
POP_TOP
JUMP_BACKWARD
```

Region Selection

What? When? Where? Why?

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
FOR_ITER_LIST
STORE_FAST
LOAD_FAST
LOAD_GLOBAL_MODULE
BINARY_OP_SUBTRACT_FLOAT
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL_PY_EXACT_ARGS
BINARY_OP_ADD_INT
STORE_FAST
LOAD_GLOBAL_BUILTIN
LOAD_FAST
LOAD_FAST
BINARY_OP_TRUE_DIVIDE_INT_FLOAT
CALL_BUILTIN_FAST_WITH_KEYWORDS
POP_TOP
JUMP_BACKWARD
```

Region Selection

Entire Methods

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

Region Selection

Entire Methods

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

Region Selection

Entire Methods

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
count = 0
for card in cards:
    decks -= ONE_CARD
    count += system(card)
    print(count / decks)
```

Region Selection

Entire Methods

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
count = 0
for card in cards:
    decks -= ONE_CARD

    count += system(card)
    print(count / decks)
```

Region Selection

Entire Methods

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
count = 0
for card in cards:
    decks -= ONE_CARD
    if system is system_hi_lo:
        if card in HI:
            count -= 1
        elif card in LO:
            count += 1
    else:
        count += system(card)
    print(count / decks)
```

Region Selection

Entire Methods

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
count = 0
for card in cards:
    decks -= ONE_CARD
    if system is system_hi_lo:
        if card in HI:
            count -= 1
        elif card in LO:
            count += 1

    else:
        count += system(card)
    print(count / decks)
```

Region Selection

Entire Methods

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
count = 0
for card in cards:
    decks -= ONE_CARD
    if system is system_hi_lo:
        if card in HI:
            count -= 1
        elif card in LO:
            count += 1
    elif system is system_a_5:
        if card == "A":
            count -= 1
        elif card == "5":
            count += 1
    else:
        count += system(card)
    print(count / decks)
```

Region Selection

Entire Methods

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
count = 0
for card in cards:
    decks -= ONE_CARD
    if system is system_hi_lo:
        if card in HI:
            count -= 1
        elif card in LO:
            count += 1
    elif system is system_a_5:
        if card == "A":
            count -= 1
        elif card == "5":
            count += 1
    else:
        count += system(card)
    print(count / decks)
```

Region Selection

Entire Methods

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
count = 0
for card in cards:
    decks -= ONE_CARD
    if system is system_hi_lo:
        if card in HI:
            count -= 1
        elif card in LO:
            count += 1
    elif system is system_a_5:
        if card == "A":
            count -= 1
        elif card == "5":
            count += 1
    else:
        count += system(card)
    print(count / decks)
```

Region Selection

Entire Methods

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
def get_cards(video):
    for card in detect(video):
        yield card
```

```
count = 0
for card in cards:
    decks -= ONE_CARD
    if system is system_hi_lo:
        if card in HI:
            count -= 1
        elif card in LO:
            count += 1
    elif system is system_a_5:
        if card == "A":
            count -= 1
        elif card == "5":
            count += 1
    else:
        count += system(card)
    print(count / decks)
```

Region Selection

Entire Methods

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
def get_cards(video):
    for card in detect(video):
        yield card
```

```
count = 0
for card in cards:
    decks -= ONE_CARD
    if system is system_hi_lo:
        if card in HI:
            count -= 1
        elif card in LO:
            count += 1
    elif system is system_a_5:
        if card == "A":
            count -= 1
        elif card == "5":
            count += 1
    else:
        count += system(card)
    print(count / decks)
```

Region Selection

Entire Methods

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
def get_cards(video):
    for image in video:
        for card in detect(image):
            yield card
```

```
count = 0
for card in cards:
    decks -= ONE_CARD
    if system is system_hi_lo:
        if card in HI:
            count -= 1
        elif card in LO:
            count += 1
    elif system is system_a_5:
        if card == "A":
            count -= 1
        elif card == "5":
            count += 1
    else:
        count += system(card)
    print(count / decks)
```

Region Selection

Recorded Traces

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

Region Selection

Recorded Traces

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

Region Selection

Recorded Traces

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
for card in cards:
    decks -= ONE_CARD
    if system is not system_hi_lo: ...
    if card not in HI: ...
    count -= 1
    print(count / decks)
```

Region Selection

Recorded Traces

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
for card in cards:
    decks -= ONE_CARD
    if system is not system_hi_lo: ...
    if card not in HI: ...
    count -= 1
    print(count / decks)
```

Region Selection

Recorded Traces

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
for card in cards:
    decks -= ONE_CARD
    if system is not system_hi_lo: ...
    if card not in HI: !!!
    count -= 1
    print(count / decks)
```

Region Selection

Recorded Traces

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
for card in cards:
    decks -= ONE_CARD
    if system is not system_hi_lo: ...
    if card not in HI:
        if card not in LO: ...
        count += 1
        print(count / decks)
        continue
    count -= 1
    print(count / decks)
```

Region Selection

Recorded Traces

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
for card in cards:
    decks -= ONE_CARD
    if system is not system_hi_lo: ...
    if card not in HI:
        if card not in LO: ...
        count += 1
        print(count / decks)
        continue
    count -= 1
    print(count / decks)
```

Region Selection

Recorded Traces

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
for card in cards:
    decks -= ONE_CARD
    if system is not system_hi_lo: ...
    if card not in HI:
        if card not in LO: !!!
        count += 1
        print(count / decks)
        continue
    count -= 1
    print(count / decks)
```

Region Selection

Recorded Traces

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
for card in cards:
    decks -= ONE_CARD
    if system is not system_hi_lo: ...
    if card not in HI:
        if card not in LO:
            print(count / decks)
            continue
        count += 1
        print(count / decks)
        continue
    count -= 1
    print(count / decks)
```

Region Selection

Recorded Traces

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
for card in cards:
    decks -= ONE_CARD
    if system is not system_hi_lo: ...
    if card not in HI:
        if card not in LO:
            print(count / decks)
            continue
        count += 1
        print(count / decks)
        continue
    count -= 1
    print(count / decks)
```

Region Selection

Recorded Traces

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
for card in cards:
    decks -= ONE_CARD
    if system is not system_hi_lo: !!!
    if card not in HI:
        if card not in LO:
            print(count / decks)
            continue
        count += 1
        print(count / decks)
        continue
    count -= 1
    print(count / decks)
```

Region Selection

Recorded Traces

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
for card in cards:
    decks -= ONE_CARD
    if system is not system_hi_lo:



    if card not in HI:
        if card not in LO:
            print(count / decks)
            continue
        count += 1
        print(count / decks)
        continue
    count -= 1
    print(count / decks)
```

Region Selection

Recorded Traces

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
for card in cards:
    decks -= ONE_CARD
    if system is not system_hi_lo:
        if system is not system_a_5: ...
        if card == "A": ...
        if card == "5": ...
        print(count / decks)
        continue
    if card not in HI:
        if card not in LO:
            print(count / decks)
            continue
        count += 1
        print(count / decks)
        continue
    count -= 1
    print(count / decks)
```

Region Selection

Recorded Traces

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
for card in cards:
    decks -= ONE_CARD
    if system is not system_hi_lo:
        if system is not system_a_5: ...
        if card == "A": ...
        if card == "5": ...
        print(count / decks)
        continue
    if card not in HI:
        if card not in LO:
            print(count / decks)
            continue
        count += 1
        print(count / decks)
        continue
    count -= 1
    print(count / decks)
```

Region Selection

Recorded Traces

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
for card in cards:
    decks -= ONE_CARD
    if system is not system_hi_lo:
        if system is not system_a_5: ...
        if card == "A": ...
        if card == "5": ...
        print(count / decks)
        continue
    if card not in HI:
        if card not in LO:
            print(count / decks)
            continue
        count += 1
        print(count / decks)
        continue
    count -= 1
    print(count / decks)
```

Region Selection

Projected Traces

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

Region Selection

Projected Traces

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
for card in cards:
    decks -= ONE_CARD
    if system is not system_a_5: ...
    if card == "A": ...
    if card == "5": ...
    print(count / decks)
```

Region Selection

Projected Traces

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
for card in cards:
    decks -= ONE_CARD
    if system is not system_a_5: ...
    if card == "A": ...
    if card == "5": ...
    print(count / decks)
```

Region Selection

Projected Traces

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
for card in cards:
    decks -= ONE_CARD
    if system is not system_a_5: !!!
    if card == "A": ...
    if card == "5": ...
    print(count / decks)
```

Region Selection

Projected Traces

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
for card in cards:
    decks -= ONE_CARD
    if system is not system_a_5:
        if system is not system_hi_lo:
            if card in HI: ...
            if card not in LO: ...
            count += 1
            print(count / decks)
            continue
    if card == "A": ...
    if card == "5": ...
    print(count / decks)
```

Region Selection

Projected Traces

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
for card in cards:
    decks -= ONE_CARD
    if system is not system_a_5:
        if system is not system_hi_lo:
            if card in HI: ...
            if card not in LO: ...
            count += 1
            print(count / decks)
            continue
    if card == "A": ...
    if card == "5": ...
    print(count / decks)
```

Region Selection

Projected Traces

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
for card in cards:
    decks -= ONE_CARD
    if system is not system_a_5:
        if system is not system_hi_lo:
            if card in HI: !!!
            if card not in LO: ...
            count += 1
            print(count / decks)
            continue
    if card == "A": ...
    if card == "5": ...
    print(count / decks)
```

Region Selection

Projected Traces

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
for card in cards:
    decks -= ONE_CARD
    if system is not system_a_5:
        if system is not system_hi_lo:
            if card in HI:
                count -= 1
                print(count / decks)
                continue
            if card not in LO: ...
            count += 1
            print(count / decks)
            continue
    if card == "A": ...
    if card == "5": ...
    print(count / decks)
```

Region Selection

Projected Traces

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
for card in cards:
    decks -= ONE_CARD
    if system is not system_a_5:
        if system is not system_hi_lo:
            if card in HI:
                count -= 1
                print(count / decks)
                continue
            if card not in LO: ...
            count += 1
            print(count / decks)
            continue
    if card == "A": ...
    if card == "5": ...
    print(count / decks)
```

Region Selection

Long Traces

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

Region Selection

Short Traces

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

Region Selection

Short Traces

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
for card in cards:
    ...
```

Region Selection

Short Traces

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
for card in cards:
    decks -= ONE_CARD
    if system is system_hi_lo: ...
    ...
```

Region Selection

Short Traces

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
for card in cards:
    decks -= ONE_CARD
    if system is system_hi_lo: ...
    if system is not system_a_5: ...
    if card == "A": ...
    ...
```

Region Selection

Short Traces

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
for card in cards:
    decks -= ONE_CARD
    if system is system_hi_lo: ...
    if system is not system_a_5: ...
    if card == "A": ...
    if card == "5": ...
    ...
```

Region Selection

Short Traces

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
for card in cards:
    decks -= ONE_CARD
    if system is system_hi_lo: ...
    if system is not system_a_5: ...
    if card == "A": ...
    if card == "5": ...
    print(count / decks)
```

Region Selection

Short Traces

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
for card in cards:
    decks -= ONE_CARD
    if system is system_hi_lo: ...
    if system is not system_a_5: ...
    if card == "A": ...
    if card == "5": ...
    print(count / decks)
```

Region Selection

Short Traces

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
for card in cards:
    decks -= ONE_CARD
    if system is system_hi_lo: ...
    if system is not system_a_5: ...
    if card == "A": !!!
    if card == "5": ...
    print(count / decks)
```

Region Selection

Short Traces

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
for card in cards:
    decks -= ONE_CARD
    if system is system_hi_lo: ...
    if system is not system_a_5: ...
    if card == "A":
        count -= 1
        print(count / decks)
        continue
    if card == "5": ...
    print(count / decks)
```

Region Selection

Short Traces

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
for card in cards:
    decks -= ONE_CARD
    if system is system_hi_lo: ...
    if system is not system_a_5: ...
    if card == "A":
        count -= 1
        print(count / decks)
        continue
    if card == "5": !!!
    print(count / decks)
```

Region Selection

Short Traces

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
for card in cards:
    decks -= ONE_CARD
    if system is system_hi_lo: ...
    if system is not system_a_5: ...
    if card == "A":
        count -= 1
        print(count / decks)
        continue
    if card == "5":
        count += 1
        print(count / decks)
        continue
    print(count / decks)
```

Region Selection

Short Traces

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

```
for card in cards:
    decks -= ONE_CARD
    if system is system_hi_lo: ...
    if system is not system_a_5: ...
    if card == "A":
        count -= 1
        print(count / decks)
        continue
    if card == "5":
        count += 1
        print(count / decks)
        continue
    print(count / decks)
```

Region Selection

Short Traces

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

Region Selection

Meta Traces

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

Region Selection

Meta-Traces

```
def count(cards, decks, system):
    count = 0
    for card in cards:
        decks -= ONE_CARD
        count += system(card)
        print(count / decks)

def system_hi_lo(card):
    if card in HI:
        return -1
    if card in LO:
        return 1
    return 0

def system_a_5(card):
    if card == "A":
        return -1
    if card == "5":
        return 1
    return 0
```

Region Selection

Meta-Traces

- Ask Chat about the "first Futurama projection".

Region Selection

Meta-Traces

- Ask Chat about the "first Futamura projection".

Region Selection

Meta-Traces

- `partial(f(x, y), A) → f(A, y) → f'(y)`
- `interpreter(program, input) → output`

Region Selection

Meta-Traces

1. `partial(interpreter, program)`
   - `thing(input) → output`
     - "Executable"

2. `partial(partial, interpreter)`
   - `thing(program) → executable`
     - "Compiler"

3. `partial(partial, partial)`
   - `thing(interpreter) → compiler`
     - "Universal compiler factory"

Region Selection

Meta-Traces

1. `partial(interpreter, program)`
   - `thing(input) → output`
     - "Executable"

2. `partial(partial, interpreter)`
   - `thing(program) → executable`
     - "Compiler"

3. `partial(partial, partial)`
   - `thing(interpreter) → compiler`
     - "Universal compiler factory (powered by Dark CS Magic™)"

Region Selection

Meta-Traces

1. `partial(interpreter, program)`
   - `thing(input) → output`
     - "Executable"

2. `partial(partial, interpreter)`
   - `thing(program) → executable`
     - "Compiler"

3. `partial(partial, partial)`
   - `thing(interpreter) → compiler`
     - "Universal compiler factory (powered by Dark CS Magic™)"

Region Selection

Region Selection

Memory Management

Memory Management

Memory Management

Readable Data

Memory Management

Readable Data

```
const char data[4] = {0xc0, 0x03, 0x5f, 0xd6};
```

Memory Management

Readable / Writable Data

```
const char data[4] = {0xc0, 0x03, 0x5f, 0xd6};
```

Memory Management

Readable / Writable Data

```
char data[4];
```

Memory Management

Readable / Writable Data

```
char data[4];


data[0] = 0xc0;
data[1] = 0x03;
data[2] = 0x5f;
data[3] = 0xd6;
```

Memory Management

Readable / Writable Data

```
char *data = malloc(4);


data[0] = 0xc0;
data[1] = 0x03;
data[2] = 0x5f;
data[3] = 0xd6;




free(data);
```

Memory Management

Readable / Writable / Executable Data

```
char *data = malloc(4);


data[0] = 0xc0;
data[1] = 0x03;
data[2] = 0x5f;
data[3] = 0xd6;




free(data);
```

Memory Management

Readable / Writable / Executable Data

```
char *data = malloc(4);


data[0] = 0xc0;
data[1] = 0x03;
data[2] = 0x5f;
data[3] = 0xd6;


typedef int (*function)(int);


free(data);
```

Memory Management

Readable / Writable / Executable Data

```
char *data = malloc(4);


data[0] = 0xc0;
data[1] = 0x03;
data[2] = 0x5f;
data[3] = 0xd6;


typedef int (*function)(int);
data;

free(data);
```

Memory Management

Readable / Writable / Executable Data

```
char *data = malloc(4);


data[0] = 0xc0;
data[1] = 0x03;
data[2] = 0x5f;
data[3] = 0xd6;


typedef int (*function)(int);
((function)data);

free(data);
```

Memory Management

Readable / Writable / Executable Data

```
char *data = malloc(4);


data[0] = 0xc0;
data[1] = 0x03;
data[2] = 0x5f;
data[3] = 0xd6;


typedef int (*function)(int);
((function)data)(42);

free(data);
```

Memory Management

Readable / Writable / Executable Data

```
char *data = malloc(4);


data[0] = 0xc0;
data[1] = 0x03;
data[2] = 0x5f;
data[3] = 0xd6;


typedef int (*function)(int);
((function)data)(42);

free(data);
```

Memory Management

Readable / Writable / Executable Data

```
char *data = mmap(NULL, 4096,
                  PROT_READ | PROT_WRITE | PROT_EXEC,
                  MAP_ANONYMOUS | MAP_PRIVATE, -1, 0);
data[0] = 0xc0;
data[1] = 0x03;
data[2] = 0x5f;
data[3] = 0xd6;


typedef int (*function)(int);
((function)data)(42);

munmap(data, 4096);
```

Memory Management

Readable / Writable / Executable Data

```
char *data = mmap(NULL, 4096,
                  PROT_READ | PROT_WRITE | PROT_EXEC,
                  MAP_ANONYMOUS | MAP_PRIVATE, -1, 0);
data[0] = 0xc0;
data[1] = 0x03;
data[2] = 0x5f;
data[3] = 0xd6;


typedef int (*function)(int);
((function)data)(42);

munmap(data, 4096);
```

Memory Management

Readable / Writable / Executable Data

```
char *data = mmap(NULL, 4096,
                  PROT_READ | PROT_WRITE | PROT_EXEC,
                  MAP_ANONYMOUS | MAP_PRIVATE, -1, 0);
data[0] = 0xc0;
data[1] = 0x03;
data[2] = 0x5f;
data[3] = 0xd6;


typedef int (*function)(int);
((function)data)(42);

munmap(data, 4096);
```

Memory Management

Readable / Writable / Executable Data

```
char *data = mmap(NULL, 4096,
                  PROT_READ | PROT_WRITE | PROT_EXEC,
                  MAP_ANONYMOUS | MAP_PRIVATE, -1, 0);
data[0] = 0xc0;
data[1] = 0x03;
data[2] = 0x5f;
data[3] = 0xd6;


typedef int (*function)(int);
((function)data)(42);

munmap(data, 4096);
```

Memory Management

Readable / Writable / Executable Data

```
char *data = mmap(NULL, 4096,
                  PROT_READ | PROT_WRITE | PROT_EXEC,
                  MAP_ANONYMOUS | MAP_PRIVATE, -1, 0);
data[0] = 0xc0;
data[1] = 0x03;
data[2] = 0x5f;
data[3] = 0xd6;


typedef int (*function)(int);
((function)data)(42);

munmap(data, 4096);
```

Memory Management

Readable / Writable / Executable Data

```
char *data = mmap(NULL, 4096,
                  PROT_READ | PROT_WRITE | PROT_EXEC,
                  MAP_ANONYMOUS | MAP_PRIVATE, -1, 0);
data[0] = 0xc0;
data[1] = 0x03;
data[2] = 0x5f;
data[3] = 0xd6;


typedef int (*function)(int);
((function)data)(42);

munmap(data, 4096);
```

Memory Management

Readable / Writable / Executable Data

```
char *data = mmap(NULL, 4096,
                  PROT_READ | PROT_WRITE | PROT_EXEC,
                  MAP_ANONYMOUS | MAP_PRIVATE, -1, 0);
data[0] = 0xc0;
data[1] = 0x03;
data[2] = 0x5f;
data[3] = 0xd6;


typedef int (*function)(int);
((function)data)(42);

munmap(data, 4096);
```

Memory Management

Readable / Writable / Executable Data

```
char *data = mmap(NULL, 4096,
                  PROT_READ | PROT_WRITE,
                  MAP_ANONYMOUS | MAP_PRIVATE, -1, 0);
data[0] = 0xc0;
data[1] = 0x03;
data[2] = 0x5f;
data[3] = 0xd6;
mprotect(data, 4096, PROT_READ | PROT_EXEC);

typedef int (*function)(int);
((function)data)(42);

munmap(data, 4096);
```

Memory Management

Readable / Writable / Executable Data

```
char *data = VirtualAlloc(NULL, 4096, MEM_COMMIT | MEM_RESERVE,
                          PAGE_READWRITE)
int old;
data[0] = 0xc0;
data[1] = 0x03;
data[2] = 0x5f;
data[3] = 0xd6;
VirtualProtect(data, 4096, PAGE_EXECUTE_READ, &old);

typedef int (*function)(int);
((function)data)(42);

VirtualFree(data, 0, MEM_RELEASE);
```

Memory Management

Memory Management

Debugging and Profiling

Debugging and Profiling

Debugging and Profiling

Python Profilers

Debugging and Profiling
Python Profilers

```
FOR_ITER_LIST
STORE_FAST
LOAD_FAST
LOAD_GLOBAL_MODULE
BINARY_OP_SUBTRACT_FLOAT
STORE_FAST
LOAD_FAST
LOAD_FAST
LOAD_FAST
CALL_PY_EXACT_ARGS
BINARY_OP_ADD_INT
STORE_FAST
LOAD_GLOBAL_BUILTIN
LOAD_FAST
LOAD_FAST
BINARY_OP_TRUE_DIVIDE_INT_FLOAT
CALL_BUILTIN_FAST_WITH_KEYWORDS
POP_TOP
JUMP_BACKWARD
```

Debugging and Profiling

Python Profilers

```
INSTRUMENTED_FOR_ITER
STORE_FAST
INSTRUMENTED_LINE
LOAD_GLOBAL_MODULE
BINARY_OP_SUBTRACT_FLOAT
STORE_FAST
INSTRUMENTED_LINE
LOAD_FAST
LOAD_FAST
INSTRUMENTED_CALL
BINARY_OP_ADD_INT
STORE_FAST
INSTRUMENTED_LINE
LOAD_FAST
LOAD_FAST
BINARY_OP_TRUE_DIVIDE_INT_FLOAT
INSTRUMENTED_CALL
POP_TOP
INSTRUMENTED_JUMP_BACKWARD
```

Debugging and Profiling

Python Profilers

```
INSTRUMENTED_FOR_ITER
STORE_FAST
INSTRUMENTED_LINE
LOAD_GLOBAL_MODULE
BINARY_OP_SUBTRACT_FLOAT
STORE_FAST
INSTRUMENTED_LINE
LOAD_FAST
LOAD_FAST
INSTRUMENTED_CALL
BINARY_OP_ADD_INT
STORE_FAST
INSTRUMENTED_LINE
LOAD_FAST
LOAD_FAST
BINARY_OP_TRUE_DIVIDE_INT_FLOAT
INSTRUMENTED_CALL
POP_TOP
INSTRUMENTED_JUMP_BACKWARD
```

```
_CHECK_VALIDITY
_ITER_CHECK_LIST
_GUARD_NOT_EXHAUSTED_LIST
_ITER_NEXT_LIST
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
_CHECK_FUNCTION
_LOAD_CONST_INLINE
_GUARD_NOS_FLOAT
_BINARY_OP_SUBTRACT_FLOAT
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
_LOAD_FAST
_LOAD_FAST
_SET_IP
_CHECK_FUNCTION_VERSION
_CHECK_FUNCTION_EXACT_ARGS
_CHECK_STACK_SPACE
```

```
_INIT_CALL_PY_EXACT_ARGS
_SAVE_RETURN_OFFSET
_PUSH_FRAME
_CHECK_VALIDITY_AND_SET_IP
_GUARD_NOS_INT
_BINARY_OP_ADD_INT
_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_CONST_INLINE
_LOAD_FAST
_LOAD_FAST
_SET_IP
_BINARY_OP_TRUE_DIVIDE_INT_FLOAT
_CHECK_VALIDITY_AND_SET_IP
_CALL_BUILTIN_FAST_WITH_KEYWORDS
_CHECK_PERIODIC
_CHECK_VALIDITY
_POP_TOP
_CHECK_VALIDITY_AND_SET_IP
_CHECK_PERIODIC
_JUMP_TO_TOP
```

Debugging and Profiling

Python Profilers

```
INSTRUMENTED_FOR_ITER
STORE_FAST
INSTRUMENTED_LINE
LOAD_GLOBAL_MODULE
BINARY_OP_SUBTRACT_FLOAT
STORE_FAST
INSTRUMENTED_LINE
LOAD_FAST
LOAD_FAST
INSTRUMENTED_CALL
BINARY_OP_ADD_INT
STORE_FAST
INSTRUMENTED_LINE
LOAD_FAST
LOAD_FAST
BINARY_OP_TRUE_DIVIDE_INT_FLOAT
INSTRUMENTED_CALL
POP_TOP
INSTRUMENTED_JUMP_BACKWARD
```

```
_CHECK_VALIDITY
_ITER_CHECK_LIST
_GUARD_NOT_EXHAUSTED_LIST
_ITER_NEXT_LIST
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
_CHECK_FUNCTION
_LOAD_CONST_INLINE
_GUARD_NOS_FLOAT
_BINARY_OP_SUBTRACT_FLOAT
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
_LOAD_FAST
_LOAD_FAST
_SET_IP
_CHECK_FUNCTION_VERSION
_CHECK_FUNCTION_EXACT_ARGS
_CHECK_STACK_SPACE
```

```
_INIT_CALL_PY_EXACT_ARGS
_SAVE_RETURN_OFFSET
_PUSH_FRAME
_CHECK_VALIDITY_AND_SET_IP
_GUARD_NOS_INT
_BINARY_OP_ADD_INT
_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_CONST_INLINE
_LOAD_FAST
_LOAD_FAST
_SET_IP
_BINARY_OP_TRUE_DIVIDE_INT_FLOAT
_CHECK_VALIDITY_AND_SET_IP
_CALL_BUILTIN_FAST_WITH_KEYWORDS
_CHECK_PERIODIC
_CHECK_VALIDITY
_POP_TOP
_CHECK_VALIDITY_AND_SET_IP
_CHECK_PERIODIC
_JUMP_TO_TOP
```

Debugging and Profiling

Python Debuggers

```
_CHECK_VALIDITY
_ITER_CHECK_LIST
_GUARD_NOT_EXHAUSTED_LIST
_ITER_NEXT_LIST
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
_CHECK_FUNCTION
_LOAD_CONST_INLINE
_GUARD_NOS_FLOAT
_BINARY_OP_SUBTRACT_FLOAT
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
_LOAD_FAST
_LOAD_FAST
_SET_IP
_CHECK_FUNCTION_VERSION
_CHECK_FUNCTION_EXACT_ARGS
_CHECK_STACK_SPACE
```

```
_INIT_CALL_PY_EXACT_ARGS
_SAVE_RETURN_OFFSET
_PUSH_FRAME
_CHECK_VALIDITY_AND_SET_IP
_GUARD_NOS_INT
_BINARY_OP_ADD_INT
_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_CONST_INLINE
_LOAD_FAST
_LOAD_FAST
_SET_IP
_BINARY_OP_TRUE_DIVIDE_INT_FLOAT
_CHECK_VALIDITY_AND_SET_IP
_CALL_BUILTIN_FAST_WITH_KEYWORDS
_CHECK_PERIODIC
_CHECK_VALIDITY
_POP_TOP
_CHECK_VALIDITY_AND_SET_IP
_CHECK_PERIODIC
_JUMP_TO_TOP
```

Debugging and Profiling

Python Debuggers

```
(Pdb)
```

```
_CHECK_VALIDITY
_ITER_CHECK_LIST
_GUARD_NOT_EXHAUSTED_LIST
_ITER_NEXT_LIST
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
_CHECK_FUNCTION
_LOAD_CONST_INLINE
_GUARD_NOS_FLOAT
_BINARY_OP_SUBTRACT_FLOAT
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
_LOAD_FAST
_LOAD_FAST
_SET_IP
_CHECK_FUNCTION_VERSION
_CHECK_FUNCTION_EXACT_ARGS
_CHECK_STACK_SPACE
```

```
_INIT_CALL_PY_EXACT_ARGS
_SAVE_RETURN_OFFSET
_PUSH_FRAME
_CHECK_VALIDITY_AND_SET_IP
_GUARD_NOS_INT
_BINARY_OP_ADD_INT
_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_CONST_INLINE
_LOAD_FAST
_LOAD_FAST
_SET_IP
_BINARY_OP_TRUE_DIVIDE_INT_FLOAT
_CHECK_VALIDITY_AND_SET_IP
_CALL_BUILTIN_FAST_WITH_KEYWORDS
_CHECK_PERIODIC
_CHECK_VALIDITY
_POP_TOP
_CHECK_VALIDITY_AND_SET_IP
_CHECK_PERIODIC
_JUMP_TO_TOP
```

Debugging and Profiling

Python Debuggers

```
(Pdb) card = "A"
```

```
_CHECK_VALIDITY
_ITER_CHECK_LIST
_GUARD_NOT_EXHAUSTED_LIST
_ITER_NEXT_LIST
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
_CHECK_FUNCTION
_LOAD_CONST_INLINE
_GUARD_NOS_FLOAT
_BINARY_OP_SUBTRACT_FLOAT
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
_LOAD_FAST
_LOAD_FAST
_SET_IP
_CHECK_FUNCTION_VERSION
_CHECK_FUNCTION_EXACT_ARGS
_CHECK_STACK_SPACE
```

```
_INIT_CALL_PY_EXACT_ARGS
_SAVE_RETURN_OFFSET
_PUSH_FRAME
_CHECK_VALIDITY_AND_SET_IP
_GUARD_NOS_INT
_BINARY_OP_ADD_INT
_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_CONST_INLINE
_LOAD_FAST
_LOAD_FAST
_SET_IP
_BINARY_OP_TRUE_DIVIDE_INT_FLOAT
_CHECK_VALIDITY_AND_SET_IP
_CALL_BUILTIN_FAST_WITH_KEYWORDS
_CHECK_PERIODIC
_CHECK_VALIDITY
_POP_TOP
_CHECK_VALIDITY_AND_SET_IP
_CHECK_PERIODIC
_JUMP_TO_TOP
```

Debugging and Profiling

Python Debuggers

```
(Pdb) card = "A"
(Pdb) count += 0.5
```

```
_CHECK_VALIDITY
_ITER_CHECK_LIST
_GUARD_NOT_EXHAUSTED_LIST
_ITER_NEXT_LIST
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
_CHECK_FUNCTION
_LOAD_CONST_INLINE
_GUARD_NOS_FLOAT
_BINARY_OP_SUBTRACT_FLOAT
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
_LOAD_FAST
_LOAD_FAST
_SET_IP
_CHECK_FUNCTION_VERSION
_CHECK_FUNCTION_EXACT_ARGS
_CHECK_STACK_SPACE
```

```
_INIT_CALL_PY_EXACT_ARGS
_SAVE_RETURN_OFFSET
_PUSH_FRAME
_CHECK_VALIDITY_AND_SET_IP
_GUARD_NOS_INT
_BINARY_OP_ADD_INT
_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_CONST_INLINE
_LOAD_FAST
_LOAD_FAST
_SET_IP
_BINARY_OP_TRUE_DIVIDE_INT_FLOAT
_CHECK_VALIDITY_AND_SET_IP
_CALL_BUILTIN_FAST_WITH_KEYWORDS
_CHECK_PERIODIC
_CHECK_VALIDITY
_POP_TOP
_CHECK_VALIDITY_AND_SET_IP
_CHECK_PERIODIC
_JUMP_TO_TOP
```

Debugging and Profiling

Python Debuggers

```
(Pdb) card = "A"
(Pdb) count += 0.5
(Pdb) jump 9
```

```
_CHECK_VALIDITY
_ITER_CHECK_LIST
_GUARD_NOT_EXHAUSTED_LIST
_ITER_NEXT_LIST
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
_CHECK_FUNCTION
_LOAD_CONST_INLINE
_GUARD_NOS_FLOAT
_BINARY_OP_SUBTRACT_FLOAT
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
_LOAD_FAST
_LOAD_FAST
_SET_IP
_CHECK_FUNCTION_VERSION
_CHECK_FUNCTION_EXACT_ARGS
_CHECK_STACK_SPACE
```

```
_INIT_CALL_PY_EXACT_ARGS
_SAVE_RETURN_OFFSET
_PUSH_FRAME
_CHECK_VALIDITY_AND_SET_IP
_GUARD_NOS_INT
_BINARY_OP_ADD_INT
_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_CONST_INLINE
_LOAD_FAST
_LOAD_FAST
_SET_IP
_BINARY_OP_TRUE_DIVIDE_INT_FLOAT
_CHECK_VALIDITY_AND_SET_IP
_CALL_BUILTIN_FAST_WITH_KEYWORDS
_CHECK_PERIODIC
_CHECK_VALIDITY
_POP_TOP
_CHECK_VALIDITY_AND_SET_IP
_CHECK_PERIODIC
_JUMP_TO_TOP
```

Debugging and Profiling

Python Debuggers

```
(Pdb) card = "A"
(Pdb) count += 0.5
(Pdb) jump 9
(Pdb) continue
```

```
_CHECK_VALIDITY
_ITER_CHECK_LIST
_GUARD_NOT_EXHAUSTED_LIST
_ITER_NEXT_LIST
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
_CHECK_FUNCTION
_LOAD_CONST_INLINE
_GUARD_NOS_FLOAT
_BINARY_OP_SUBTRACT_FLOAT
_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_FAST
_LOAD_FAST
_LOAD_FAST
_SET_IP
_CHECK_FUNCTION_VERSION
_CHECK_FUNCTION_EXACT_ARGS
_CHECK_STACK_SPACE
```

```
_INIT_CALL_PY_EXACT_ARGS
_SAVE_RETURN_OFFSET
_PUSH_FRAME
_CHECK_VALIDITY_AND_SET_IP
_GUARD_NOS_INT
_BINARY_OP_ADD_INT
_CHECK_VALIDITY_AND_SET_IP
_STORE_FAST
_CHECK_VALIDITY
_LOAD_CONST_INLINE
_LOAD_FAST
_LOAD_FAST
_SET_IP
_BINARY_OP_TRUE_DIVIDE_INT_FLOAT
_CHECK_VALIDITY_AND_SET_IP
_CALL_BUILTIN_FAST_WITH_KEYWORDS
_CHECK_PERIODIC
_CHECK_VALIDITY
_POP_TOP
_CHECK_VALIDITY_AND_SET_IP
_CHECK_PERIODIC
_JUMP_TO_TOP
```

Debugging and Profiling

Native Profilers

Debugging and Profiling

Native Profilers

<interpreter>

Debugging and Profiling

Native Profilers

```
<foo>
<interpreter>
```

Debugging and Profiling

Native Profilers

```
<bar>
<foo>
<interpreter>
```

Debugging and Profiling

Native Profilers

```
<interpreter>
    <bar>
    <foo>
<interpreter>
```

Debugging and Profiling

Native Profilers

```
    <baz>
<interpreter>
    <bar>
    <foo>
<interpreter>
```

Debugging and Profiling

Native Profilers

```
    <baz>
<interpreter>
    <bar>
    <foo>
<interpreter>
```

Debugging and Profiling

Native Profilers

```
<interpreter>
    <baz>
<interpreter>
    <bar>
    <foo>
<interpreter>
```

Debugging and Profiling

Native Profilers

```
    <qux>
<interpreter>
    <baz>
<interpreter>
    <bar>
    <foo>
<interpreter>
```

Debugging and Profiling

Native Profilers

<qux>
<python: sausage>
<baz>
<python: bacon>
<python: eggs>
<bar>
<foo>
<python: Spam>

Debugging and Profiling

Native Profilers

```
    <qux>
<interpreter>
    <baz>
<interpreter>
    <bar>
    <foo>
<interpreter>
```

Debugging and Profiling

Native Profilers

```
    <qux>
<interpreter>
    <baz>
    <JIT>
    <bar>
    <foo>
<interpreter>
```

Debugging and Profiling

Native Profilers

<???>
<???>
<???>
<JIT>
<bar>
<foo>
<interpreter>

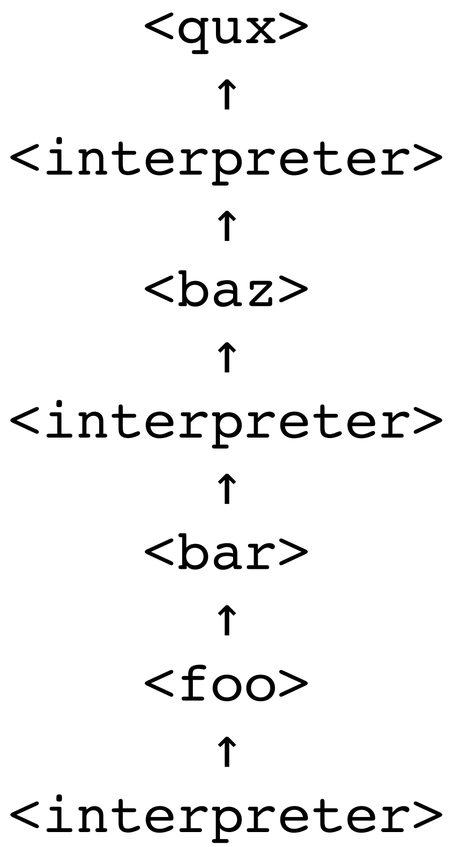
Debugging and Profiling

Native Profilers

```
    <qux>
<interpreter>
    <baz>
<interpreter>
    <bar>
    <foo>
<interpreter>
```

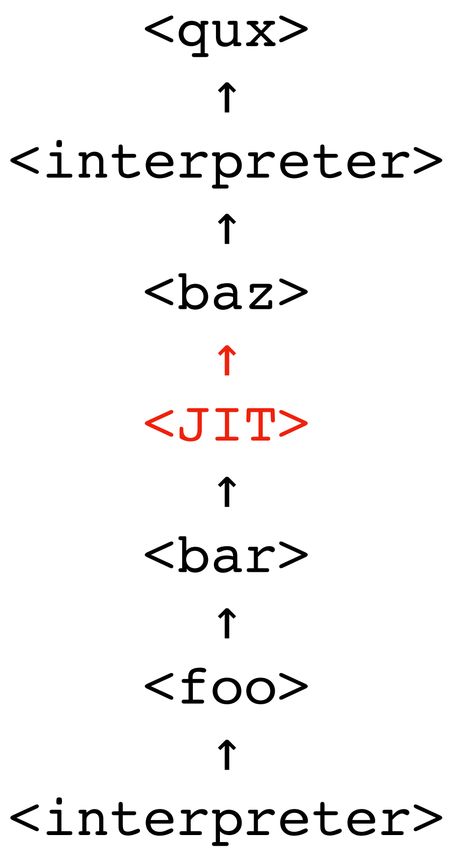
Debugging and Profiling

Native Profilers



Debugging and Profiling

Native Profilers



Debugging and Profiling

Native Profilers

```
    <qux>
<interpreter>
    <baz>
    <JIT>
    <bar>
    <foo>
<interpreter>
```

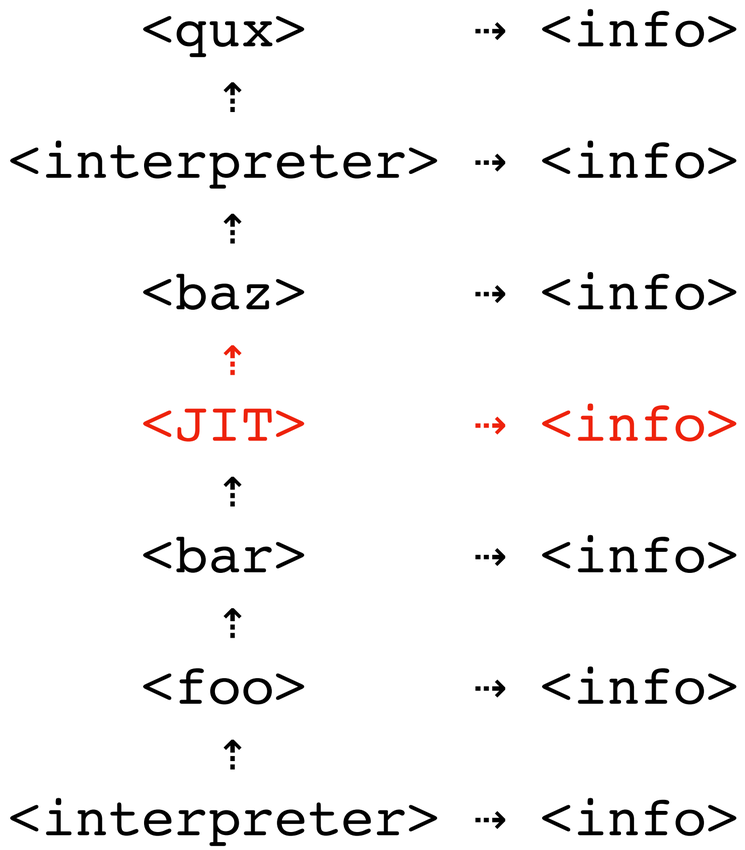
Debugging and Profiling

Native Profilers

```
    <qux>      ⇢ <info>
<interpreter>  ⇢ <info>
    <baz>      ⇢ <info>
    <JIT>      ⇢ <info>
    <bar>      ⇢ <info>
    <foo>      ⇢ <info>
<interpreter>  ⇢ <info>
```

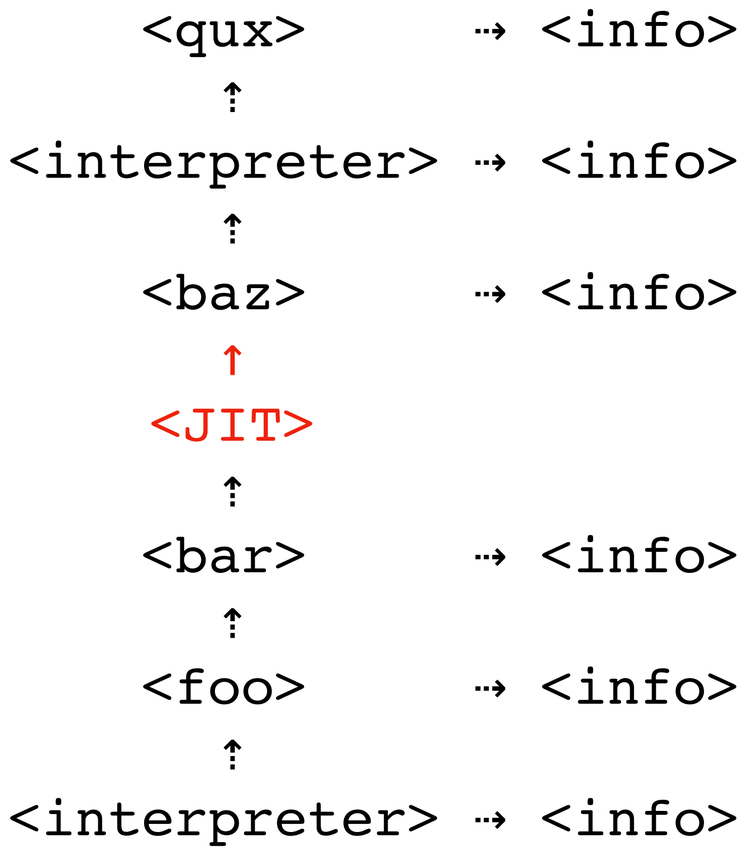
Debugging and Profiling

Native Profilers



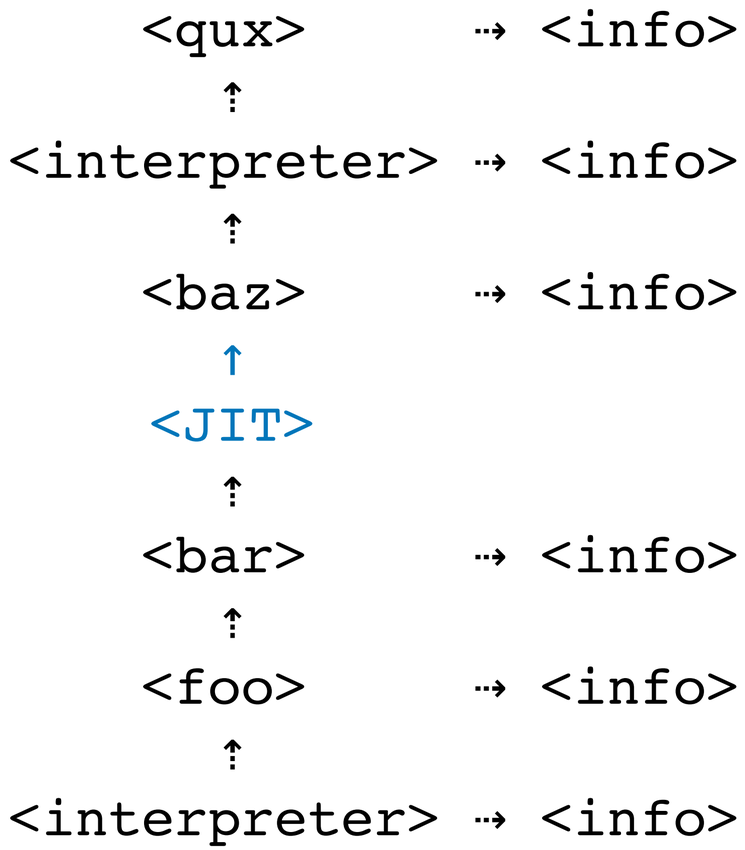
Debugging and Profiling

Native Profilers



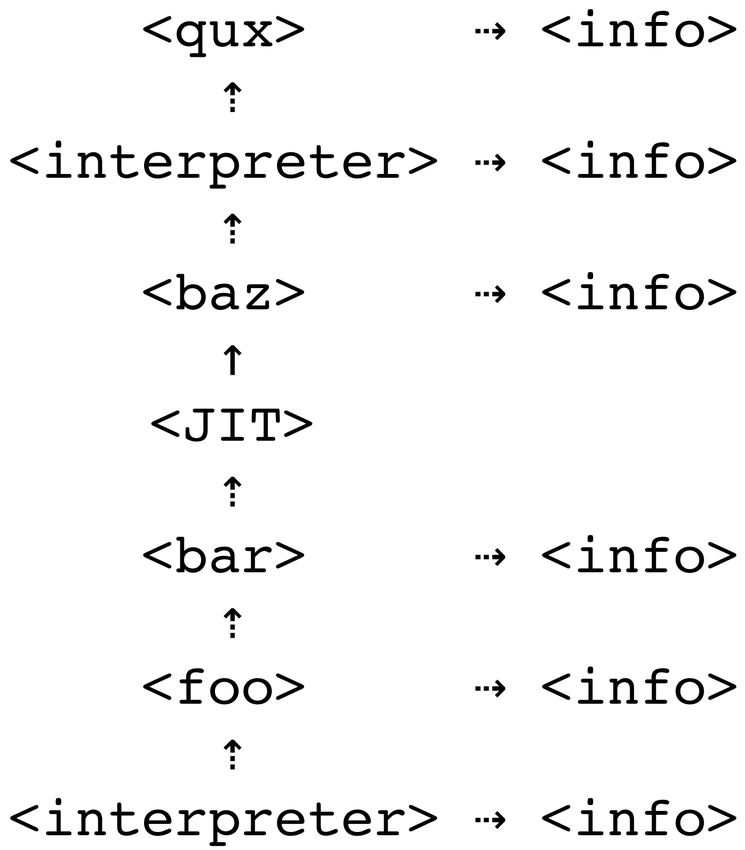
Debugging and Profiling

Native Profilers



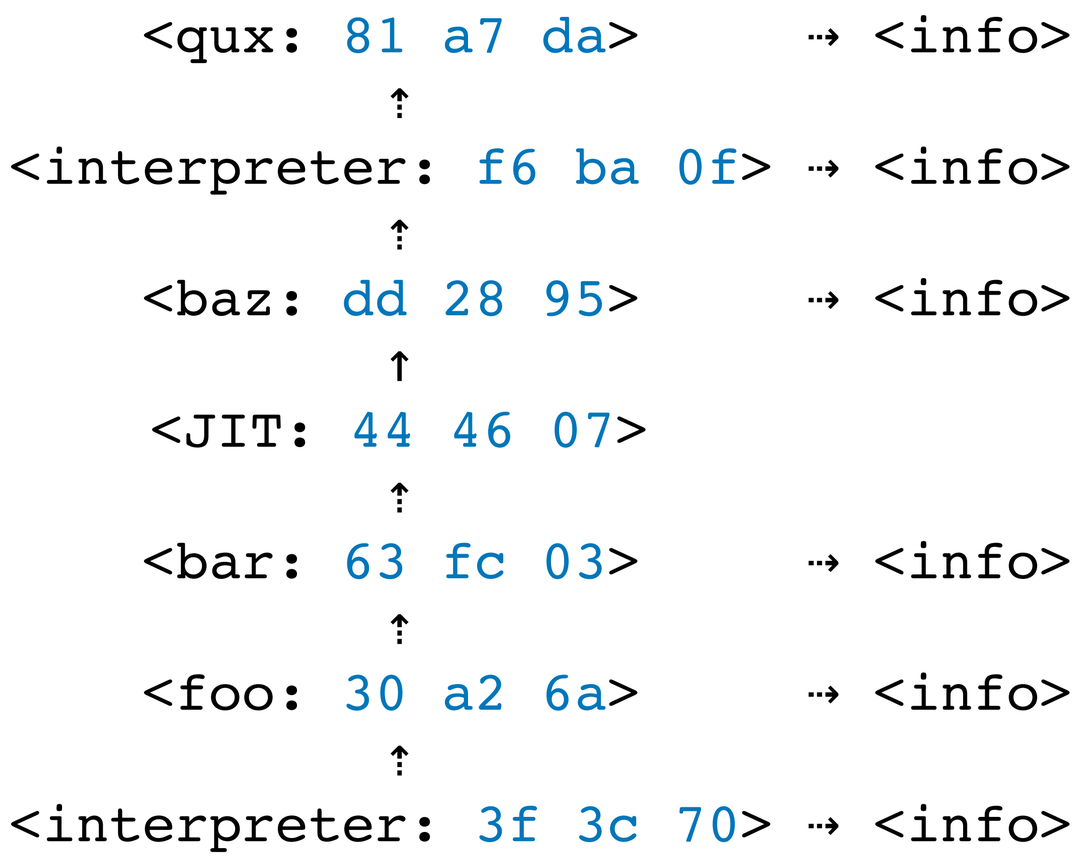
Debugging and Profiling

Native Debuggers



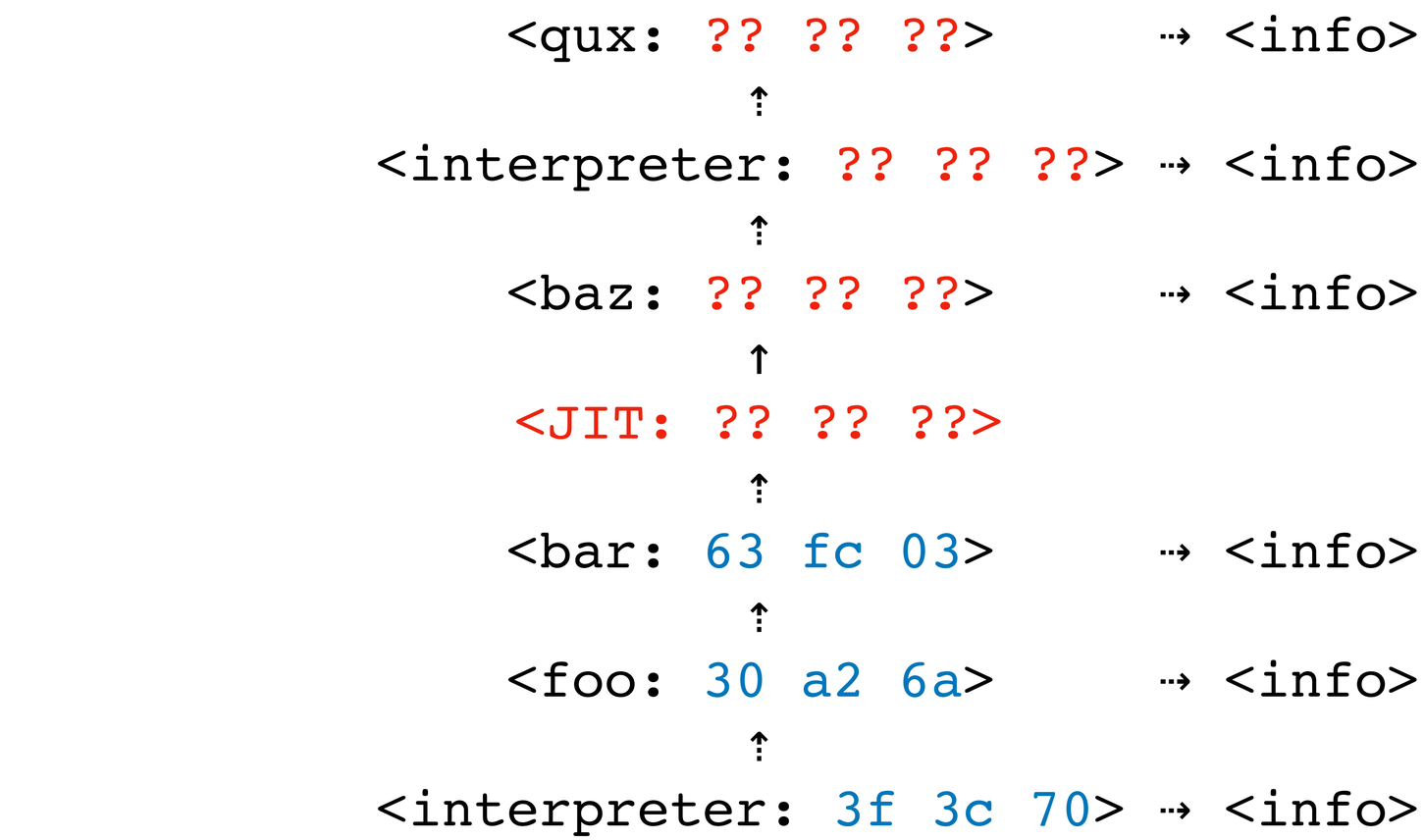
Debugging and Profiling

Native Debuggers



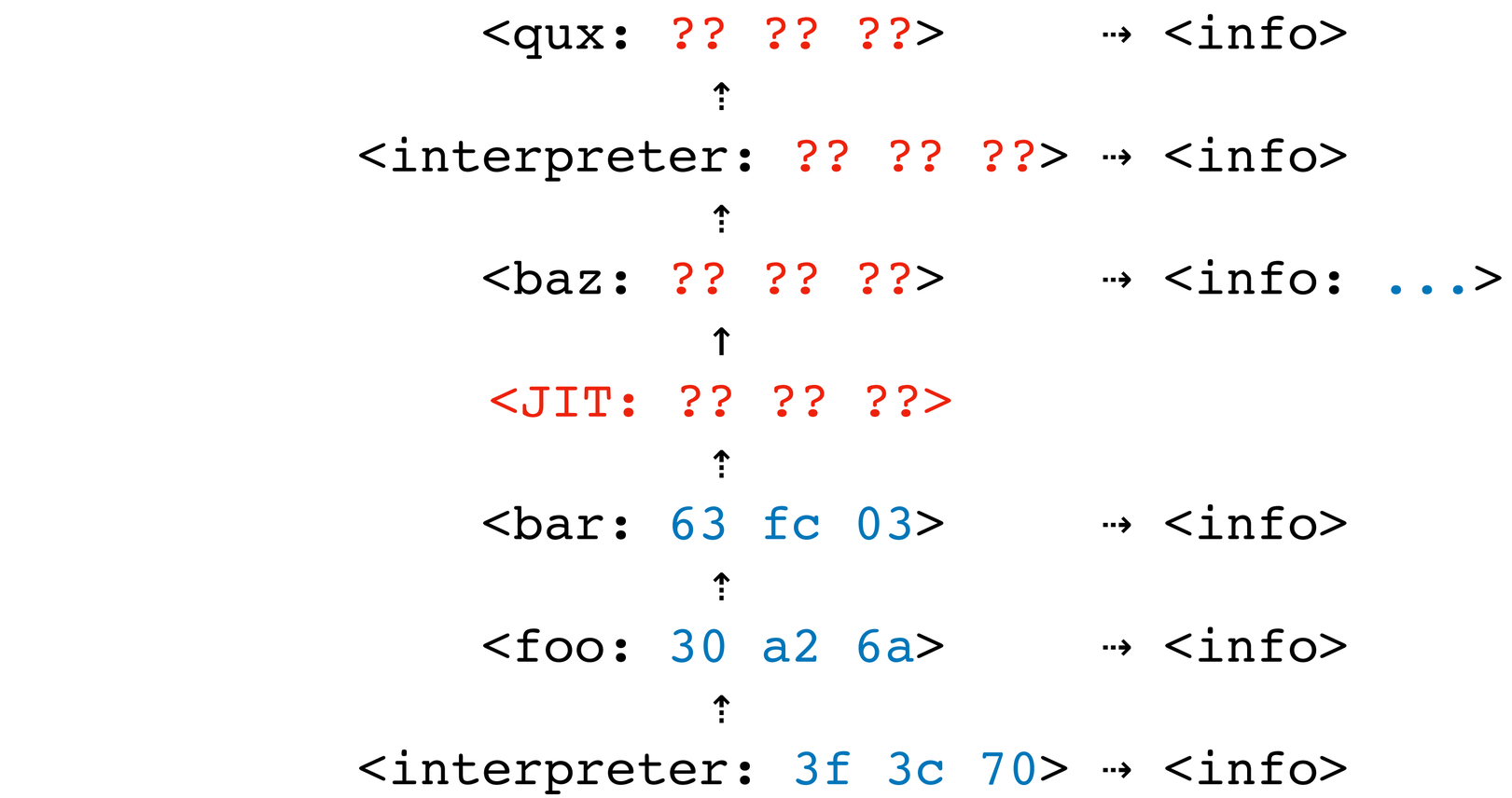
Debugging and Profiling

Native Debuggers



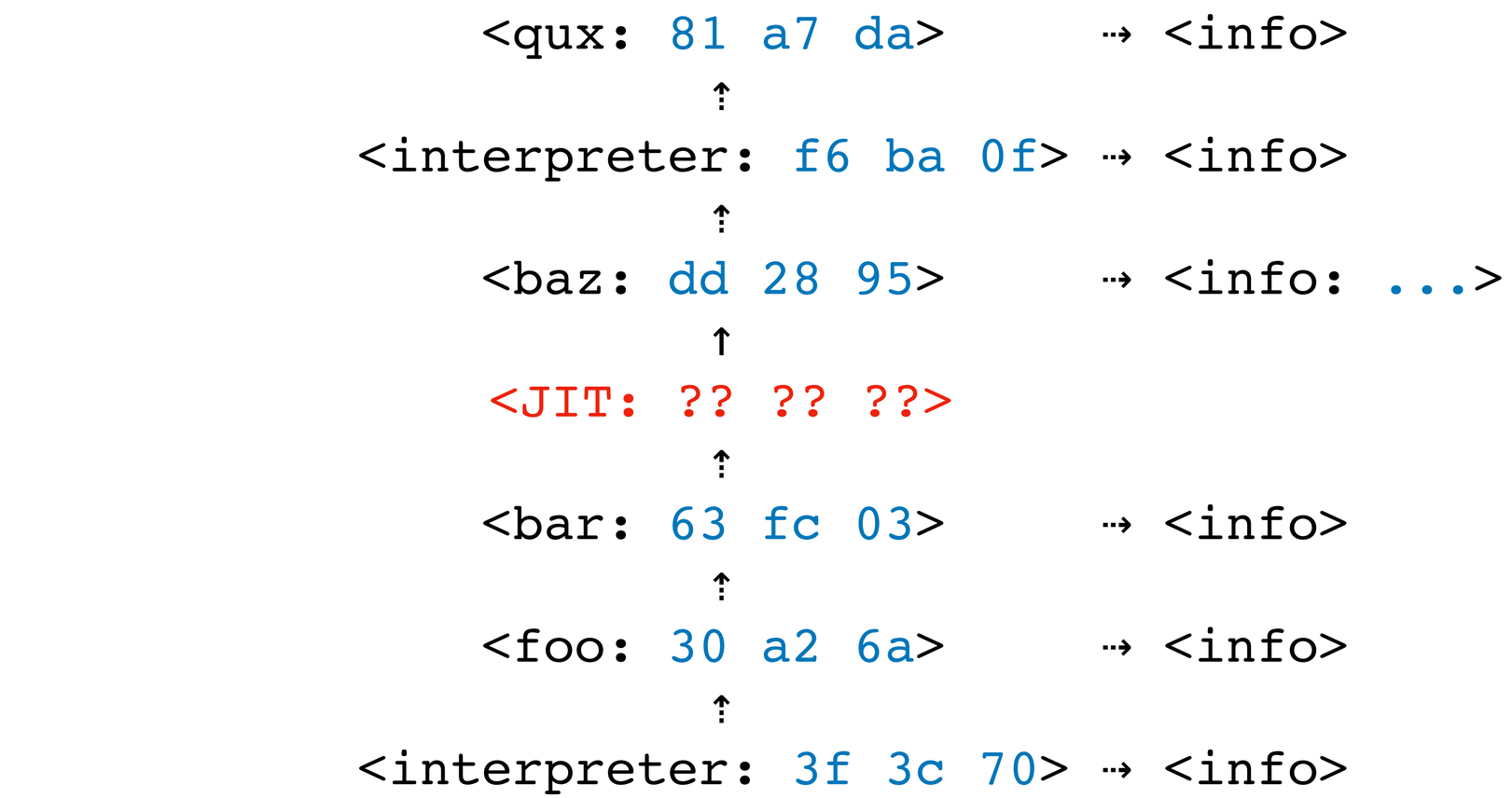
Debugging and Profiling

Native Debuggers



Debugging and Profiling

Native Debuggers



Thank you!

Thank you!

@brandtbucher

Thank you!

@brandtbucher | brandt@python.org

Thank you!

@brandtbucher | brandt@python.org | xkcd.com/451