Worksheet: OOP and Recursion

(1)

1. the parent class of Dog is Pet. Does Pet have a parent type? If so, what is it?

   object

2. how many arguments does line C pass?

   3

3. how many arguments does line B pass?

   2

4. on another paper, draw what the frames and object(s) will look like after line A. (check with PythonTutor)

```
class Pet:
    def __init__(self, name):
        self.name = name  # A

class Dog(Pet):
    def __init__(self, name, age):
        self.age = age
        Pet.__init__(self, name)  # B

pup = Dog("Sam", 1)  # C
```

(2)

```
def fact(n):
    if n == 0:
        return 1
    return n * fact(n-1)

# what is fact(5)
```

```
def fib(n):
    if n < 2:
        return n
    return fib(n-1) + fib(n-2)

# what is fib(6)?
```

recursive steps:

$fact(5) = 5 * fact(4)^{24} = 120$
$fact(4) = 4 * fact(3)^{6} = 24$
$fact(3) = 3 * fact(2)^{2} = 6$
$fact(2) = 2 * fact(1)^{1} = 2$
$fact(1) = 1 * fact(0)^{1} = 1$
$fact(0) = 1$ *base case*
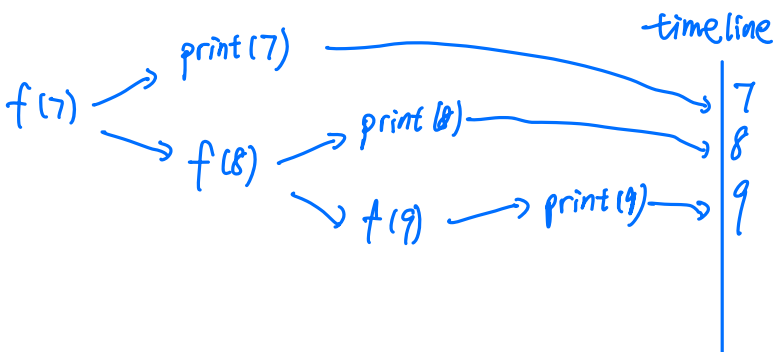
$fib(6) = fib(5) + fib(4) = 5+3 = 8$
$fib(5) = fib(4) + fib(3) = 3+2 = 5$
$fib(4) = fib(3) + fib(2) = 2+1 = 3$
$fib(3) = fib(2) + fib(1) = 1+1 = 2$
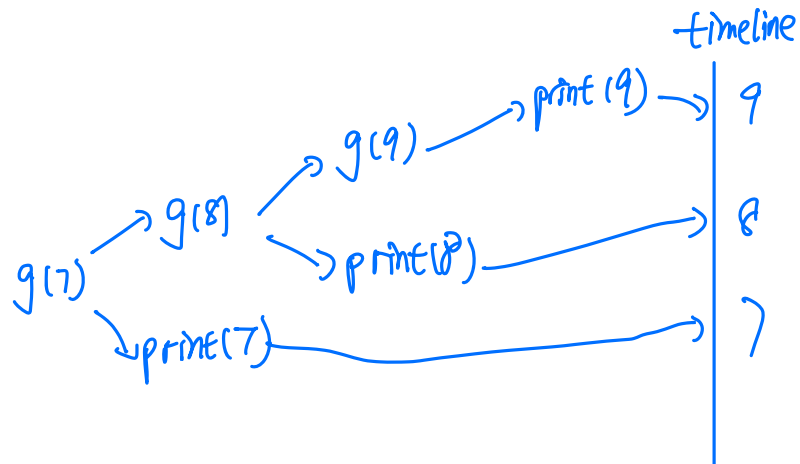$fib(2) = fib(1) + fib(0) = 1+0 = 1$
$fib(1) = 1$
$fib(0) = 0$

(3)

```
def f(n):
    print(n)
    if n < 9:
        f(n + 1)

# what does f(7) print?
```

```
def g(n):
    if n < 9:
        g(n + 1)
    print(n)

# what does g(7) print?
```
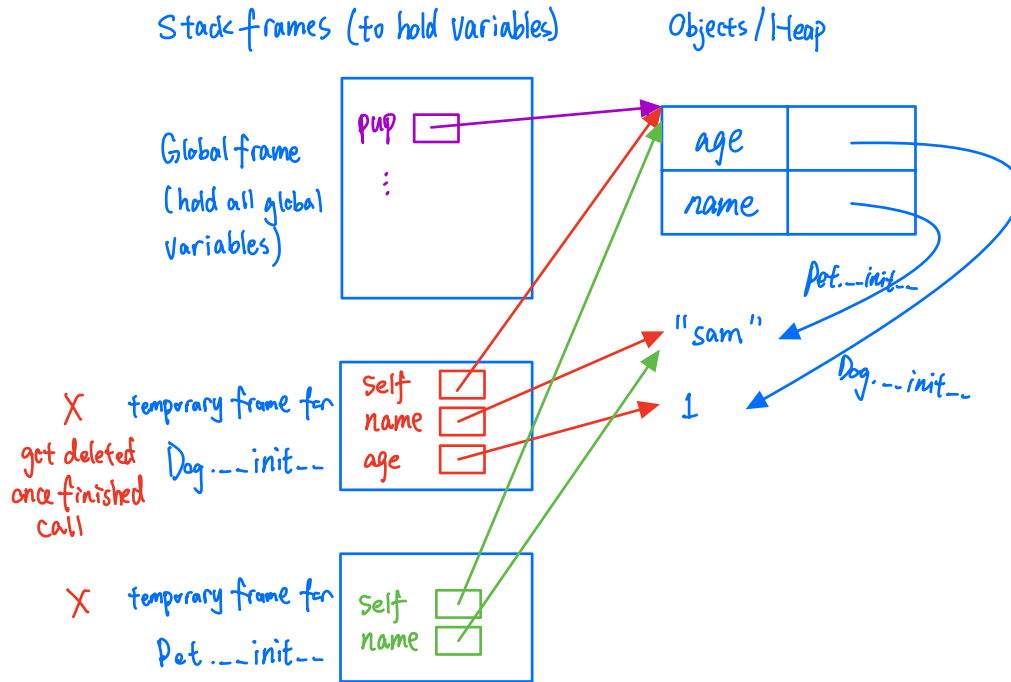
1

1

1. the parent class of Dog is Pet. Does Pet have a parent type? If so, what is it?

2. how many arguments does line C pass?

3. how many arguments does line B pass?

4. on another paper, draw what the frames and object(s) will look like after line A. (check with PythonTutor)

```python
class Pet:
    def __init__(self, name):
        self.name = name # A

class Dog(Pet):
    def __init__(self, name, age):
        self.age = age
        Pet.__init__(self, name) # B

pup = Dog("Sam", 1) # C
```


Stack frames (to hold variables)   Objects / Heap

Global frame
(hold all global
variables)

pup

age
name

Pet.__init__

Dog.__init__

"sam"

1

X   temporary frame for
get deleted   Dog.__init__
once finished
call

self
name
age

X   temporary frame for
Pet.__init__
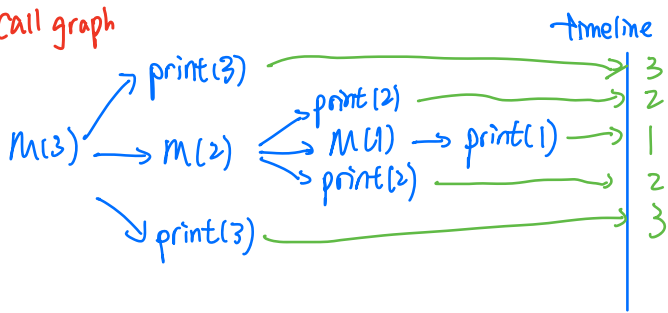
self
name

```
def M(n):
    print(n)
    if n > 1:
        M(n-1)
        print(n)

# what does M(3) print?
```
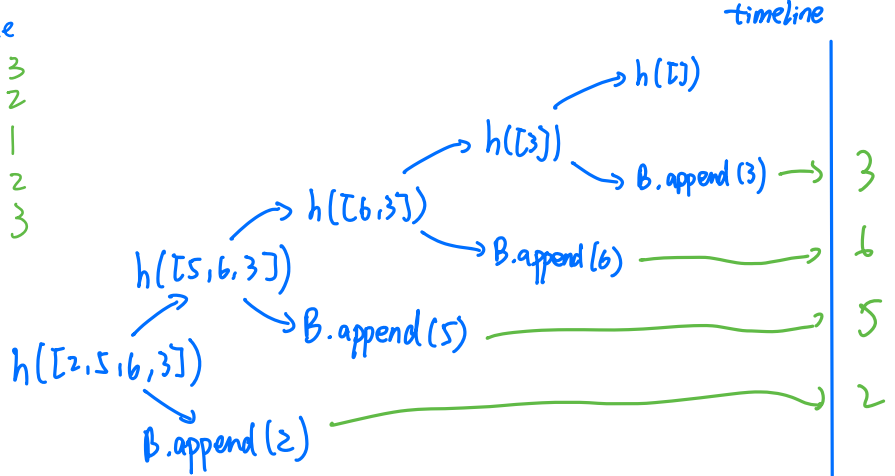
```
B = []
def h(A):
    if len(A) > 0:
        h(A[1:])
        B.append(A[0])
h([2, 5, 6, 3])
# what is in B?
```
= [3,6,5,2]

Call graph   timeline

print(3) ————————————→ 3
    print(2) ———————→ 2
M(3)→ M(2) → M(1) → print(1) →| 1
    print(2) ————————→ 2
print(3) ———————————→ 3

                                    timeline

→ h([])
→ h([3])
→ B.append(3) → | 3
→ h([6,3])
→ B.append(6) ———————→ | 6
h([5,6,3])
→ B.append(5) ———————————→ | 5
h([2,5,6,3])
→ B.append(2) ———————————————→ | 2
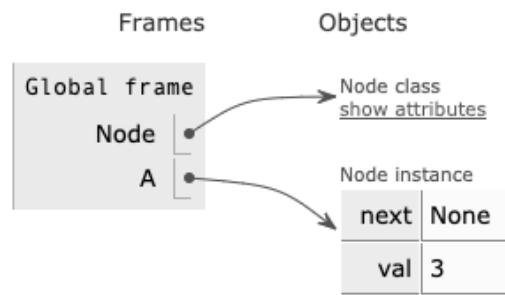
```
class Node:
    def __init__(self, val):
        self.val = val
        self.next = None

    def tot(self):
        if self.next == None:
            return self.val
        return self.val + self.next.tot()

    def __getitem__(self, idx):
        if idx == 0:
            return self.val
        return self.next[idx-1]

A = Node(3)
B = Node(5)
C = Node(7)
A.next = B
B.next = C
```

Frames   Objects

Global frame   → Node class
    Node •   show attributes

    A •   Node instance
                    | next | None |
                    | val | 3 |

1. finish the PythonTutor picture on the right
2. what is C.tot()? B.tot()? A.tot()?
3. what is A[0]? A[2]?
4. what kind of error does A[-1] produce?
5. how would the PythonTutor change if we
   added C.next = A?
6. what would C[3] be, given above change?
7. what would A.tot() do, give above change?