Trees

- [branches]

- tree recursion
  - ↳ multiple calls per layer

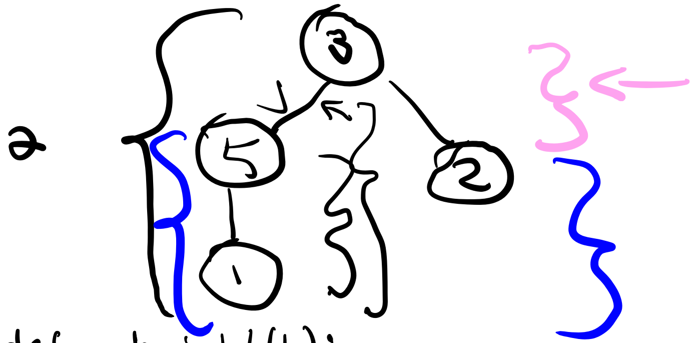
- your best friends are:
  - base cases
    - ↳ ifis_leaf(t) : ⟶
    - ↳ if label(t) == "berry" :
  - recursive calls
    - ↳ for b in branches(t):
    - ↳ [______ for b in branches(t)]
      - ↳ max min sum any all

[[ ]]

for b in branches(t):

Height: (6)

```
t = tree(3, [tree(5, [tree(1)]), tree(2)])
```



```
def height(t):
    if is_leaf(t):
        return 0
    return 1 + max([height(b) for b in branches(t)])
```

[1, 0]   []

1 + 1

```
return 1 + max([height(b) for b in branches(t)] + [-1])
```

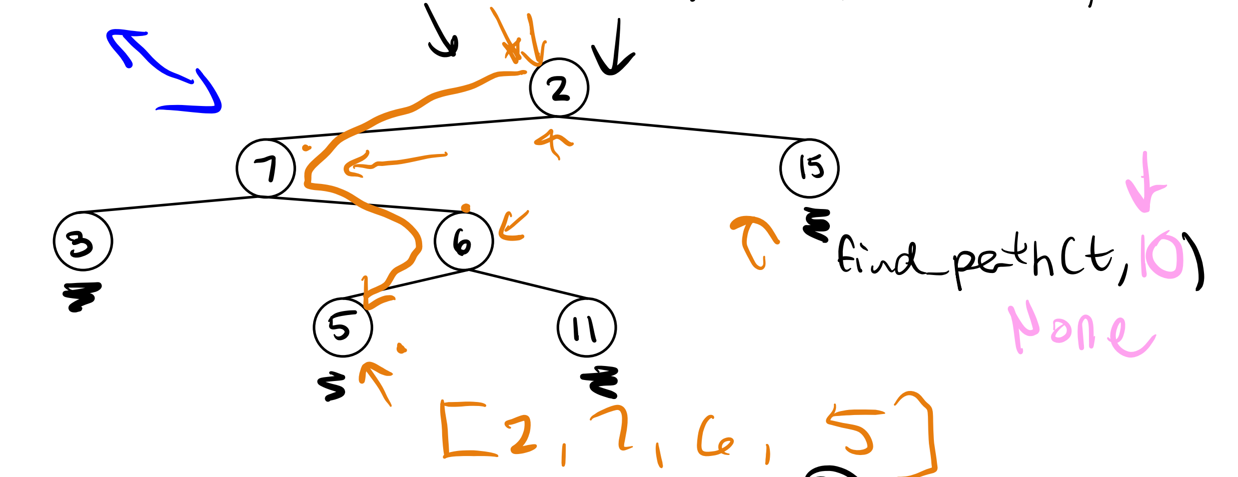
max([-1]) → -1

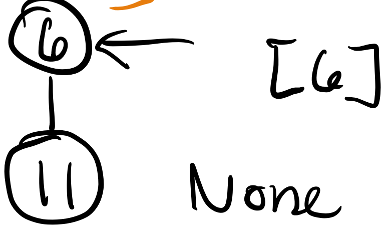
1+(-1) → 0

x if a == b else y

Find Path: (8)

t = tree(2, [tree(7, [tree(3), tree(6,[tree(5), tree(11)])]), tree(15)])





```
def find_path(tree, x):
    if label(tree) == x:
        return list(label(tree))
    ~~result = None~~

    for b in branches(tree):
        if find_path(b) != None:
    return ~~result~~ = [label(tree)] + find_path(b)
    ~~return result~~
```

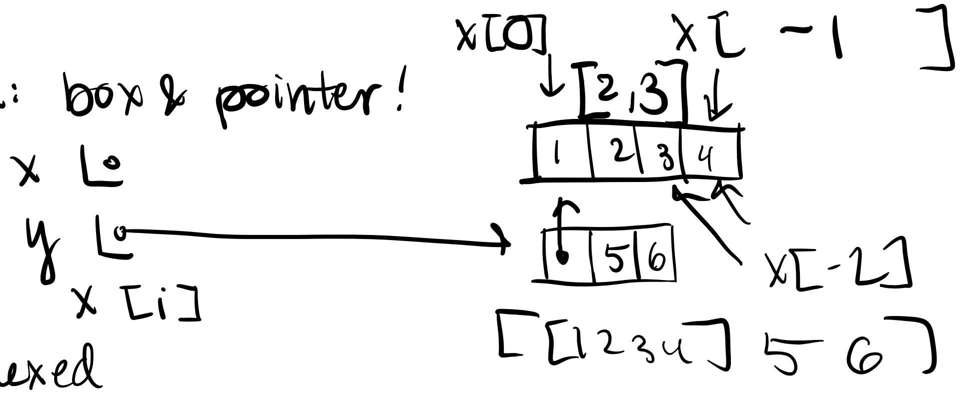
Lists:

x=

- Objects
- env dia: box & pointer!

x=[1,2,3,4] ←

y=[x,5,6]



- indexing: x[i]
  - 0 indexed
    - ↳ first item is @ index 0
    - ↳ last item is @ index n-1 (can say x[-1])
    - ↳ this is n items total 1 3

n = len(x) x[beg:end]

slicing

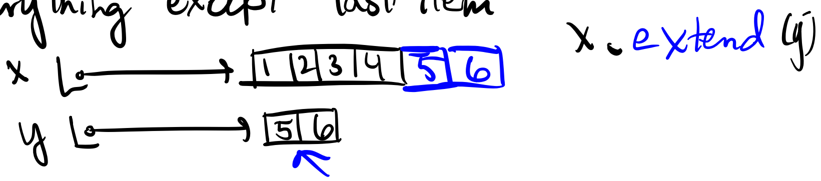
→ MAKES A COPY (x[:] for complete copy)

→ beginning is inclusive, ending exclusive

→ [1:] = "second item to the end"

→ [:-1] : "everything except last item"

- Mutation



→ x.append(arg)

→ makes a new box, shoves stuff in

→ arg can be anything

→ extend (arg)

→ arg must be a list

→ is probably what your intuition gets

→ MODIFIES ORIGINAL

[] is false-y

Max Product:

[10, 3, 1, 9, 2]

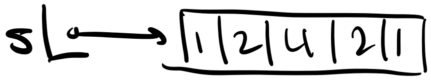
[5, 10, 5, 10, 5]

[ ]

```
def max_product(s):
```

Add This Many:

```
s = [1, 2, 4, 2, 1]
```



```
add_this_many (1, 5, s)
add_this_many (2, 2, s)
def add_this_many(x, el, s):
    for i in s
        if i == x:
            counter += 1
    s+ = [el] * counter

    s = s + ______

    s.extend ([el]·counter)
```

ad