

## Data Examples

---

## Announcements

## Examples: Lists

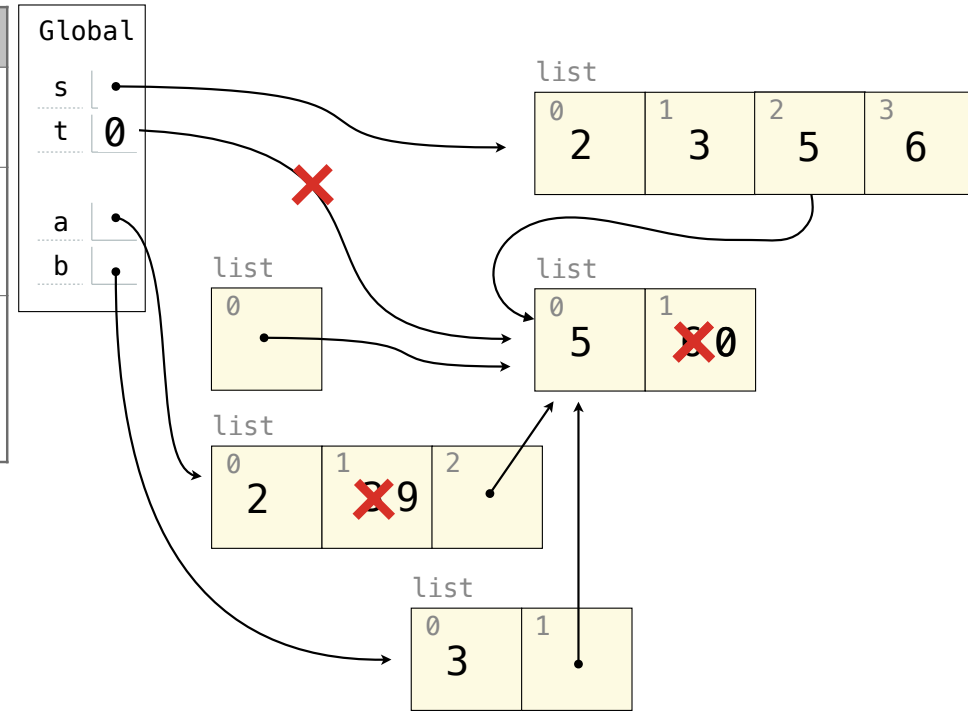
## Lists in Environment Diagrams

Assume that before each example below we execute:

`s = [2, 3]`

`t = [5, 6]`

Operation	Example	Result
<b>append</b> adds one element to a list	<code>s.append(t)</code> <code>t = 0</code>	<code>s</code> → [2, 3, [5, 6]] <code>t</code> → 0
<b>extend</b> adds all elements in one list to another list	<code>s.extend(t)</code> <code>t[1] = 0</code>	<code>s</code> → [2, 3, 5, 6] <code>t</code> → [5, 0]
<b>addition &amp; slicing</b> create new lists containing existing elements	<code>a = s + [t]</code> <code>b = a[1:]</code> <code>a[1] = 9</code> <code>b[1][1] = 0</code>	<code>s</code> → [2, 3] <code>t</code> → [5, 0] <code>a</code> → [2, 9, [5, 0]] <code>b</code> → [3, [5, 0]]



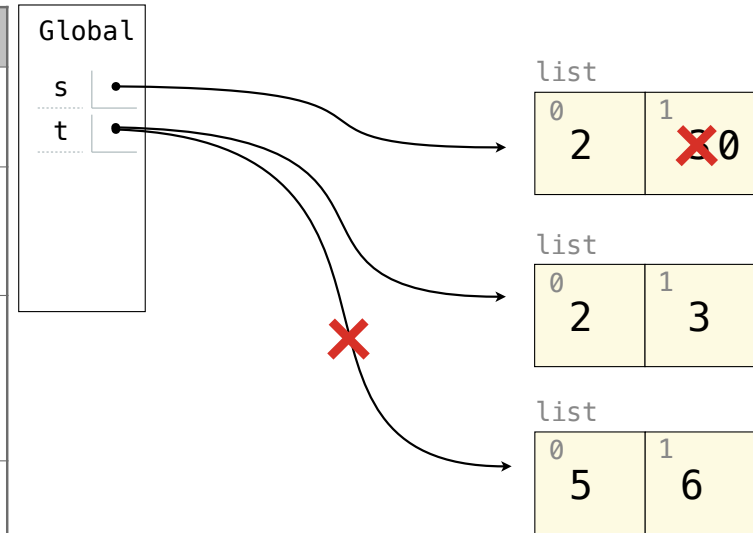
## Lists in Environment Diagrams

Assume that before each example below we execute:

`s = [2, 3]`

`t = [5, 6]`

Operation	Example	Result
<b>append</b> adds one element to a list	<code>s.append(t)</code> <code>t = 0</code>	<code>s</code> → [2, 3, [5, 6]] <code>t</code> → 0
<b>extend</b> adds all elements in one list to another list	<code>s.extend(t)</code> <code>t[1] = 0</code>	<code>s</code> → [2, 3, 5, 6] <code>t</code> → [5, 0]
<b>addition &amp; slicing</b> create new lists containing existing elements	<code>a = s + [t]</code> <code>b = a[1:]</code> <code>a[1] = 9</code> <code>b[1][1] = 0</code>	<code>s</code> → [2, 3] <code>t</code> → [5, 0] <code>a</code> → [2, 9, [5, 0]] <code>b</code> → [3, [5, 0]]
The <b>list</b> function also creates a new list containing existing elements	<code>t = list(s)</code> <code>s[1] = 0</code>	<code>s</code> → [2, 0] <code>t</code> → [2, 3]



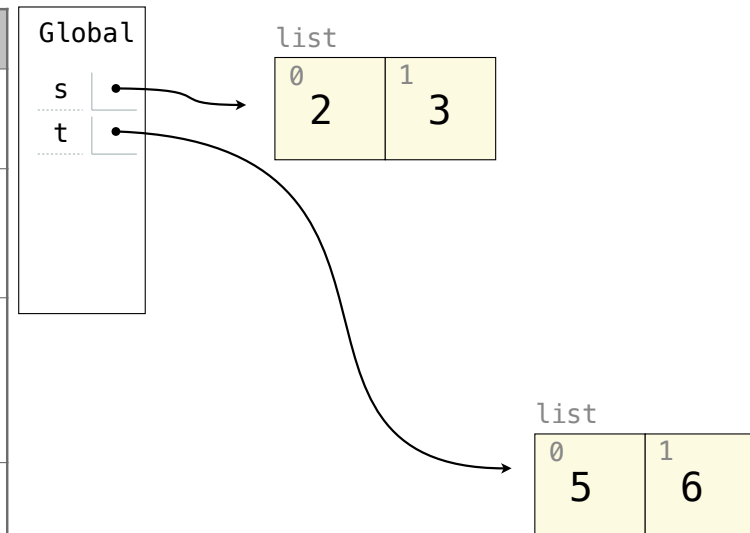
## Lists in Environment Diagrams

Assume that before each example below we execute:

`s = [2, 3]`

`t = [5, 6]`

Operation	Example	Result
<b>append</b> adds one element to a list	<code>s.append(t)</code> <code>t = 0</code>	<code>s</code> → [2, 3, [5, 6]] <code>t</code> → 0
<b>extend</b> adds all elements in one list to another list	<code>s.extend(t)</code> <code>t[1] = 0</code>	<code>s</code> → [2, 3, 5, 6] <code>t</code> → [5, 0]
<b>addition &amp; slicing</b> create new lists containing existing elements	<code>a = s + [t]</code> <code>b = a[1:]</code> <code>a[1] = 9</code> <code>b[1][1] = 0</code>	<code>s</code> → [2, 3] <code>t</code> → [5, 0] <code>a</code> → [2, 9, [5, 0]] <code>b</code> → [3, [5, 0]]
The <b>list</b> function also creates a new list containing existing elements	<code>t = list(s)</code> <code>s[1] = 0</code>	<code>s</code> → [2, 0] <code>t</code> → [2, 3]
<b>slice assignment</b> replaces a slice with new values	<code>s[0:0] = t</code> <code>s[3:] = t</code> <code>t[1] = 0</code>	



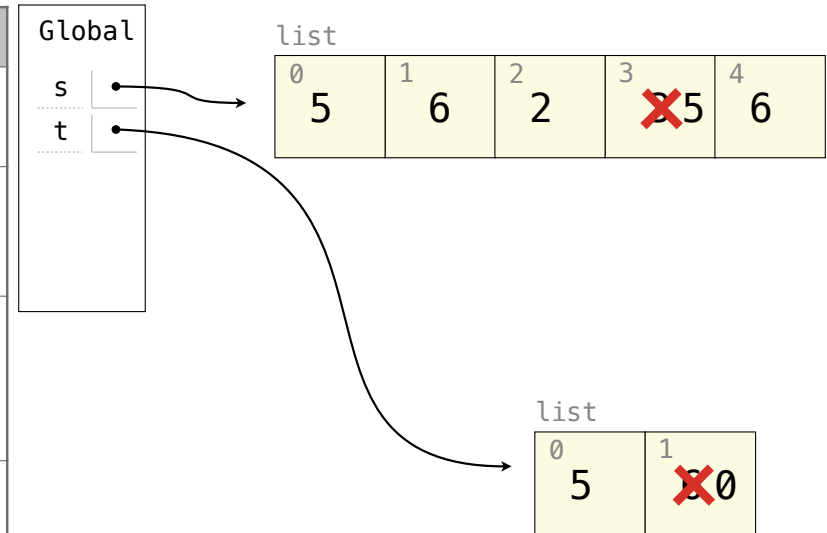
## Lists in Environment Diagrams

Assume that before each example below we execute:

`s = [2, 3]`

`t = [5, 6]`

Operation	Example	Result
<b>append</b> adds one element to a list	<code>s.append(t)</code> <code>t = 0</code>	<code>s</code> → [2, 3, [5, 6]] <code>t</code> → 0
<b>extend</b> adds all elements in one list to another list	<code>s.extend(t)</code> <code>t[1] = 0</code>	<code>s</code> → [2, 3, 5, 6] <code>t</code> → [5, 0]
<b>addition &amp; slicing</b> create new lists containing existing elements	<code>a = s + [t]</code> <code>b = a[1:]</code> <code>a[1] = 9</code> <code>b[1][1] = 0</code>	<code>s</code> → [2, 3] <code>t</code> → [5, 0] <code>a</code> → [2, 9, [5, 0]] <code>b</code> → [3, [5, 0]]
The <b>list</b> function also creates a new list containing existing elements	<code>t = list(s)</code> <code>s[1] = 0</code>	<code>s</code> → [2, 0] <code>t</code> → [2, 3]
<b>slice assignment</b> replaces a slice with new values	<code>s[0:0] = t</code> <code>s[3:] = t</code> <code>t[1] = 0</code>	<code>s</code> → [5, 6, 2, 5, 6] <code>t</code> → [5, 0]



## Lists in Environment Diagrams

---

Assume that before each example below we execute:

`s = [2, 3]`

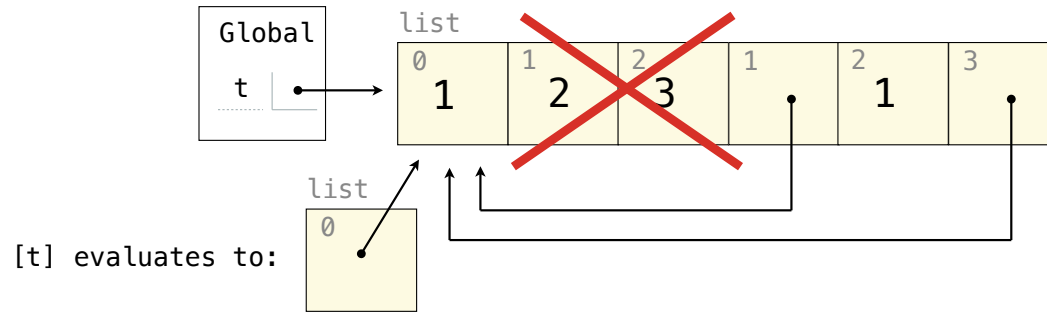
`t = [5, 6]`

Operation	Example	Result
<b>pop</b> removes & returns the last element	<code>t = s.pop()</code>	<code>s → [2]</code> <code>t → 3</code>
<b>remove</b> removes the first element equal to the argument	<code>t.extend(t)</code> <code>t.remove(5)</code>	<code>s → [2, 3]</code> <code>t → [6, 5, 6]</code>
<b>slice assignment</b> can remove elements from a list by assigning <code>[]</code> to a slice.	<code>s[:1] = []</code> <code>t[0:2] = []</code>	<code>s → [3]</code> <code>t → []</code>



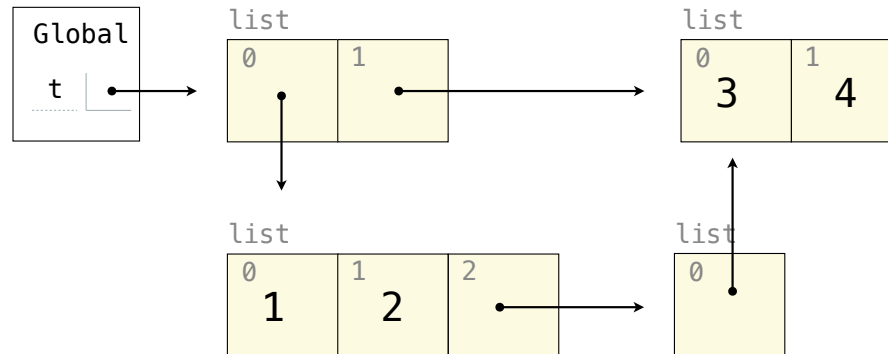
## Lists in Lists in Lists in Environment Diagrams

```
t = [1, 2, 3]
t[1:3] = [t]
t.extend(t)
```



[1, [...], 1, [...]]

```
t = [[1, 2], [3, 4]]
t[0].append(t[1:2])
```



[[1, 2, [[3, 4]]], [3, 4]]

Examples: Objects

## Land Owners

Instance attributes are found before class attributes; class attributes are inherited

```
class Worker:
    greeting = 'Sir'
    def __init__(self):
        self.elf = Worker
    def work(self):
        return self.greeting + ', I work'
    def __repr__(self):
        return Bourgeoisie.greeting
```

```
class Bourgeoisie(Worker):
    greeting = 'Peon'
    def work(self):
        print(Worker.work(self))
        return 'I gather wealth'
```

```
jack = Worker()
john = Bourgeoisie()
jack.greeting = 'Maam'
```

```
>>> Worker().work()
'Sir, I work'
```

```
>>> jack
Peon
```

```
>>> jack.work()
'Maam, I work'
```

```
>>> john.work()
Peon, I work
'I gather wealth'
```

```
>>> john.elf.work(john)
'Peon, I work'
```

<class Worker>

greeting: 'Sir'

<class Bourgeoisie>

greeting: 'Peon'

jack <Worker>

elf: \_\_\_\_\_  
greeting: 'Maam'

john <Bourgeoisie>

elf: \_\_\_\_\_

## Examples: Iterables & Iterators

## Using Built-In Functions & Comprehensions

---

What are the indices of all elements in a list `s` that have the smallest absolute value?

`[-4, -3, -2, 3, 2, 4]`  
0 1 2 3 4 5  $\triangleright$  `[2, 4]`      `[1, 2, 3, 4, 5]`  $\triangleright$  `[0]`

What's the largest sum of two adjacent elements in a list `s`? (Assume `len(s) > 1`)

`[-4, -3, -2, 3, 2, 4]`  $\triangleright$  `6`      `[-4, 3, -2, -3, 2, -4]`  $\triangleright$  `1`

Create a dictionary mapping each digit `d` to the lists of elements in `s` that end with `d`.

`[5, 8, 13, 21, 34, 55, 89]`  $\triangleright$  `{1: [21], 3: [13], 4: [34], 5: [5, 55], 8: [8], 9: [89]}`

Does every element equal some other element in `s`?

`[-4, -3, -2, 3, 2, 4]`  $\triangleright$  `False`      `[4, 3, 2, 3, 2, 4]`  $\triangleright$  `True`

## Examples: Linked Lists

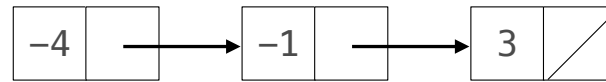
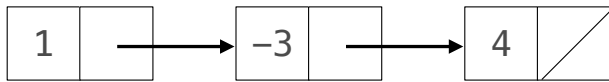
## Linked List Exercises

---

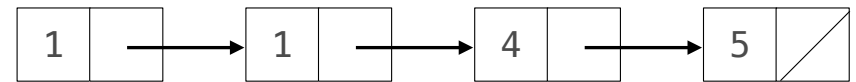
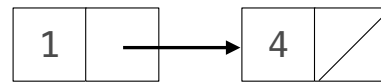
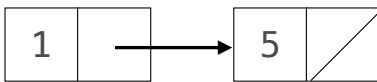
Is a linked list *s* ordered from least to greatest?



Is a linked list *s* ordered from least to greatest by absolute value (or a key function)?



Create a sorted Link containing all the elements of both sorted Links *s* & *t*.



Do the same thing, but never call Link.

