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

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

jack = Worker()
john = Bourgeoisie()

jack.greeting = 'Maam'
john.elf = Worker()

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

>>> jack
'Peon'

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

>>> john
'Peon'

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

### Using Built-In Functions & Comprehensions
- **Indices of smallest absolute value**
  ```
  >>> s = [-4, -3, -2, 3, 2, 4]
  >>> indices = [i for i, x in enumerate(s) if x == min(s, key=abs)]
  >>> indices
  [1, 2]
  ```
- **Largest sum of adjacent elements**
  ```
  >>> s = [-4, -3, -2, 3, 2, 4]
  >>> max_sum = max(s[i] + s[i+1] for i in range(len(s)-1))
  >>> max_sum
  7
  ```
- **Dictionary mapping digits**
  ```
  >>> s = [5, 8, 13, 21, 34, 55, 89]
  >>> mapping = {d: [x for x in s if str(x).endswith(str(d))] for d in range(10)}
  >>> mapping
  {0: [5], 1: [13, 21], 2: [2], 3: [34], 4: [5, 55], 5: [89], 6: [], 7: [], 8: [8], 9: []}
  ```
- **Equal element check**
  ```
  >>> s = [-4, -3, -2, 3, 2, 4]
  >>> any(x == y for x, y in zip(s, s[1:]))
  True
  >>> s = [5, 8, 13, 21, 34, 55, 89]
  >>> any(x == y for x, y in zip(s, s[1:]))
  False
  ```

### Linked List Exercises
- **Ordered from least to greatest**
  ```
  >>> link1 = Link(1, 2, 3, 4, 5)
  >>> link2 = Link(5, 4, 3, 2, 1)
  >>> link3 = link1.sort
  >>> link3
  Link(1, 2, 3, 4, 5)
  ```
- **Ordered by absolute value**
  ```
  >>> link1 = Link(1, 2, 3, 4, 5)
  >>> link2 = Link(5, 4, 3, 2, 1)
  >>> link1.sort_abs
  Link(1, 2, 3, 4, 5)
  ```
- **Create a sorted Link**
  ```
  >>> link1 = Link(1, 2, 3, 4, 5)
  >>> link2 = Link(5, 4, 3, 2, 1)
  >>> sorted_link = Link.sorted_link(link1, link2)
  >>> sorted_link
  Link(1, 1, 2, 3, 3, 4, 5)
  ```