CPE/CSC 101

Today

Functions on lists continued

- indices
- range

Nested Loops

```
new_list = [ x * 2  for  x  in  L  ]
```

if ---

```
sum = 0
for x in L:
    sum = sum + x
```

Linear Search

```
def search(nums: List[int], n: int) -> bool:
    for el in nums:
        if el == n:
            return True

    return False
```

Linear Search - index of element

→ range function

range(lowerbound, upperbound)

↑ included ↑ excluded

L = list(range(0, 3))

L = list(range(3))

↑ upperbound

lowerbound is 0

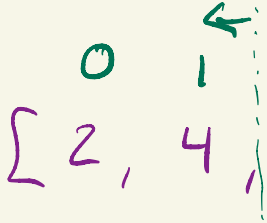
range(0, 10, 2)

↑ step

Linear Search : index @ which n appears

```
def search(nums: List[int], n: int) -> Optional[int]:
    for idx in range(0, len(nums)):
        if n == nums[idx]:
            return idx
    return None
```

range(len(nums))



invariant: values @ indices < idx != n

every element? Y

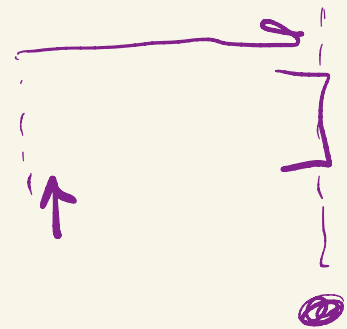
el/idx ? idx

accum idx

max list of numbers

- every element?
- idx/el?
- accumulator/invariant?

[



```
def max(nums: List[float]) -> float:
    max_num = nums[0]
    for el in nums:
        if el > max_num:
            max_num = el
    return max_num
```

max_num = 10

[4, 9, 3, 2, 10]

[-2, -3, -1]

[ ]

max_num > all elements before

```
def max(nums: List[float]) -> float:
    max_num = nums[0]
    for el in nums:
        if el > max_num:
            max_num = el

    return max_num
```

```
assertAlmostEqual([4, 7, 9], 9)
assertAlmostEqual([]  ,      )
```

```
def max(nums: List[float]) -> Optional[float]:
    if nums:
        max_num = nums[0]
        for el in nums:
            if el > max_num:
                max_num = el

        return max_num
    else:
        return None
```

```
if nums == []:
```

---

truthy

all non-falsey

bool

True/False

falsey

False, 0, [], None

bool?

```
if nums:
```