LeetCamp

A project dedicated to DS&A

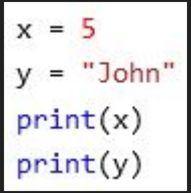
Agenda

1. Python Review
2. Time Complexity Review
3. Q&A Time

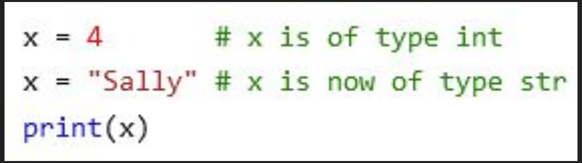
Python Review

Variables

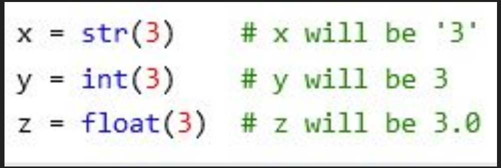
- Numbers
  - ints
  - Floats
- Strings
  - index
- Lists
  - index
  - append
  - pop
  - Len
- Sets
  - No duplicates
  - Add
  - set()



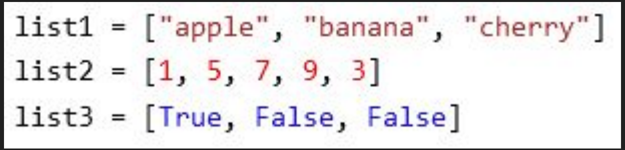
```python
x = 5
y = "John"
print(x)
print(y)
```



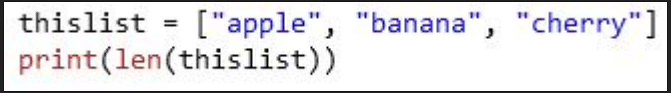
```python
x = 4       # x is of type int
x = "Sally" # x is now of type str
print(x)
```



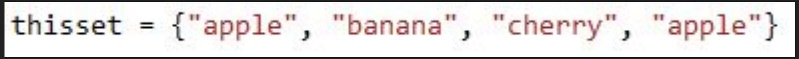
```python
x = str(3)    # x will be '3'
y = int(3)    # y will be 3
z = float(3)  # z will be 3.0
```



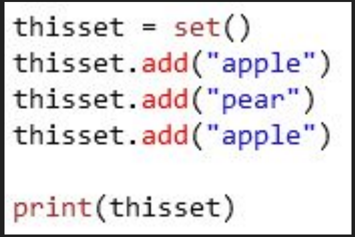
```python
list1 = ["apple", "banana", "cherry"]
list2 = [1, 5, 7, 9, 3]
list3 = [True, False, False]
```



```python
thislist = ["apple", "banana", "cherry"]
print(len(thislist))
```



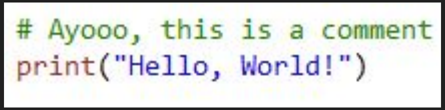
```python
thisset = {"apple", "banana", "cherry", "apple"}
```



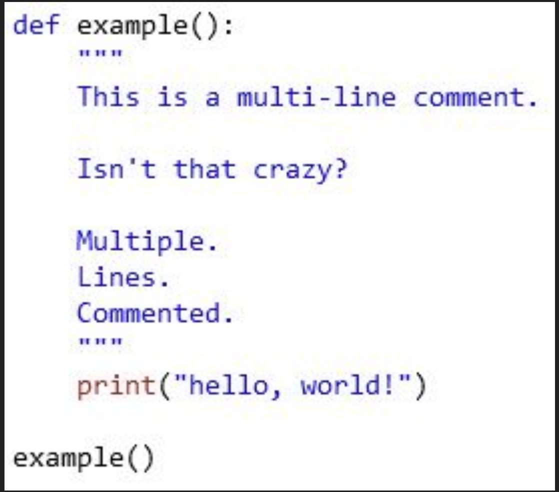
```python
thisset = set()
thisset.add("apple")
thisset.add("pear")
thisset.add("apple")

print(thisset)
```

Comments

- Single-line
- Multi-line



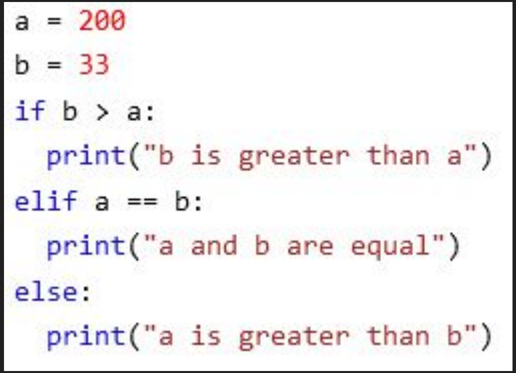
```python
# Ayooo, this is a comment
print("Hello, World!")
```



```python
def example():
    """
    This is a multi-line comment.

    Isn't that crazy?

    Multiple.
    Lines.
    Commented.
    """
    print("hello, world!")

example()
```

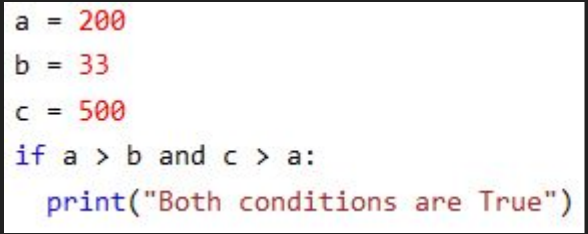
Control Flow

- If, elif, else

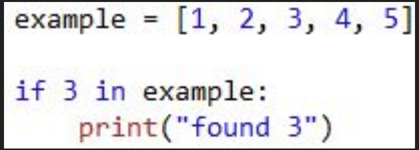


```python
a = 200
b = 33
if b > a:
  print("b is greater than a")
elif a == b:
  print("a and b are equal")
else:
  print("a is greater than b")
```

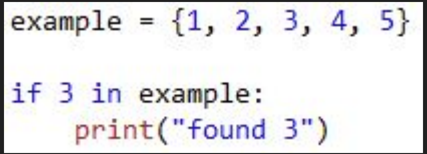
- ==, !=, <, <=, >, >=
- and, or, not
- in



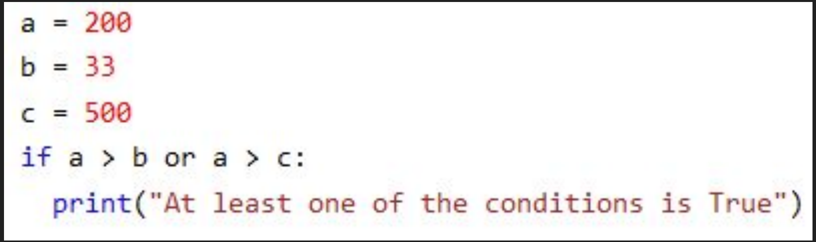
```python
a = 200
b = 33
c = 500
if a > b and c > a:
  print("Both conditions are True")
```



```python
example = [1, 2, 3, 4, 5]

if 3 in example:
    print("found 3")
```



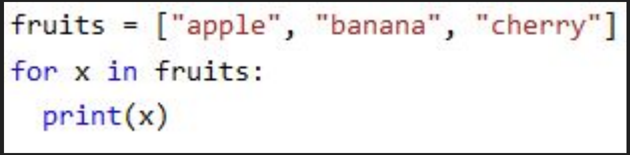
```python
example = {1, 2, 3, 4, 5}

if 3 in example:
    print("found 3")
```



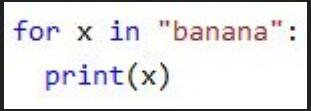
```python
a = 200
b = 33
c = 500
if a > b or a > c:
  print("At least one of the conditions is True")
```

Loops

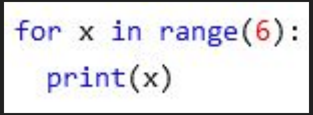
- for
- range()
- while
- break and continue



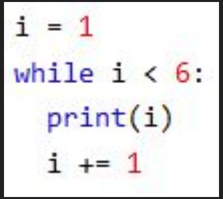
```python
fruits = ["apple", "banana", "cherry"]
for x in fruits:
  print(x)
```



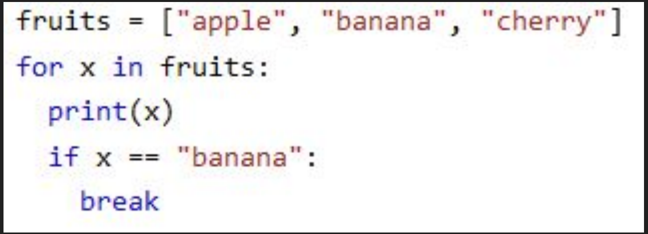
```python
for x in "banana":
  print(x)
```



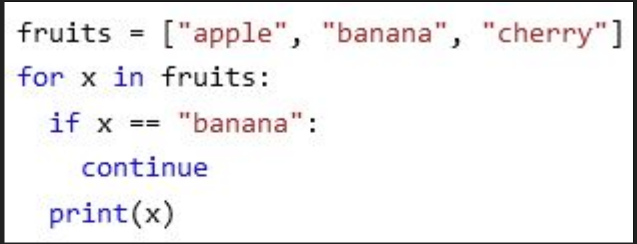
```python
for x in range(6):
  print(x)
```



```python
i = 1
while i < 6:
  print(i)
  i += 1
```



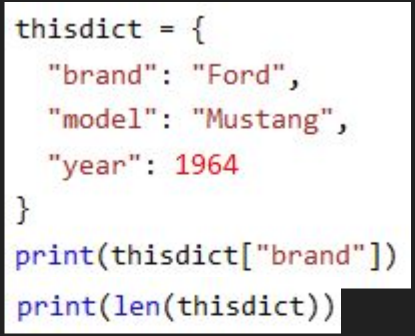
```python
fruits = ["apple", "banana", "cherry"]
for x in fruits:
  print(x)
  if x == "banana":
    break
```



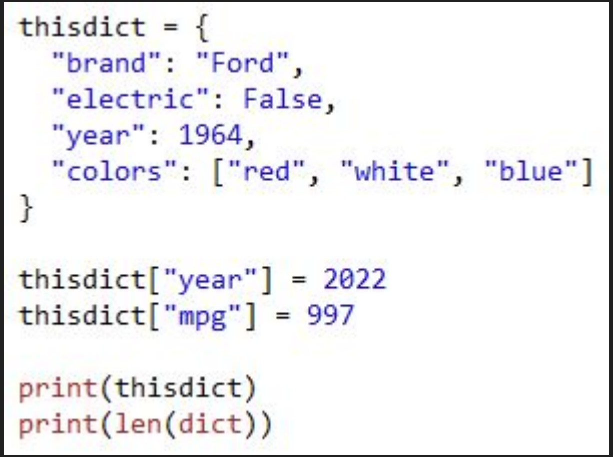
```python
fruits = ["apple", "banana", "cherry"]
for x in fruits:
  if x == "banana":
    continue
  print(x)
```

Dictionaries

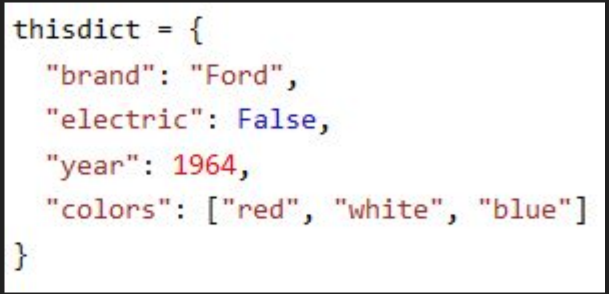
- key: value pairs
- Adding new key: value pair
- Finding if a key is in a dictionary using *in*
- for k, v in dictionary



```
thisdict = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
print(thisdict["brand"])
print(len(thisdict))
```



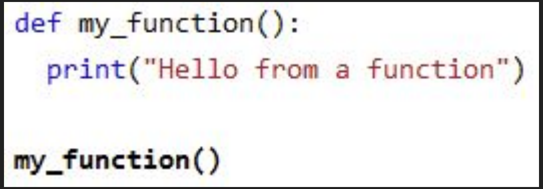
```
thisdict = {
  "brand": "Ford",
  "electric": False,
  "year": 1964,
  "colors": ["red", "white", "blue"]
}

thisdict["year"] = 2022
thisdict["mpg"] = 997

print(thisdict)
print(len(dict))
```



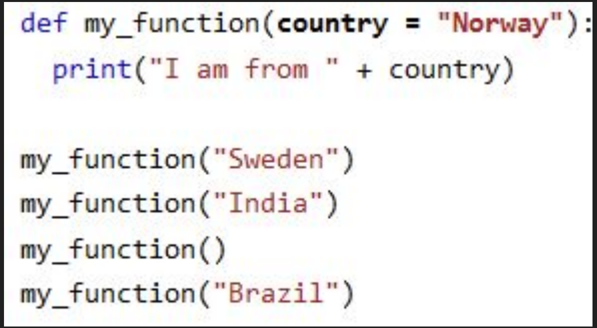
```
thisdict = {
  "brand": "Ford",
  "electric": False,
  "year": 1964,
  "colors": ["red", "white", "blue"]
}
```

Functions

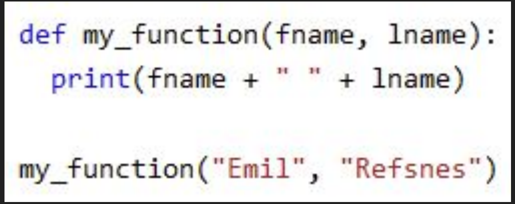
- defining functions
- Arguments
  - Defaults
  - typing
- return



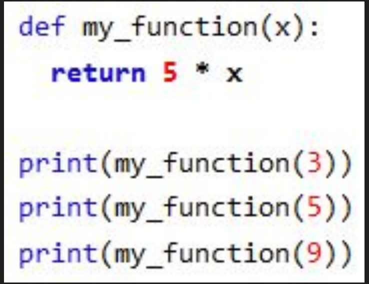
```python
def my_function():
  print("Hello from a function")

my_function()
```



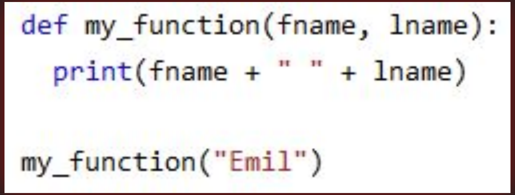
```python
def my_function(country = "Norway"):
  print("I am from " + country)

my_function("Sweden")
my_function("India")
my_function()
my_function("Brazil")
```



```python
def my_function(fname, lname):
  print(fname + " " + lname)

my_function("Emil", "Refsnes")
```



```python
def my_function(x):
  return 5 * x

print(my_function(3))
print(my_function(5))
print(my_function(9))
```



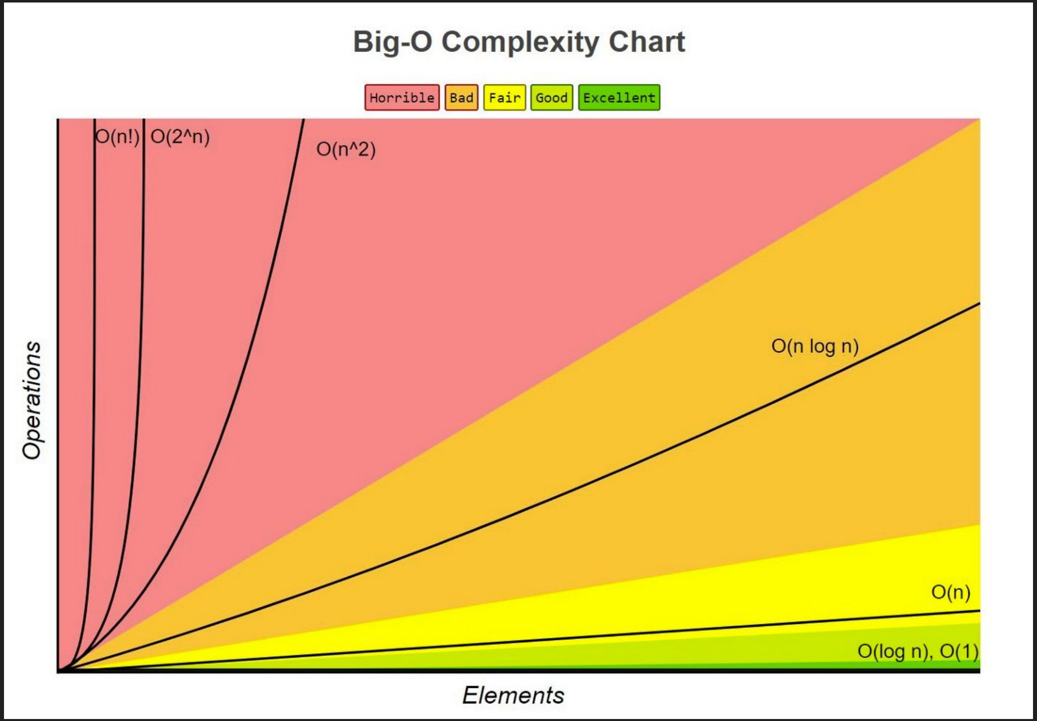
```python
def my_function(fname, lname):
  print(fname + " " + lname)

my_function("Emil")
```

Time Complexity

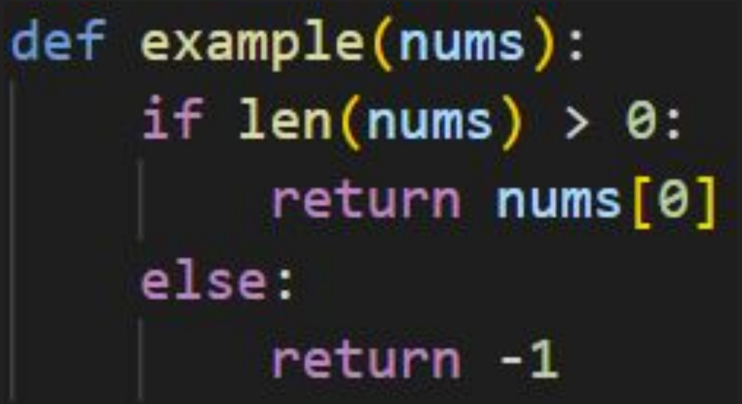
What is Time Complexity?

- How we measure the performance of your algorithms
- It’s necessary how well, or how poorly, your code performs as more data is provided for the algorithm to process
- Various levels of complexity:
    1. O(1)
    2. O(log N)
    3. O(N)
    4. O(k * log N)
    5. O(N^2)
    6. O(2^N)
    7. O(N!)

Time Complexity - Chart

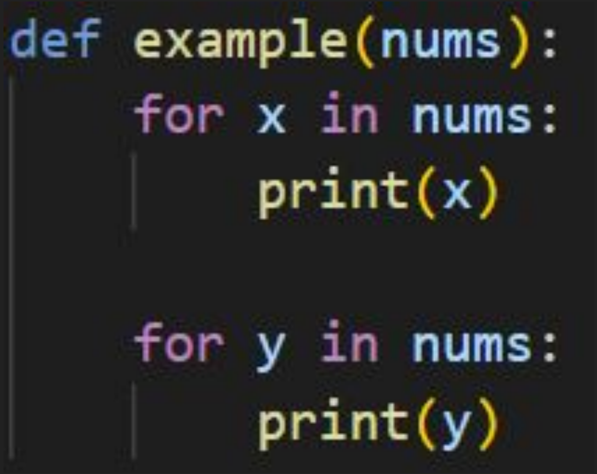


https://www.bigocheatsheet.com/



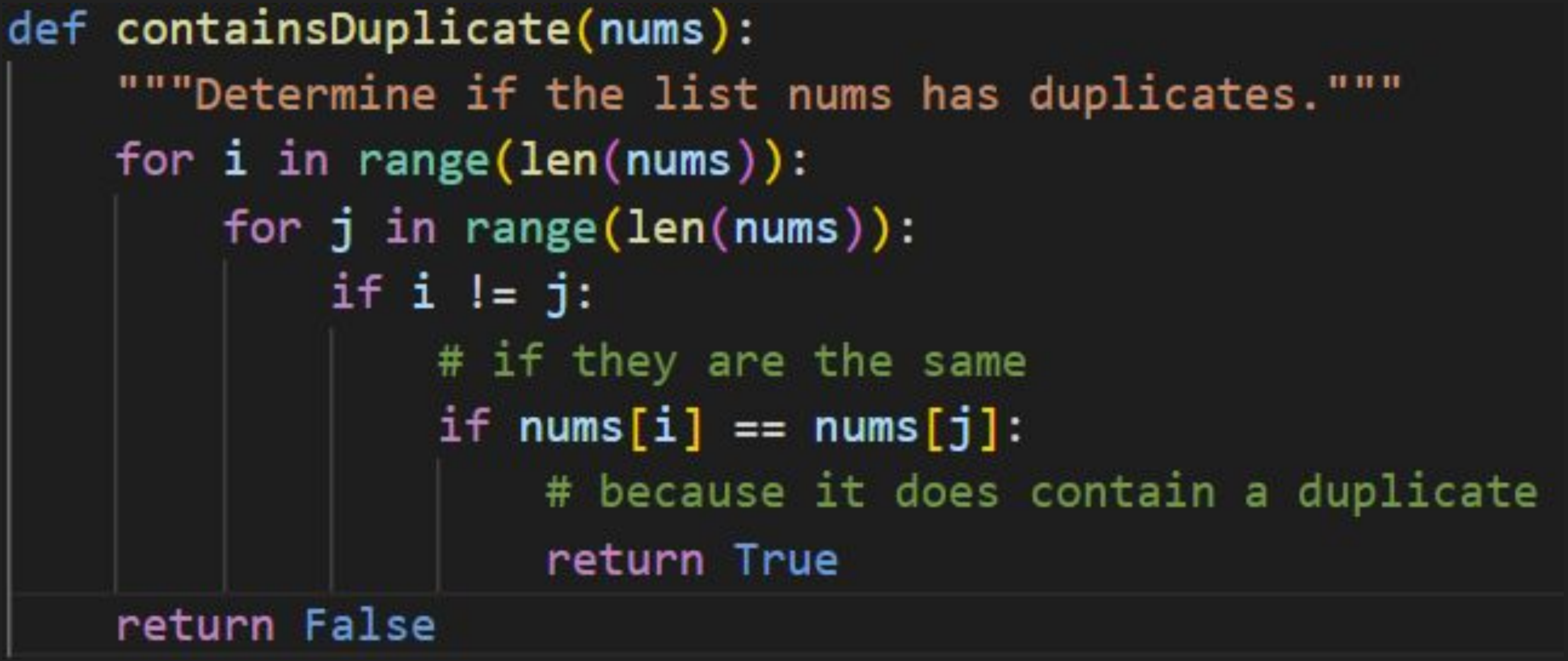
```python
def example(nums):
    if len(nums) > 0:
        return nums[0]
    else:
        return -1
```

Time Complexity: O(1)
Space Complexity: O(1)



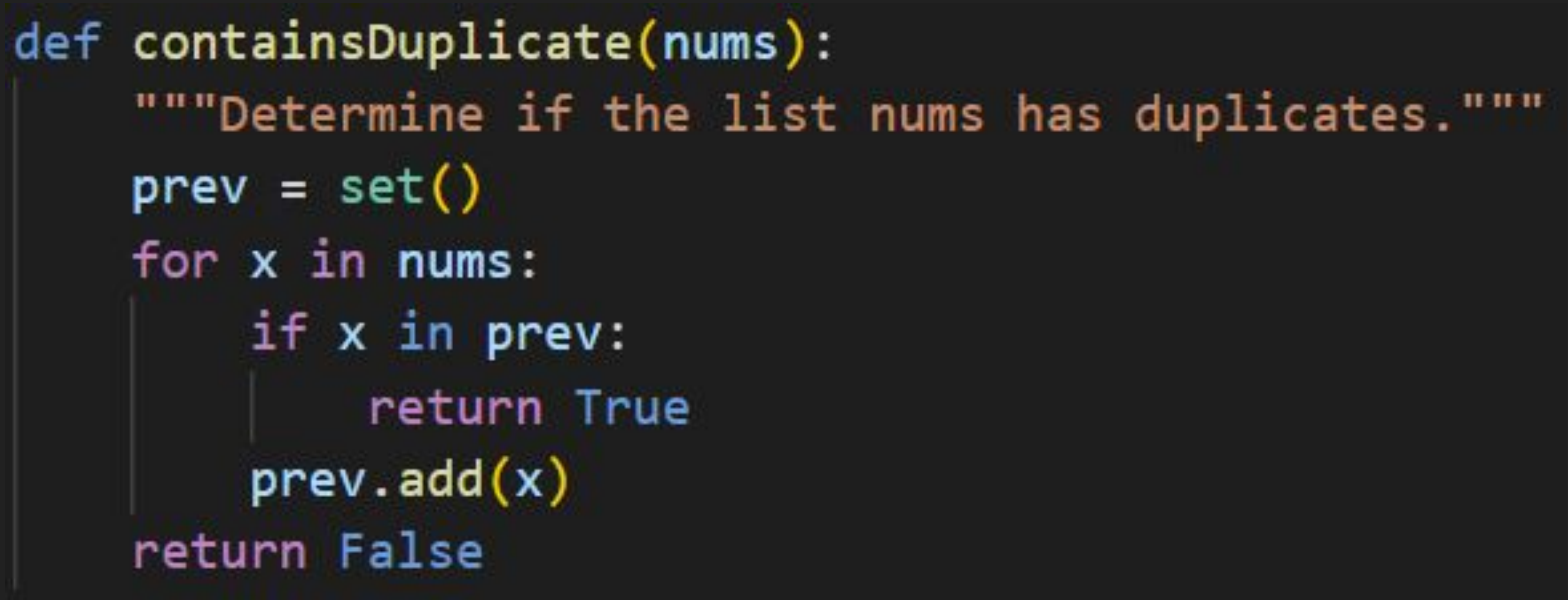
```python
def example(nums):
    for x in nums:
        print(x)

    for y in nums:
        print(y)
```

Time Complexity: O(N)
Space Complexity: O(1)



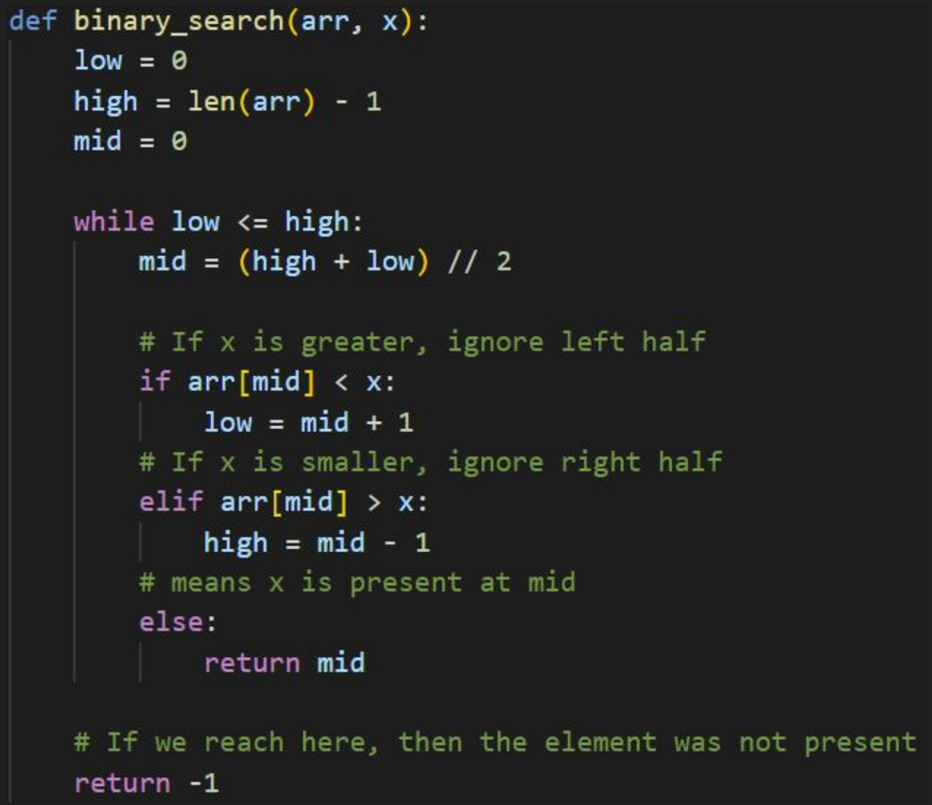
```python
def containsDuplicate(nums):
    """Determine if the list nums has duplicates."""
    for i in range(len(nums)):
        for j in range(len(nums)):
            if i != j:
                # if they are the same
                if nums[i] == nums[j]:
                    # because it does contain a duplicate
                    return True
    return False
```

Time Complexity: O(n^2)   aka: Hot Garbage
Space Complexity: O(1)



```python
def containsDuplicate(nums):
    """Determine if the list nums has duplicates."""
    prev = set()
    for x in nums:
        if x in prev:
            return True
        prev.add(x)
    return False
```

Time Complexity: O(n)
Space Complexity: O(n)



```python
def binary_search(arr, x):
    low = 0
    high = len(arr) - 1
    mid = 0

    while low <= high:
        mid = (high + low) // 2

        # If x is greater, ignore left half
        if arr[mid] < x:
            low = mid + 1
        # If x is smaller, ignore right half
        elif arr[mid] > x:
            high = mid - 1
        # means x is present at mid
        else:
            return mid

    # If we reach here, then the element was not present
    return -1
```

Time Complexity: O(log n)
Space Complexity: O(1)

Last Week’s Solutions

Practice Problems (given last week)

- Easier:
  - Remove All Adjacent Duplicates in String - Solution
  - Implement Queue Using Stacks
  - Remove Outermost Parentheses

- A bit more challenging:
  - Minimum Remove to Make Valid Parenthesis

- Pretty hard:
  - Design a Text Editor
  - Parsing a Boolean Expression

Until next time...

Keep practicing

Practice Problems

- https://neetcode.io/practice
  - Try working on the easy questions in the "NeetCode 150" tab
  - Consider doing this order:
    - Arrays & Hashing
    - Two Pointers
    - Stack