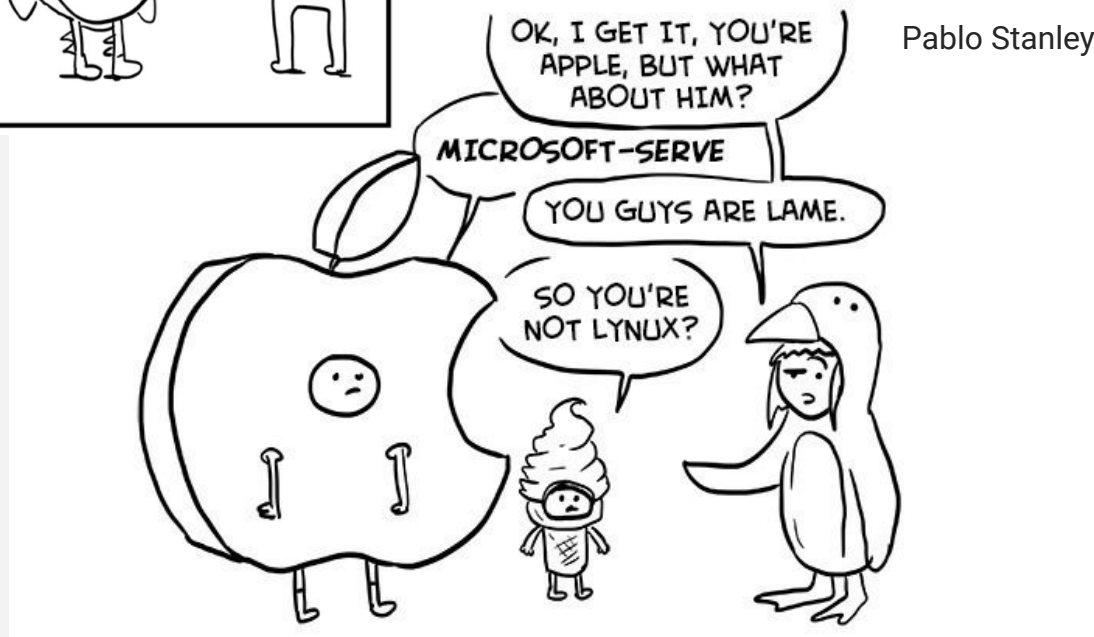
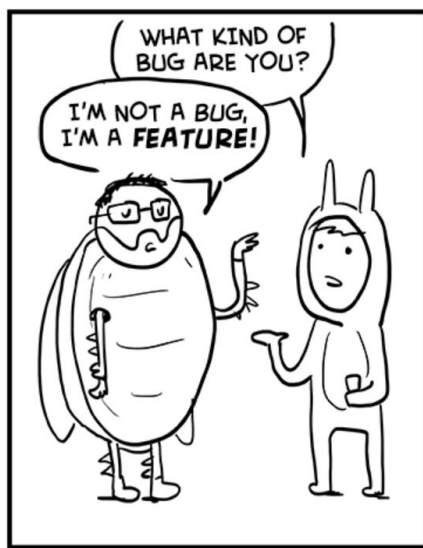


# CSc 110

## Mapping, Dictionaries

Adriana Picoral  
(she/her/hers)



# Data Structure

- **A data-structure is a way of arranging and organizing data in a computer program**
- Python has several useful data-structures built into the language
  - One is a **list** (already covered)
  - Another, **dictionary**
- A data type that stores a single value is generally not considered to be a data structure
  - Integers, boolean, floats

# Mapping

- Many data structures allow data to be stored and retrieved using a concept called ***mapping***
- ***Mapping*** is the process of associating one value with another (a **key** with a **value**)
  - Sometimes also referred to as Hashing or Associativity

# Mapping example

We can map **addresses** to **buildings**

**1040 E. 4th St, Tucson, AZ 85721 → Gould Simpson Building**



**1428 E. University Blvd, Tucson, AZ 85719 → Old Main Building**



**1737 E. University Blvd, Tucson, AZ 85719 → Ina Gittings Building**



# Mapping example:

We can map **Words** to **Definitions**

(<http://webstersdictionary1828.com>)

**Human** → Belonging to man or mankind; pertaining or relating to the race of man; as a human voice; human shape; human nature; human knowledge; human life. . . .

**Weapon** → Any instrument of offense; any thing used or designed to be used in destroying or annoying an enemy. The weapons of rude nations are clubs, stones and bows and arrows. Modern weapons of war are swords, muskets, pistols, cannon and the like. . . .

**Attack** → To assault; to fall upon with force; to assail, as with force and arms. It is the appropriate word for the commencing act of hostility between armies and navies. . . .

# Mapping

- Lists (sort of) map *keys* to *values* too!
  - Indexes *of* the list are the *keys*
  - Elements *in* the list are the *values*
- Keys (indexes) are used to accessing or modifying the elements in the list

# Mapping and Lists

```
numbers = [12, 49, -2, 26, 5, 17, -6]
```

index	0	1	2	3	4	5	6
value	12	49	-2	26	5	17	-6

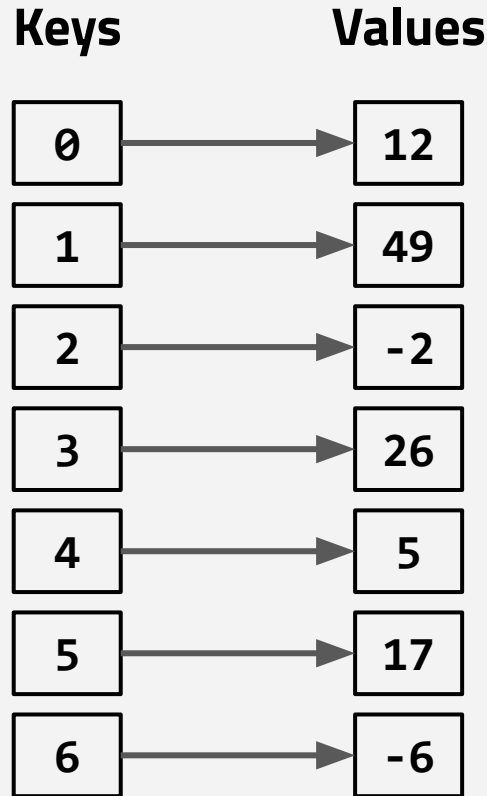
- What are the keys?
- What are the values?
- Which keys “map” to which values?

# Mapping and Lists

```
numbers = [12, 49, -2, 26, 5, 17, -6]
```

index	0	1	2	3	4	5	6
value	12	49	-2	26	5	17	-6

- What are the keys?
- What are the values?
- Which keys “map” to which values?



*Maps ints to ints*



# Mapping and Lists

```
numbers = [12, 49, -2, 26, 5, 17, -6]
```

```
# Using the key 3 to lookup the associated value of 26
```

```
# and then save the value into variable
```

```
new = numbers[3]
```

```
# Modifying the list so that the key 5 now maps to 77
```

```
# instead of 17
```

```
numbers[5] = 77
```

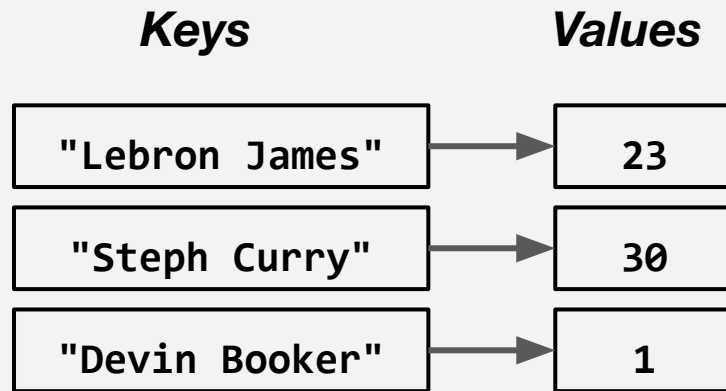
# Dictionary

- Associates a set of keys to their corresponding values *(like lists)*
- Each key has exactly 1 associated value *(like lists)*
- The keys can be types other than ints *(unlike lists)*

# Dictionary

## *Example*

```
players = { "Lebron James": 23,  
            "Steph Curry": 30,  
            "Devin Booker": 1 }
```



*Map strings to ints*

# Dictionary

```
players = { "Lebron James":23,  
            "Steph Curry":30 ,  
            "Devin Booker":1  }
```

*# Using the key "Lebron James"*

*# to Lookup the number 23*

```
number = players["Lebron James"]
```

*# Modifying the number associated with*

*# "Devin Booker" from 1 to 12*

```
numbers["Devin Booker"] = 12
```

# Dictionary Questions

What will the keys and values be after this code runs?

```
word_count = {"and":324, "why":134, "cannot":76, "sanded":13}  
word_count["cannot"] = 90  
word_count["and"] = 110  
word_count["foot"] = "feet"  
word_count["and"] += 10
```

# Dictionary Questions

What will be in num\_to\_player at the end of this program?

```
num_to_player = {}    # A valid, but empty dictionary
num_to_player[13] = "Paul George"
num_to_player[3]  = "Chris Paul"
num_to_player[23] = "Lebron James"
num_to_player[13] = "James Harden"
```

# Dictionary Operations

- Basic dictionary operations are:
  - **Add** a new mapping from a key to a value
  - **Modify** what value an existing key maps to
  - **Retrieve** a value using a key
  - **Remove** a key/value association
- Some of these look similar to lists

# Dictionary Operations

```
scores = {'A': 10, 'B':25, 'C':27, 'D':10, 'E':5}
```

```
scores['A+'] = 7      # Adds a key/value pair
```

```
scores['B'] = 20      # Changes value associated with a key
```

```
c_scores = scores['C']    # Retrieves a value, given a key
```

```
del scores['E']          # Removes a key/value pair
```



# What will print?

- `del dictionary[key]`
  - Deletes the key (and the value it is paired with) from the dictionary

```
word_count = {"and":324, "why":134,  
              "cannot":76, "Sanded":13}
```

```
del word_count["why"]
```

```
del word_count["nothing"]
```

```
del word_count["and"]
```

```
print(word_count)
```

# Check for key

- **if key in dictionary:**
  - True condition if key exists in dictionary, false otherwise
- **if key **not** in dictionary:**
  - The opposite
- Important to check for existence of keys if you don't want to get an error!

# What will print? What does it accomplish?

```
ds1 = {}  
word = 'statistic'  
for char in word:  
    if char not in ds1:  
        ds1[char] = 0  
    ds1[char] += 1  
print(ds1)
```

# What will print? What does it accomplish?

```
ds2 = {}  
grades = [70, 55, 91, 95, 80]  
for grade in grades:  
    key = int( (grade - 50) / 10 )  
    if key not in ds2:  
        ds2[key] = 0  
    ds2[key] += 1  
print(ds2)
```

What will it print?

```
word_counts = {"Beluga" : 1, "Bar" : 15,  
               "Bend" : 3, "Behave" : 8}
```

```
words = ["End", "Have", "Bar"]
```

```
for word in words:
```

```
    if word in word_counts:
```

```
        print(word, "has a count")
```