CSE231 - Lab 09

COVID-19, Nested Dictionaries, Sets

COVID-19

As I’m sure you’re all aware, MSU administration is extremely on-their-toes about the recent coronavirus disease. In the event of campus shutdown, we will, (probably like most of your other classes), move to an online format.

In the case this happens, labs (for you guys) will be due on Friday, on Mimir, at 11:59 PM. They will be graded on ***perfection.*** Meaning that you’ll have to pass the test cases, alike Lab 08.

Everything else will pretty much stay the same, since most of this class takes place online anyways.

COVID-19 (cont.)

If you are afraid to come to lab at all, contact Dr. Enbody *and* Dr. Zaabar (contact details on the course website/GitHub) so they can accommodate you.

If you are sick, please for all that is holy, do ***not*** come to lab. Contact me beforehand so I can adjust your lab grade accordingly.

There is an option to take Exam 2 remotely. Contact Dr. Enbody *and* Dr. Zaabar by the end of today if you wish to do this. There is a cost and some arrangements that have to be made if you choose to do this, I was never told what they were 🤷‍♀️

Nesting Dictionaries

Like lists and tuples, dictionaries can be nested. You can have a dictionary whose values are *other* dictionaries. Remember that keys in a dictionary have to be immutable, thus keys *cannot* be dictionaries.

Having a nested dictionary can be a bit confusing so hopefully an example might help. We’re not going to spend much time on this since it should all be review, might come in handy for the lab today.

L9-1

Sets

The last Python data structure! We've finally made it. Sets are an idea taken from mathematics, most of you should know them, but if not, we'll briefly go over the essentials.

Sets are a collection of *unique*, *unordered* objects. Meaning that two sets like the following are mathematically ***equivalent***.

$A = \{1, 2, 3, 3, 4, 5\} = \{ x \mid 1 \leq x \leq 5, x \in \mathbb{Z} \}$

$B = \{5, 4, 4, 4, 3, 2, 1, 1\} = \{ x \mid 1 \leq x \leq 5, x \in \mathbb{Z} \}$

"$\{ x \mid 1 \leq x \leq 5, x \in \mathbb{Z} \}$" can be read as: "x such that x is inclusively bounded between 1 and 5, and is within the set of all integers."

Sets (cont.)

How do we translate this to Python? In much the same way, actually.

```
A = {1, 2, 3, 3, 4, 5}

B = {5, 4, 4, 4, 3, 2, 1, 1}

print(A == B)    # True

print(A)    # {1, 2, 3, 4, 5}

print(B)    # {1, 2, 3, 4, 5}

print(len(A))    # 5
```

Set Initialization

```
D = {}    # curly braces with nothing = empty dictionary

S = set()    # correct empty set initialization

S = {20, 5, 10}    # initialization of a set with values

S = set(“abcabbcd”)    # can convert from iterables

print(S)    # {‘b’, ‘d’, ‘a’, ‘c’}

S = {x for x in “abracadabra” if x not in “abc”}

print(S)    # {‘r’, ‘d’}, set comprehension!
```

Adding/Discarding

```
S = set()

S.add(100)    # S = {100, }, adds a given element

S.discard(100)    # S = set(), discards a given element

S.remove(100)    # KeyError

# .remove() removes the object from the set, but raises

# KeyError if the object doesn't exist
```

Set Operations

Like in mathematics, sets have a lot of unique operations. If you already know about mathematical sets, you may recognize these guys: ⊆, ⊂, ∪, ∩, etc.. All of these operations have a Python equivalent.

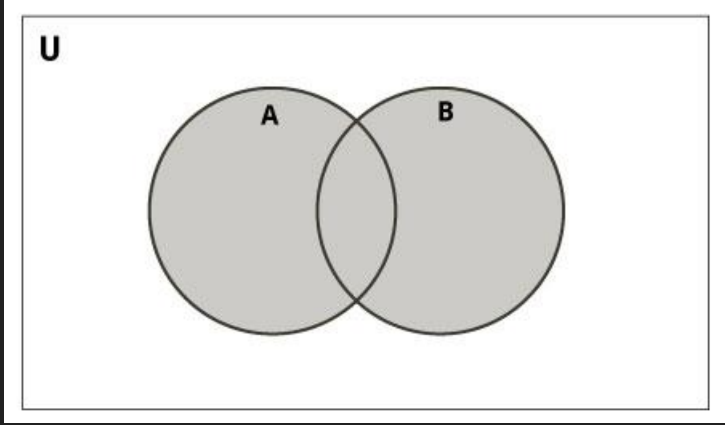
In addition to this, you can of course use `len()` to determine the number of elements within the set (| | in mathematics, also known as the “cardinality” when talking about sets), and use the `in` keyword to determine if an element exists within your set (∈ in mathematics).

For each of the operators in Python, there is also a corresponding method function as you’ll see.

Union

```
A = {1, 2}

B = {3, 4}

print(A | B)    # {1, 2, 3, 4}

print(A.union(B))    # {1, 2, 3, 4}
```

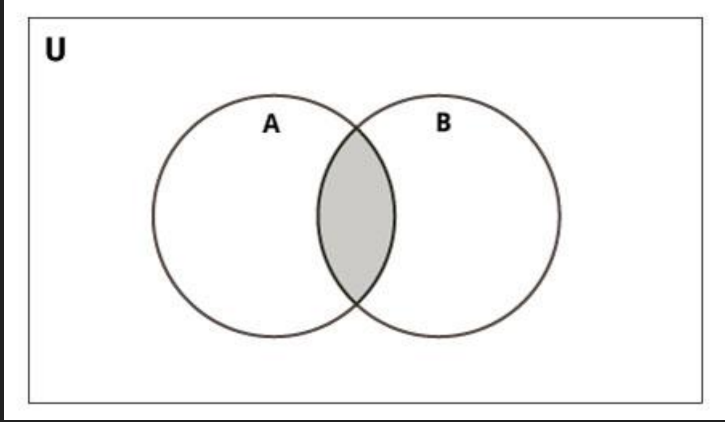
All elements, within both sets. Equivalent to ∪ in mathematics.



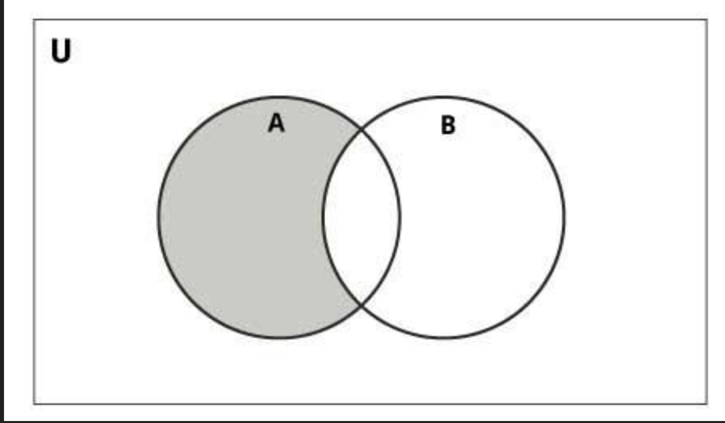
Intersection

```
A = {1, 2, 3}

B = {2, 3, 4}

print(A & B)    # {2, 3}

print(A.intersection(B))    # {2, 3}
```

All elements ***shared*** between both sets. Equivalent to ∩ in mathematics.



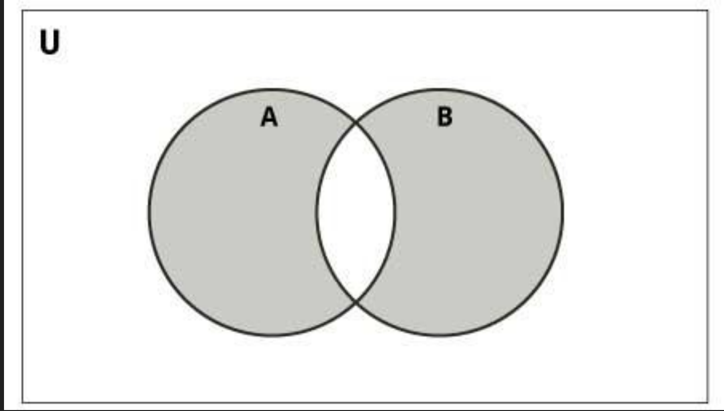
Difference



```
A = {1, 2, 3}

B = {2, 3, 4}

print(A - B)    # {1}

print(A.difference(B))    # {1}
```

Elements ***in A but not in B***. Equivalent to − in mathematics.

Symmetric Difference



```
A = {1, 2, 3}

B = {2, 3, 4}

print(A ^ B)    # {1, 4}

print(A.symmetric_difference(B))    # {1, 4}
```

All elements ***unique*** to both sets. Equivalent to ⊖ in mathematics.

Today’s Lab: tinyurl.com/vnv3mmq

Today’s lab is a hot mess.

Dr. Enbody made some whoopsies on the original starter code. The code was also very badly formatted in my opinion, so use my stuff instead. I cleaned and reworked some of the code to make it clearer as to what you’re supposed to do.

The PDF of today’s lab also says to rename your part a file. DON’T DO THIS. Mimir wants BOTH parts submitted. Make a copy of your part a file, and then rename it to be part b. Part b works off of the assumption that you did part a.

Have fuuuuuuuuuuuunnnnnnn