Zum warm werden

- Wodurch zeichnen sich Mengen aus?

```
set_0 = {}
print(set_0}          # Ausgabe?
```

- Welche Möglichkeiten gibt es Sets zu verändern?
  - Elemente hinzufügen
  - Elemente entfernen

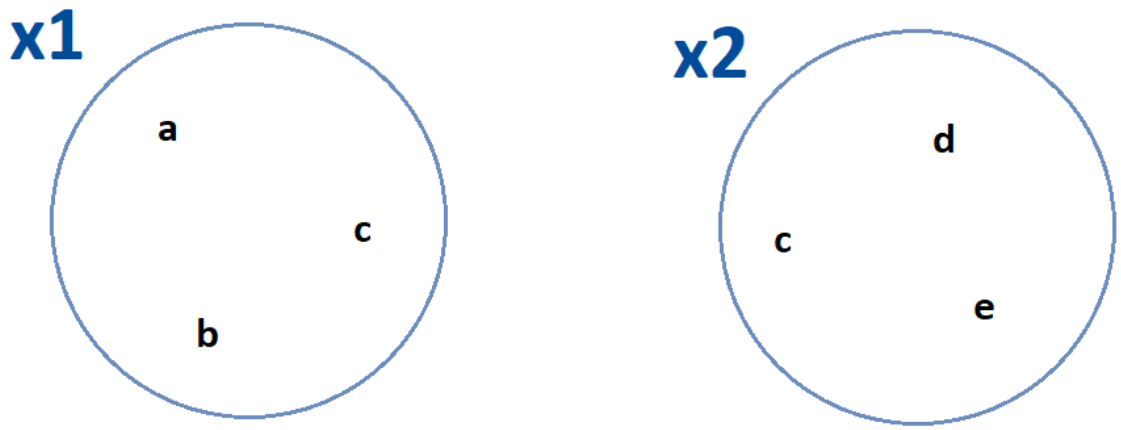
- 13 – Hausi.py

Sets können noch mehr!

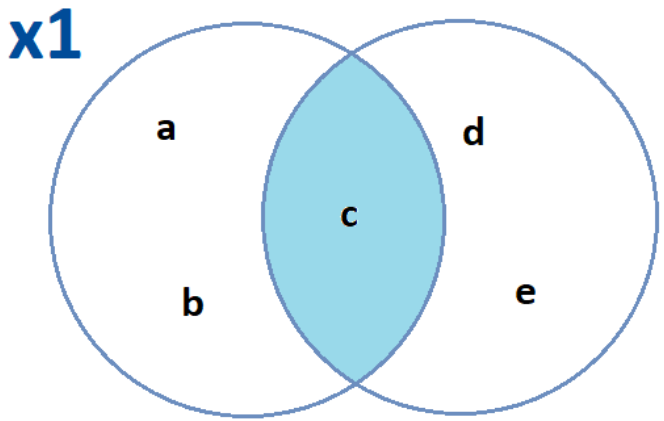
- Mengenoperationen:
  - Intersection (Schnittmenge)
  - Difference (Differenz)
  - Symmetric Difference (Symmetrische Differenz)
  - Union (Vereinigungsmenge)

Intersection (Schnittmenge)

```
x1 = {"a", "b", "c"}
x2 = {"c", "d", "e"}

print(x1.intersection(x2))

# output:
# {"c"}

# alternative Schreibweise:
print(x1 & x2)
```

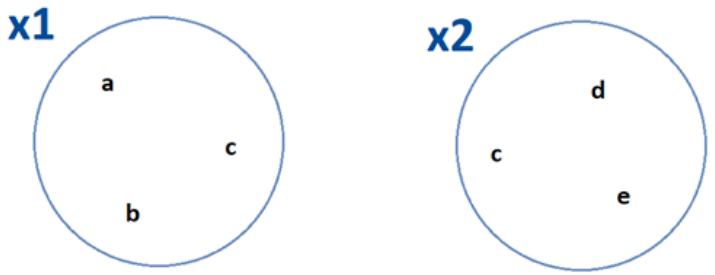



```
>> x1.intersection(x2)
```

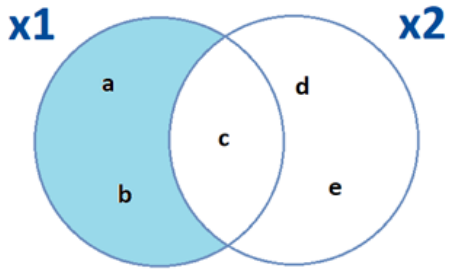


Symmetric Difference (symmetrische Differenz)

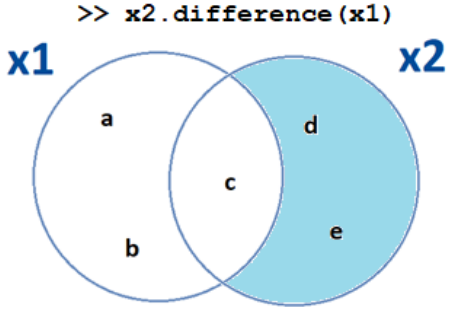
```
x1 = {"a", "b", "c"}
x2 = {"c", "d", "e"}

print(x1.difference(x2))
# output: {"a", "b"}
# alternativ:
print(x1 - x2)

print(x2.difference(x1))
# output: {"d", "e"}
# alternativ:
print(x2 - x1)

print(x1.symmetric_difference(x2))
# output: {"a", "b", "d", "e"}

print(x2.symmetric_difference(x1))
# output: {"a", "b", "d", "e"}

# alternativ:
print(x1 ^ x2)
```



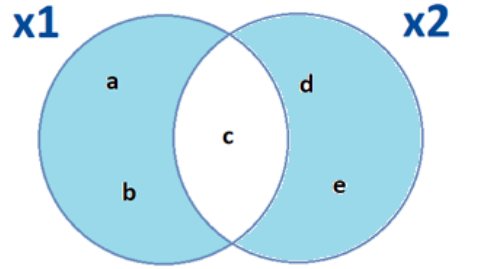

```
>> x1.difference(x2)
```




```
>> x2.difference(x1)
```

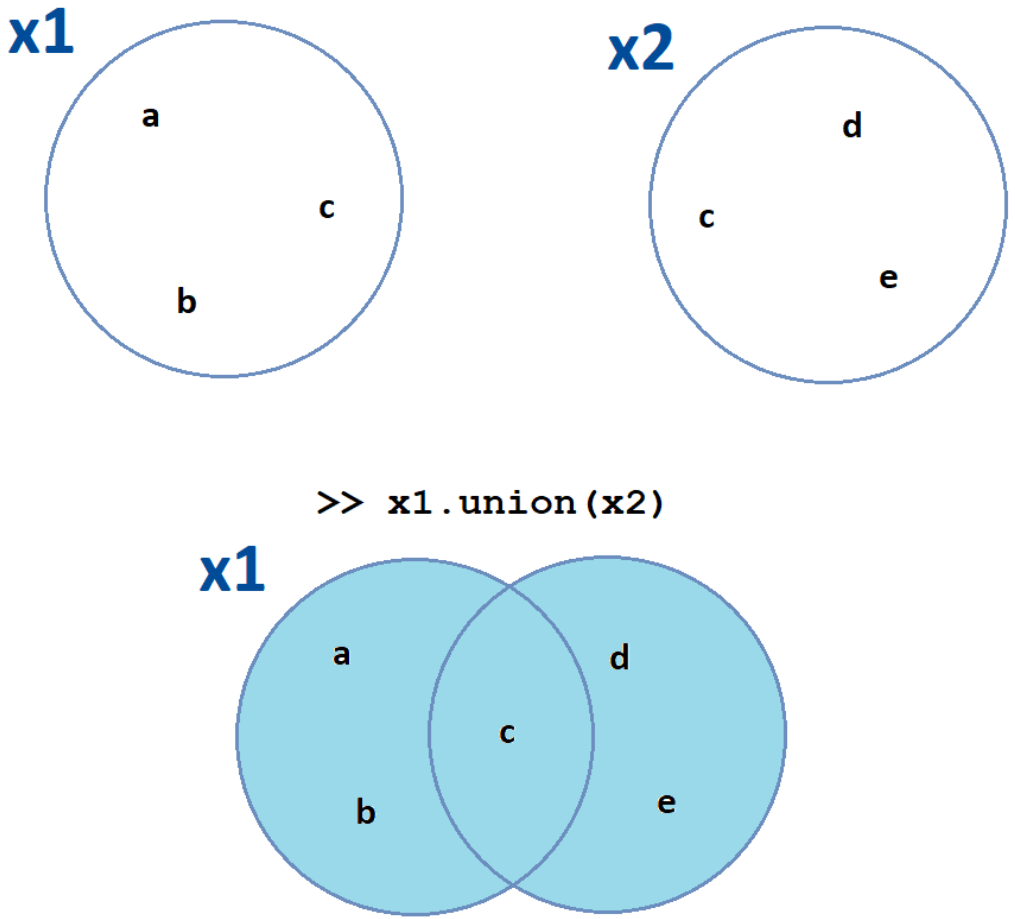



```
>> x1.symmetric_difference(x2)
>> x2.symmetric_difference(x1)
```



Union (Vereinigungsmenge)

```
x1 = {"a", "b", "c"}
x2 = {"c", "d", "e"}

print(x1.union(x2))
# output:
# {"a", "b", "c", "d", "e"}

# alternative Schreibweise:
print(x1 | x2)
```



Ein paar Beispiele

- Siehe 14 – Übung.py

Zusatz zu Mengenoperationen

- Zu jeder eben vorgestellten Methode, existiert auch eine _update()-Version:
  - z.B. .difference() -> .difference_update()
- Die _update()-Methoden verändern die ursprüngliche Menge:

```
my_set = {1,2,3}
print(my_set.difference({1,2}))          # Ausgabe: {3}
print(my_set)                            # Ausgabe: {1, 2, 3}

my_set.difference_update({1,2})
print(my_set)                            # Ausgabe: {3}

# _update()-Methoden haben kein return (bzw. return None)
print(my_set.difference_update({4, 5, 6}))    # Ausgabe: None
```

Und dann gibt es noch…

- 3 Checker-Methoden:

x.isdisjoint(y)

Gibt True zurück, wenn x und y keine Schnittmenge haben (=disjunkt sind).

x.issubset(y)

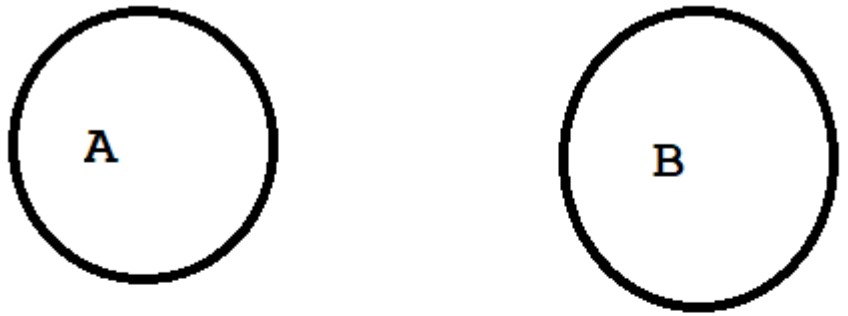
Gibt True zurück, wenn x eine Teilmenge von y ist (=x in y steckt).

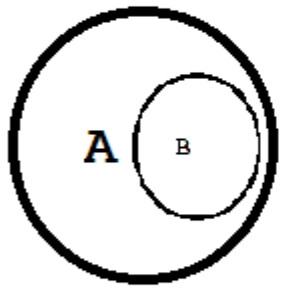
x.issuperset(y)

Gibt True zurück, wenn x die Obermenge von y ist (=y in x steckt).

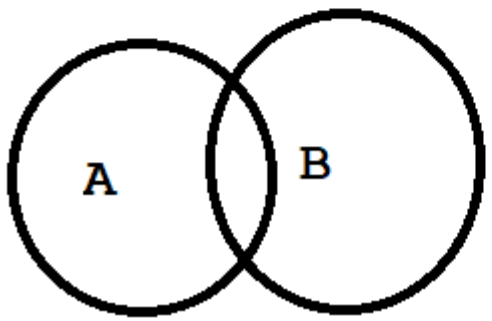
Beispiel

A.isdisjoint(B) -> True





B.issubset(A) -> True



A.isdisjoint(B) ->
False

Hausaufgaben

- Folien nochmal durchgehen
  - Mit eigenen Mengen die Mengenoperationen durchführen
- In 14 – Übung.py die Aufgabe 3 machen