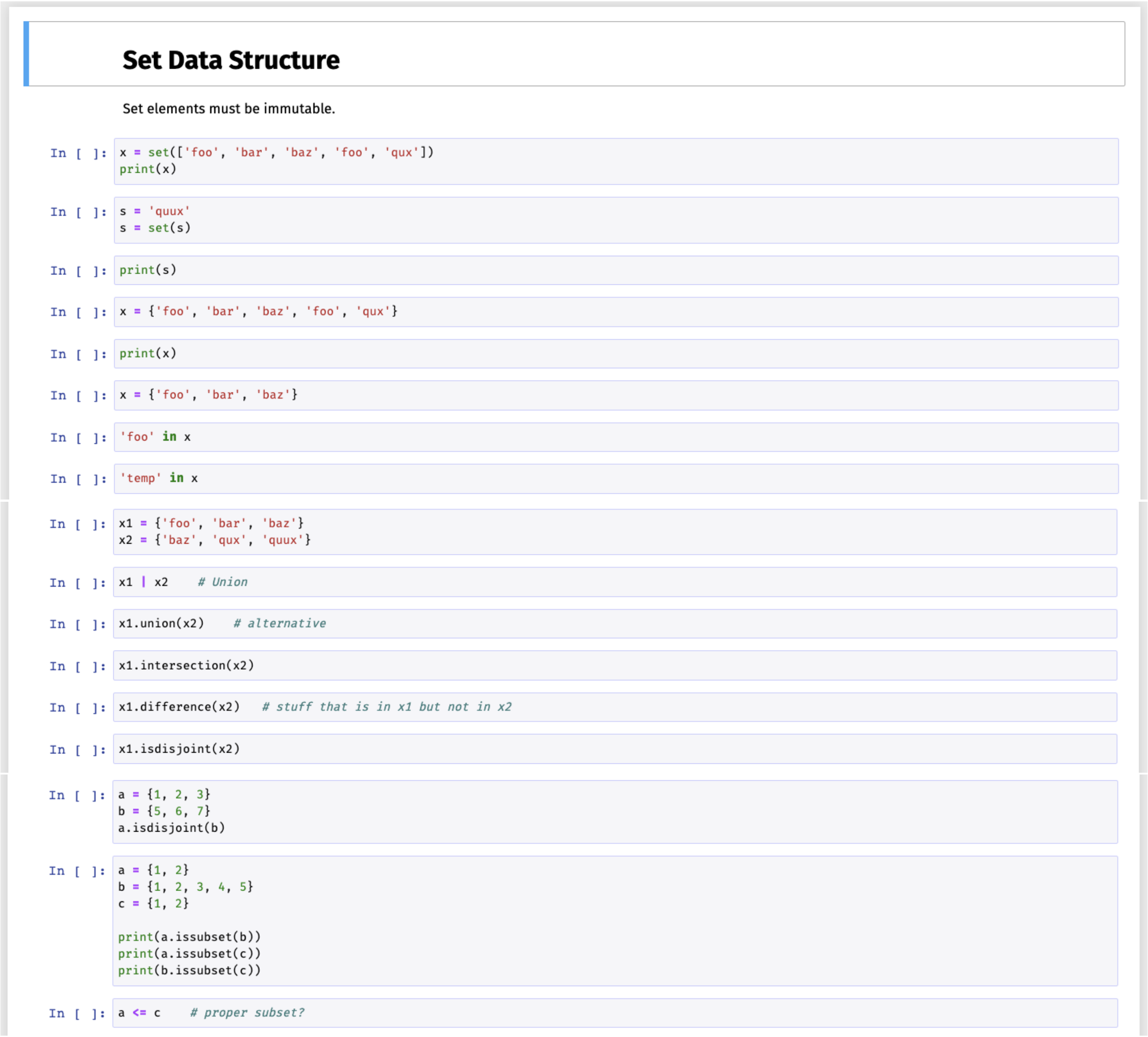
CS218 - Data Structures
FAST NUCES Peshawar Campus
Dr. Nauman (recluze.net)

September 15, 2019

1 Set Data Structure and Counter

Raster images of the notebook 08-set-counter



Set Data Structure

Set elements must be immutable.

```
In [ ]: x = set(['foo', 'bar', 'baz', 'foo', 'qux'])
        print(x)
```

```
In [ ]: s = 'quux'
        s = set(s)
```

```
In [ ]: print(s)
```

```
In [ ]: x = {'foo', 'bar', 'baz', 'foo', 'qux'}
```

```
In [ ]: print(x)
```

```
In [ ]: x = {'foo', 'bar', 'baz'}
```

```
In [ ]: 'foo' in x
```

```
In [ ]: 'temp' in x
```

```
In [ ]: x1 = {'foo', 'bar', 'baz'}
        x2 = {'baz', 'qux', 'quux'}
```

```
In [ ]: x1 | x2    # Union
```

```
In [ ]: x1.union(x2)    # alternative
```

```
In [ ]: x1.intersection(x2)
```

```
In [ ]: x1.difference(x2)   # stuff that is in x1 but not in x2
```

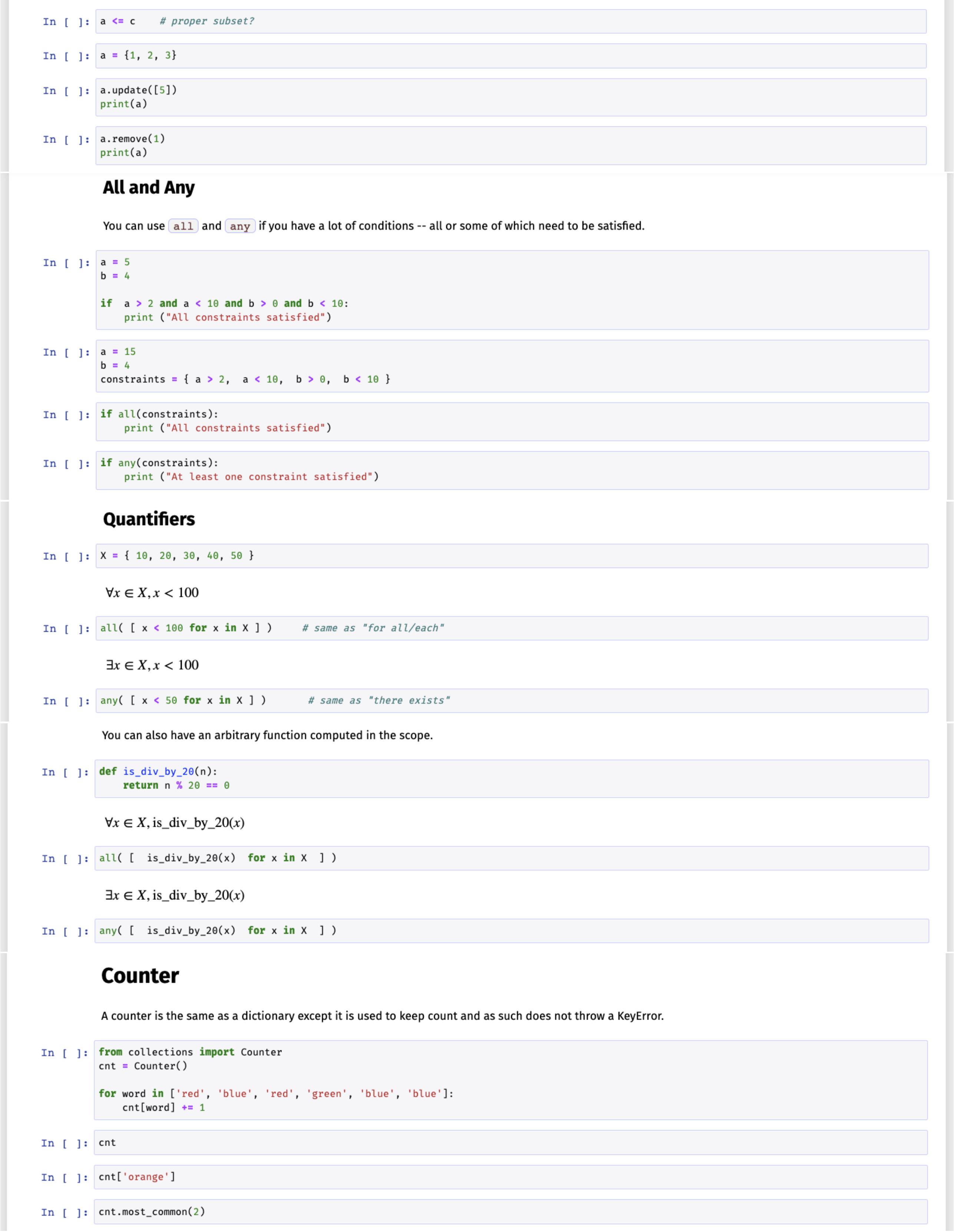
```
In [ ]: x1.isdisjoint(x2)
```

```
In [ ]: a = {1, 2, 3}
        b = {5, 6, 7}
        a.isdisjoint(b)
```

```
In [ ]: a = {1, 2}
        b = {1, 2, 3, 4, 5}
        c = {1, 2}

        print(a.issubset(b))
        print(a.issubset(c))
        print(b.issubset(c))
```

```
In [ ]: a <= c    # proper subset?
```



```
In [ ]: a <= c    # proper subset?
```

```
In [ ]: a = {1, 2, 3}
```

```
In [ ]: a.update([5])
        print(a)
```

```
In [ ]: a.remove(1)
        print(a)
```

All and Any

You can use `all` and `any` if you have a lot of conditions -- all or some of which need to be satisfied.

```
In [ ]: a = 5
        b = 4

        if  a > 2 and a < 10 and b > 0 and b < 10:
            print ("All constraints satisfied")
```

```
In [ ]: a = 15
        b = 4
        constraints = { a > 2,  a < 10,  b > 0,  b < 10 }
```

```
In [ ]: if all(constraints):
            print ("All constraints satisfied")
```

```
In [ ]: if any(constraints):
            print ("At least one constraint satisfied")
```

Quantifiers

```
In [ ]: X = { 10, 20, 30, 40, 50 }
```

$\forall x \in X, x < 100$

```
In [ ]: all( [ x < 100 for x in X ] )     # same as "for all/each"
```

$\exists x \in X, x < 100$

```
In [ ]: any( [ x < 50 for x in X ] )      # same as "there exists"
```

You can also have an arbitrary function computed in the scope.

```
In [ ]: def is_div_by_20(n):
            return n % 20 == 0
```

$\forall x \in X, \text{is\_div\_by\_20}(x)$

```
In [ ]: all( [  is_div_by_20(x)  for x in X  ] )
```

$\exists x \in X, \text{is\_div\_by\_20}(x)$

```
In [ ]: any( [  is_div_by_20(x)  for x in X  ] )
```

Counter

A counter is the same as a dictionary except it is used to keep count and as such does not throw a KeyError.

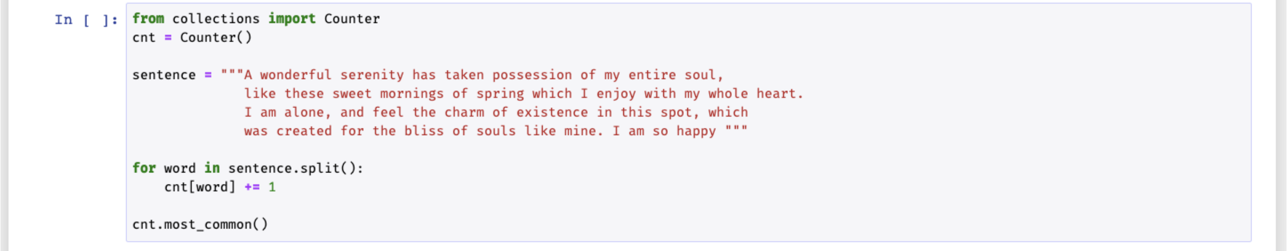
```
In [ ]: from collections import Counter
        cnt = Counter()

        for word in ['red', 'blue', 'red', 'green', 'blue', 'blue']:
            cnt[word] += 1
```

```
In [ ]: cnt
```

```
In [ ]: cnt['orange']
```

```
In [ ]: cnt.most_common(2)
```



```
In [ ]: from collections import Counter
        cnt = Counter()

        sentence = """A wonderful serenity has taken possession of my entire soul,
                      like these sweet mornings of spring which I enjoy with my whole heart.
                      I am alone, and feel the charm of existence in this spot, which
                      was created for the bliss of souls like mine. I am so happy """

        for word in sentence.split():
            cnt[word] += 1

        cnt.most_common()
```