Coding Challenge Notes

Stefan Webb
University of Oxford

1 Unorganized

- I appear to be much slower at implementing recursive solutions! It really helps to draw a picture of the recursion stack here.
- A general problem: generate all subsets of a string/list.

2 Useful functions

- **`reversed`**`(s)` reverses an iterable and returns an iterator.

3 Converting types

3.1 How do I convert a list of chars/strings to string?

```
>>> string = ['a', 'b', 'c']
>>> "".join(strings)
```

3.2 How do I convert a number to a string/vice versa?

```
>>> a = str(6.24)
>>> b = int("8")
>>> c = float("1.25")
```

4 Dictionary manipulation

4.1 How do I get the keys/values of a dictionary?

```
>>> x = {'a': 1, 'b': 2, 'c': 3}
>>> x.keys(), x.values()
```

Get the values with, e.g., `x.values()` *not* `x.vals()`.

5 List manipulation

5.1 How do I get the max/min element of a list?

```
>>> x = [5,7,12,1,2]
>>> min(x), max(x)
```

5.2 How do I filter a list?

```
>>> [x for x in arr if x < 5]
```

The **`if`** *must come after the* **`for`** *in the comprehension, unlike when you do an* **`if/else`** *clause.*

5.3 How do I remove duplicates?

```
>>> x = [1, 3, 7, 9, 1]
>>> x = list(set(x))
```

This will change the order.

6 String manipulation

6.1 How do I convert from a character to its ASCII/vice versa?

```
>>> n = ord('a')
>>> a = chr(n)
```

6.2 How to I convert a character/string to lower/upper case?

```
>>> 'Abc'.lower()
>>> 'Abc'.upper()
```

7 Discriminating between different types of characters

7.1 How do I determine if a char is punctuation?

```
>>> import string
>>> if c is in string.punctuation:
>>>   ...
```

7.2 How do I determine if a char is whitespace?

```
>>> if c.isspace():
>>>   ...
```

7.3 How do I determine if a char is alphabetical/alphanumeric?

```
>>> if c.isalpha():
>>>   ...
>>> if c.isalnum():
>>>   ...
```

7.4 How do I determine if upper or lower case?

```
>>> if c.isupper():
>>>   ...
>>> if c.islower():
>>>   ...
```

8 Common gotchas

- **range**(n) goes from 0 to $n-1$, *not* up to n.
- There is neither ++ nor -- operators in Python. Use += 1 and -= 1 instead.
- The not operator is **not**, not !
- To increment a dictionary value that may not have a corresponding key:

```
>>> x.setdefault('x', 0) += 1
>>> x['x'] += 1
```

- You cannot overwrite the value of a **for** loop inside the loop body. Instead you should use **while** for this idiom:

```
    >>> i = 0
    >>> while i < 10:
    >>>   ...
    >>>   if cond:
    >>>     i += 5
    >>>   i += 1
```