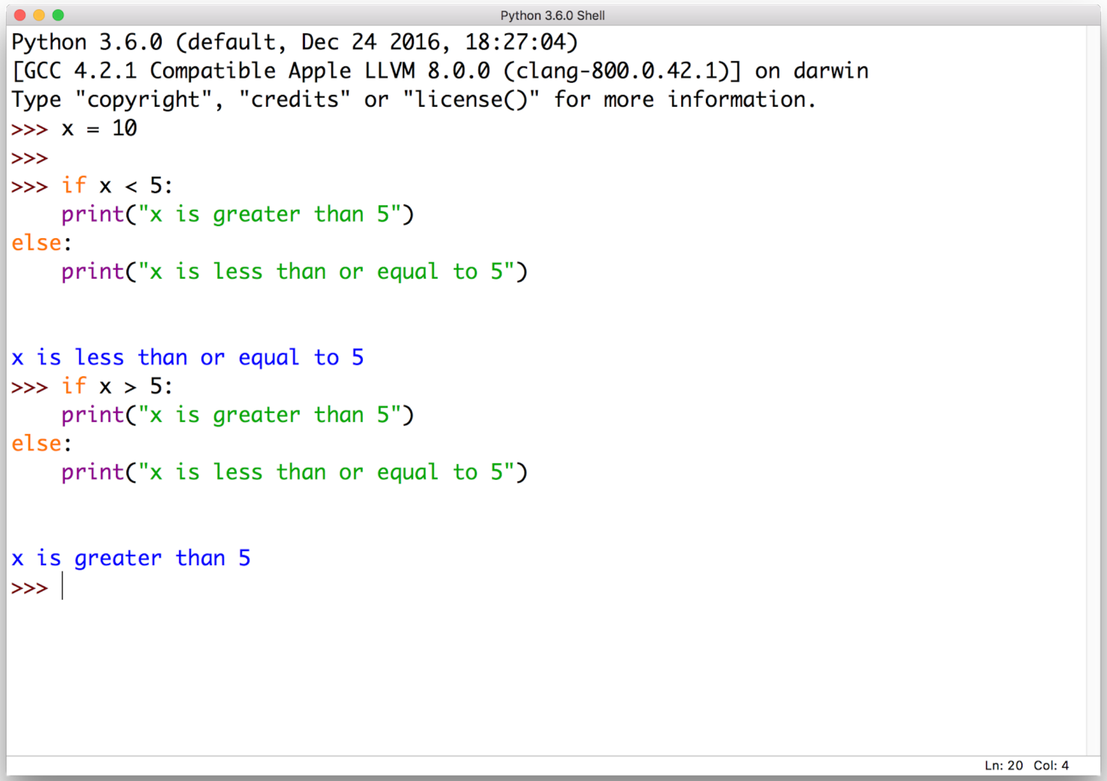
#CodeYork

# Solutions 1B

Summer 2017

## Task 1 - If Statements

This task introduces if statements. The intentional typo was the equality in the guard was incorrect (should be x greater than 5).

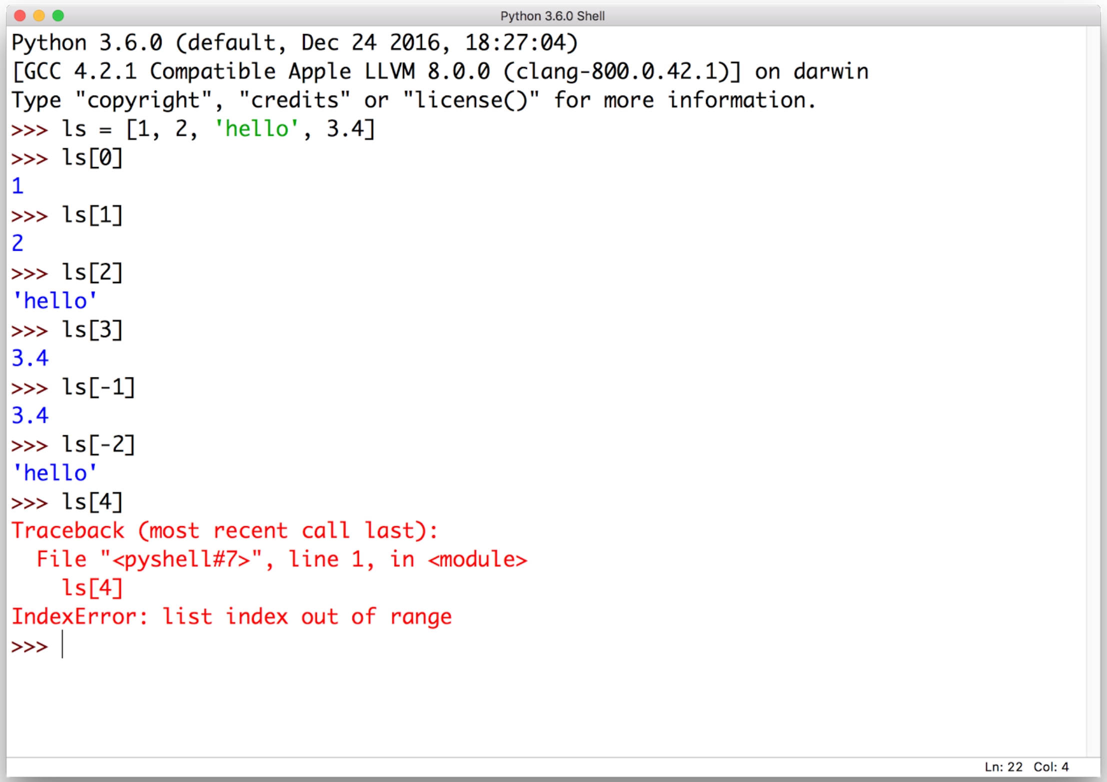


```
Python 3.6.0 (default, Dec 24 2016, 18:27:04)
[GCC 4.2.1 Compatible Apple LLVM 8.0.0 (clang-800.0.42.1)] on darwin
Type "copyright", "credits" or "license()" for more information.
>>> x = 10
>>>
>>> if x < 5:
    print("x is greater than 5")
else:
    print("x is less than or equal to 5")


x is less than or equal to 5
>>> if x > 5:
    print("x is greater than 5")
else:
    print("x is less than or equal to 5")


x is greater than 5
>>>
```

## Task 2 - List Access

Here we're looking at accessing elements from a list. We note that negative indexes can be used in order to query the list from the RHS rather than the LHS.

We also observe that Python will raise an *IndexError* when an index too large i used, although Exceptions themselves are not something formally covered by this course.

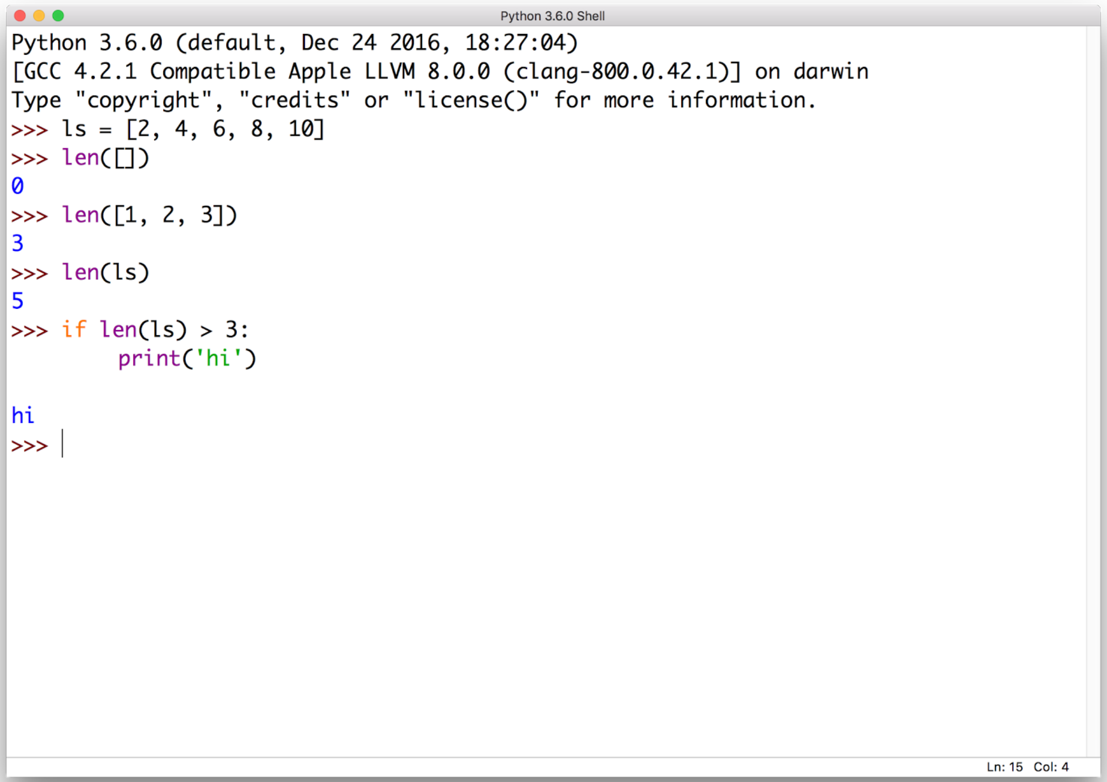


```
Python 3.6.0 (default, Dec 24 2016, 18:27:04)
[GCC 4.2.1 Compatible Apple LLVM 8.0.0 (clang-800.0.42.1)] on darwin
Type "copyright", "credits" or "license()" for more information.
>>> ls = [1, 2, 'hello', 3.4]
>>> ls[0]
1
>>> ls[1]
2
>>> ls[2]
'hello'
>>> ls[3]
3.4
>>> ls[-1]
3.4
>>> ls[-2]
'hello'
>>> ls[4]
Traceback (most recent call last):
  File "<pyshell#7>", line 1, in <module>
    ls[4]
IndexError: list index out of range
>>> 
```

## Task 3 - Length of List

This task starts by ensuring that both the len function is understood, but also again re-enforces the idea of variables, and passing them to functions. We more formally look at functions in the next session.

Next a short code snippet is provided, where the guard needs writing, and the body of the if statement needs writing. The guard is a simple length test, and the body is a print statement. Note that when we refer to the guard of an if statement, we're referring to the expression that it evaluated in order to determine its truthiness.



```
Python 3.6.0 (default, Dec 24 2016, 18:27:04)
[GCC 4.2.1 Compatible Apple LLVM 8.0.0 (clang-800.0.42.1)] on darwin
Type "copyright", "credits" or "license()" for more information.
>>> ls = [2, 4, 6, 8, 10]
>>> len([])
0
>>> len([1, 2, 3])
3
>>> len(ls)
5
>>> if len(ls) > 3:
        print('hi')

hi
>>> 
```

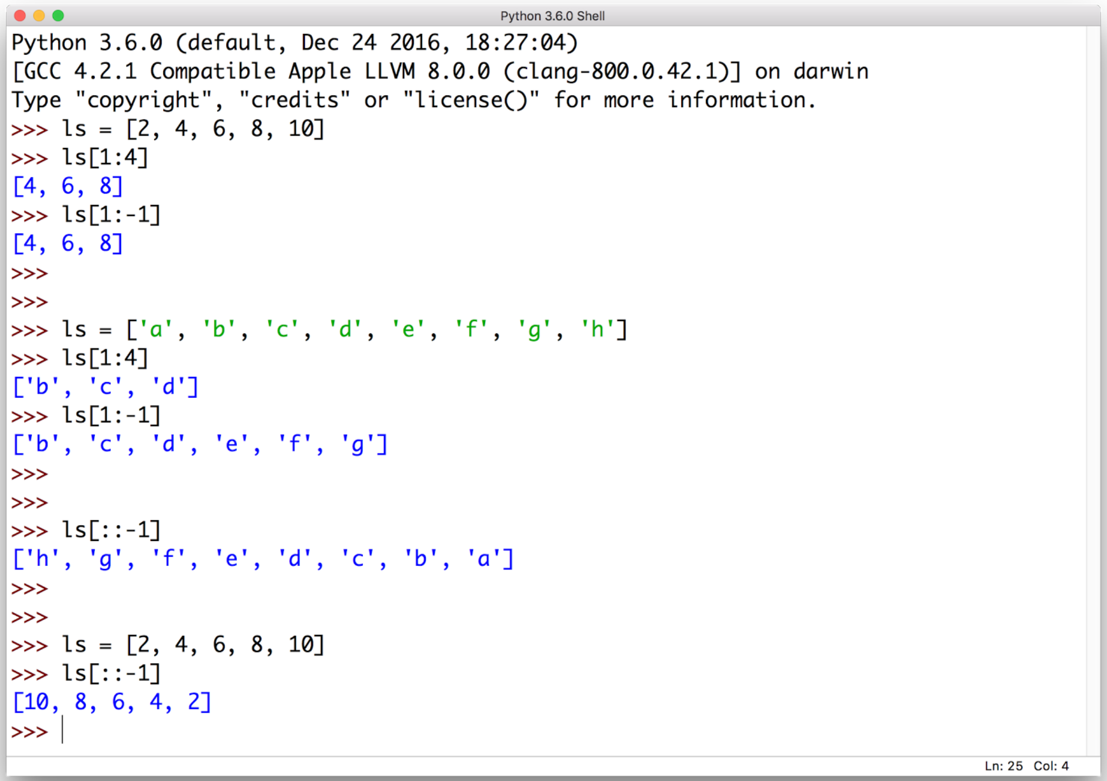
## Task 4 - Splicing a List

Finally, we're now looking at some basic list splicing operations supported by Python.

We start by looking at taking a sublist by specify the start and end index. Note in particular how the start index is inclusive, but the end index is exclusive. We can use a negative index in order to count from the RHS, like in task 2.

We observe that the negative index is particularly useful in the case where we don't know the length of the input list. This is the most general case.

We finish by demonstrating the easy ability to reverse a list (or indeed a string) in Python.



```
Python 3.6.0 (default, Dec 24 2016, 18:27:04)
[GCC 4.2.1 Compatible Apple LLVM 8.0.0 (clang-800.0.42.1)] on darwin
Type "copyright", "credits" or "license()" for more information.
>>> ls = [2, 4, 6, 8, 10]
>>> ls[1:4]
[4, 6, 8]
>>> ls[1:-1]
[4, 6, 8]
>>>
>>>
>>> ls = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h']
>>> ls[1:4]
['b', 'c', 'd']
>>> ls[1:-1]
['b', 'c', 'd', 'e', 'f', 'g']
>>>
>>>
>>> ls[::-1]
['h', 'g', 'f', 'e', 'd', 'c', 'b', 'a']
>>>
>>>
>>> ls = [2, 4, 6, 8, 10]
>>> ls[::-1]
[10, 8, 6, 4, 2]
>>>
```