Week 2 Quiz

Name - UNI

Instructions

Replace all '____' below using the instructions provided.

When completed,

- make sure you've replaced Name and UNI in the first cell and filename (eg: Week_2_Quiz-brg2130)
- Kernel->Restart & Run All to run all cells in order
- use Print Preview, Print-> Save to pdf
- and post pdf to GradeScope

---

1. Lists

In [1]:
```
# Create a list containing the strings 'blue','red','green'
colors = ['blue', 'red', 'green']

# Assert that value at index 0 of colors is equal to 'blue'
assert colors[0] == 'blue'

# Using list indexing, print out the value of colors at index 1
# You should see the output "red" without quotes
print(colors[1])
```

```
red
```

2. Dicts

In [2]:
```
# Create a dictionary which maps the string keys 'zero','one','two'
#    to the int values 0,1,2
str_to_int = {'zero':0, 'one':1, 'two':2}

# Assert that key 'two' equals 2 in str_to_int
assert str_to_int['two'] == 2

# Using str_to_int, print out the value for the key 'one'
# You should see the output 1
print(str_to_int['one'])
```

```
1
```

3. String Formatting And For Loops

In [3]:
```
# Using the len function and f"" string formatting, print the number of eleme
print(f"the length of colors is {len(colors)}")

# Using the enumerate function, the colors list defined above, and f"" string
#   for every index, value pair,
#   print "the value at index {index} is {value}"
for index, value in str_to_int.items():
    print(f"the value at index {index} is {value}")
```

```
the length of colors is 3
the value at index zero is 0
the value at index one is 1
the value at index two is 2
```

4. List Comprehension

In [4]:
```
# Using a list comprehension and the len() function,
#    create a list of the lengths of the strings in colors
# Store the resulting list in variable color_lengths
color_lengths = list(len(x) for x in colors)

# Assert that the first value in color_lengths is 4 (the length of 'blue')
assert color_lengths[0] == 4
```

5. Functions and Control Flow

In [5]:
```
# Define a function called append_x
# It should take in a string
#    and return that string with '_x' appended to the end of it.
# For example: 'blue' should become 'blue_x'
# You don't need to use Type Annotations, but feel free to
def append_x(str):
    return(f'{str}_x')


assert append_x('test') == 'test_x'
```

6. Sorting

In [6]:
```
# Using sorted(), sort the list color_lengths created above, ascending in val
# Save as color_lengths_sorted
color_lengths_sorted = sorted(color_lengths, reverse=False)

# Assert that the first element of color_lengths_sorted is 3
assert color_lengths_sorted[0] == 3
```

In [7]:
```
# Bonus:
# Create a list of the key,value pairs in the str_to_int dictionary sorted by
str_to_int_sorted = sorted(str_to_int.items(), key=lambda x: x[1], reverse=Tr
str_to_int_sorted
# Use this assert to check your work but removing the comment
assert str_to_int_sorted[0] == ('two',2)
```

In [ ]:
```

```