You will find a private GitHub repo called `<github-username>-hw` where you will submit all your homework assignments. Clone this repo and create a `hw1` directory inside. Add this directory to the repo using `$ git add hw1`. All your code should appear in a file called `hw1.py` that lives inside the `hw1` directory. Make sure to add `hw1.py` to the repo and commit your changes with `$ git commit -a -m "good log message"`.

**Question 1 (5 points).** Let `l = list('diving into the deluge of data')`. Without using the python interpreter, but with the use of documentation, what does `.join(l)` equal after the following operations? Verify your answer on the computer. Were you right? Give your guess and whether you were right in a comment (i.e., a line starting with `#`) in `hw1.py`.

```
>>> l.remove('i')
>>> del l[1]
>>> del l[4:9]
>>> l.reverse()
>>> del l[:8]
>>> l.reverse()
>>> l.pop()
>>> l.append('a')
>>> l[-6] = 'b'
```

**Question 2 (10 points).** A run-length encoding of a string compresses runs of consecutive identical characters into a pair `(x, y)` where `x` is the character and `y` is the count. For example, a run-length encoding of the string `"aaabbccccdddddabbb"` is the list

```
[("a", 3), ("b", 2), ("c", 4), ("d", 6), ("a", 1), ("b", 3)]
```

(a) Define a function `run_length_encode(s)` that takes a string and produces a run-length encoded representation (i.e., a list of 2-tuples that appropriately encodes `s`).

(b) Define a function `run_length_decode(l)` that takes a run-length encoded list and returns the appropriately decoded string. You may find the following example for loop syntax useful. Let `lst = [("a", 3), ("b", 2), ("c", 6)]`. Consider the following loop.

```
>>> for (x,y) in lst:
...     print("{} {}.
```

Your code should contain an informative doc string and should be edited for clarity.

**Question 3 (15 points).** This question explores writing functions that other functions as arguments (i.e., higher-order functions). Note that Python supports defining functions inside the body of other functions, so the following is perfectly legal Python code.

```python
def mult_of_two_and_three(x):
    """returns True if and only if x is a multiple of 2 and 3""
    return mult_of_two(x) and mult_of_three(x)
```

```python
def mult_of_two(x):
    return x % 2 == 0

def mult_of_three(x):
    return x % 3 == 0
```

```python
return mult_of_two(x) and mult_of_three(x)
```
(a) Define a function called `count_even(lst)` that accepts an iterable of integers and returns the number of integers that are even. For example

```python
>>> count_even(range(10))
5
>>> count_even([0,0,2,2,3])
4
```

(b) Define a function called `count_pred(lst, pred)` that accepts an iterable and a predicate (i.e., a function that returns either True or False) and returns a count of the number of objects in `lst` where `pred` is True.

(c) Rewrite your function from (a) in terms of (b).