10F, q2: loop & conditional count

```python
sum = 0
for k in range(10):
    if k % 3 == 0:
        sum = sum + 1
print(sum)
```

10F q3: function call

```python
def abdiff(x, y):
    if x > y:
        x = x - y
    else:
        x = y - x
    return x
def q3():
x = 7
y = 4
x = abdiff(x, 5)
y = abdiff(y, 14)
y = y + x
print(y)
```

10F q4

```python
def avg_pos(ar, k):
    """Find the average (arithmetic mean) of the positive entries in the first k elements of an array.
    Arguments:
    ar: array (list) of numbers.
    with at least k elements.
    k: an integer: how many elements in ar to consider.
    Returns:
    Average of the positive numbers in the range ar[0]..ar[k-1], or -1.0 if there are no positive numbers in that range.
    Example: avg_pos([2.0, -2.0, 3.0, 0.0, 5.0], 4) returns 2.5. (2.0 + 3.0)/2; 0.0 and -2.0 are omitted as negative, and 5.0 is beyond k position k.
    """
```
10F, q6 (search)

def present(dict, word):
    """Determine whether string word is in dict.
    Arguments:
    dict: a list of strings, in sorted order
    (dict[i] > dict[i-1] for all
    i in range 1..len(dict)-1 )
    word: a string to search for
    Returns:
    True if word is in dict (else False)
    """

(up to 15 points for binary search,
10 points for linear search)

11F, q6

In the imaginary word game *Words with Pets*, a vowel is worth 2
points, and a consonant (any other letter) is worth 3 points. Write
a function to determine the value of a word, using the is_vowel
function to distinguish vowels from consonants.

def is_vowel(letter):
    return ( letter in ['a', 'e', 'i', 'o', 'u'] )

def score(word):
    """Calculate value of word.
    Arguments: word is a string containing only lower
    case letters ('a'-'z').
    Returns: score for word, 2 pts for each vowel and 3 for
    each other letter.
    """


10F, q6 (search)

def present(dict, word):
    """Determine whether string word is in dict.
    Arguments:
    dict: a list of strings, in sorted order
    (dict[i] > dict[i-1] for all
    i in range 1..len(dict)-1 )
    word: a string to search for
    Returns:
    True if word is in dict (else False)
    """

(up to 15 points for binary search,
10 points for linear search)

11F, q6

In the imaginary word game *Words with Pets*, a vowel is worth 2
points, and a consonant (any other letter) is worth 3 points. Write
a function to determine the value of a word, using the is_vowel
function to distinguish vowels from consonants.

def is_vowel(letter):
    return ( letter in ['a', 'e', 'i', 'o', 'u'] )

def score(word):
    """Calculate value of word.
    Arguments: word is a string containing only lower
    case letters ('a'-'z').
    Returns: score for word, 2 pts for each vowel and 3 for
    each other letter.
    """


longest run

A “run” is a sequence of consecutive, equal elements.
For example, “abcccdedffghccl” has a run of 3 c’s and 2 f’s
(and runs of 1 ‘a’, and 0 ‘z’).

def longest_run_length(str):
    """Return the length of the longest run in string str"


def longest_run_letter(str):
    """What character appears in the longest run. str has
    at least one character. In case of a tie, return the letter
    that appears in the first of the longest runs"


flattten

given a nested array like [[a, b], c, [d, e, [f, [ ]]]]
(which is really a tree)
return a flattened version like [a, b, c, d, e, f]
(all the nutritious leaves, none of the stringy bark)

def flatten(tree):
filter (function variables)

Write a function

```
filter(ar, f)
```

which returns a list of elements in ar for which f (el) is True

```
def pos(x):
    return x > 0
```

```
filter([0, -1, 1, 2, 3, -5], pos) returns [1, 2, 3]
```