Write your name at the bottom of each page before you begin. 1 point for each page.

1. [5 points] What does q1( ) print?

```python
def q1():
    x = 5
    y = 2
    while y < x :
        x += 1
        y += 2
    print(y - x)
```

0

2. [5 points] What does q2( ) print?

```python
def add_time( start, elapsed ):
    """There should be a good docstring here,
    but it’s an exam so I left it off.
    """
    hh = start // 100
    mm = start % 100
    el_hh = elapsed // 100
    el_mm = elapsed % 100
    mm = mm + el_mm
    if mm >= 60 :
        hh += 1
        mm = mm - 60
    hh = hh + el_hh
    if hh > 12 :
        hh = hh % 12
    return 100*hh + mm

def q2():
    t = add_time(1130, 245)
    print(t)
```

215
3. [5 points] What does q3( ) print?

```python
def q3():
    x = 4
    y = 7
    x = f(x, y)
    y = f(y, x)
    print(x, y)

def f(x, y):
    x = 2 * x
    y = 2 * y
    if y > x:
        return y - x
    return x - y
```

6 2

4. [5 points] What does function q4 return?

```python
def q4():
    ev_count = 0
    for i in [2, 4, 6, 8, 10]:
        for j in [1, 2, 3, 4, 5]:
            sum = i + j
            if sum % 2 == 0:
                ev_count += 1
    return ev_count
```

10
5. [12 points] Twitter messages (“tweets”) usually contain “hashtag” words, which are key words or phrases beginning with the “hash” character ‘#’. Twitter documentation says:

```
Don’t #spam #with #hashtags. Don’t over-tag a single Tweet. (Best practices recommend using no more than 2 hashtags per Tweet.)
```

Complete this pair of Python functions to determine whether a tweet contains more than two hashtags. The first function counts the number of hash characters in a string. Using this, the second function classifies tweets.

```
HASH = "#"
def num_hash(str):
    """Count hash tags in a string.
    Args:
        str: A string (usually a tweet)
    Returns:
        The number of times HASH ("#") appears in str.
    """
    n = 0
    for ch in str:
        if ch == HASH:
            n += 1
    return n

def is_spam_hash(tweet):
    """Determines whether a tweet is "hash spam" containing three or more hash tags.
    Args:
        tweet: A string
    Return:
        True if tweet contains three or more hash tags.
    """
    if num_hash(tweet) > 2:
        return True
    return False
```

Note: Many people used the built-in "count" function of Python for num_hash. That’s also ok (although if I had remembered it existed, I would have told you to write your own).
6. [13 points] Write the body of function is_progression, consistent with its docstring header comment.

def is_progression(ar):
    """Determines whether ar is an arithmetic progression, that is, whether the differences between each element and its successor is the same.
    Args:
        ar: A list of zero or more integers
    Returns:
        True if the elements of ar are an arithmetic progression, otherwise False.
    Examples: Returns True for
        [1, 2, 3, 4, 5, 6] (steps of 1)
        [3, 7, 11, 15, 19] (steps of 4)
        [5, 1, -3, -7] (steps of -4)
        [42, 42, 42, 42] (steps of 0)
    And note in particular, returns True for
        [ ], [2], and [2, 19]
    because if there are fewer than 2 steps, they must be the same.
    Returns False for [5, 7, 10] because 7-5 != 10-7
    """
    if len(ar) < 3:
        return True
    step = ar[1] - ar[0]
    i = 2
    while i < len(ar):
        if (ar[i] - ar[i-1]) != step:
            return False
        i += 1
    return True

There are many correct ways to write this function (and even more incorrect ways). It is fairly challenging ... so far I am seeing far more missing, incomplete, or incorrect answers than correct answers.