Name:_____ CM: _____ Section: _____ Grade: _____ of 10

1. You are about to write a method that does some complex math. As usual, ***you start by working a concrete example by hand.*** What are some things you should do? Check all that apply:

   _____ Choose your numbers to be as complicated as possible.

   _____ Choose numbers that avoid symmetry.

   _____ Give names to the relevant items.

   _____ Track how you calculate the answer by hand.

   _____ Use this example as a unit test that you write first before writing your code.

2. The two code segments shown to the right both compute the same thing when given the same value for ***n***. Which code segment is better? (circle your choice)

```
if n > 5:
    return 888
if n <= 5:
    return 999
```

```
if n > 5:
    return 888
else:
    return 999
```

The one on the LEFT ***The one on the RIGHT*** ***Neither (they are equally good)***

3. What does the code snippet shown to the right print when it runs?

```
b = 0
for k in range(4):
    if (k + 3) % 3 >= 1:
        b = b + 1
    print(k, b)
print(b)
```

4. In the space to the right, write code for a function that has a single parameter whose value must be one of the following letter grades:
   "A" "B" "C"
   (everyone gets a passing grade in this function!)
   The function returns the value of the letter grade (4 for an A, 3 for a B, 2 for a C).

```
def grade(letter):
```

This quiz continues on the back of this page.

5. The function defined in the box to the right generates n random integers between 0 and 9, inclusive. Augment the code so that the function returns the number of generated integers that are odd.

   Hint: Think about the expression X % 2 and what it evaluates to when X is odd, and when X is even. For example, what is 17 % 2 ? What is 18 % 2 ?

```
import random

def odds(n):


    for k in range(n):
        r = random.randrange(10)
```

6. Consider the function whose specification appears below:

```
def primes(m, n):
    """
    What comes in: Integer m and n with m >= 2 and n >= m.
    What comes out:  Returns the number of integers
       between m and n, inclusive, that are prime.
    Example:  If m is 5 and n is 11, this function returns 3,
      since there are 3 primes between 5 and 11
      (namely, 5, 7 and 11).
    """
```

   Write code for 3 good ***test cases*** for the above function. Show both the expected and actual values.