Name: _____

Directions:

Use OMSI for this quiz. Note the following:

- Your OMSI window must fill your laptop screen at all times. Use of any Internet communication other than intended OMSI functions is not allowed.
- For coding problems, you will be given a test function, which you will copy-and-paste into your Answers window, together with the function(s) you write for the problem. When you hit Run, OMSI will run what is in your window. (Do NOT insert a line **if __name__ ...**.)
- You must hit Save for all problems.
- For noncoding problems, do not hit Run.
- If you wish, you may take a picture of your screen during the last 60 seconds of the quiz, or within 60 seconds of your leaving the room if you do so early.

1. (25) Fill in the blank: The entity in standard Python that disallows 2 computational threads from running at the same time is _____.

2. (25) Briefly (say, using at most a dozen words or so), state what the two ways in which a thread (say under the **threading** module) can lose its turn.

3. (25) Write a generator version of our "circular queue" example, in a function declared as

```
def cq(q):
```

Your code is to be tested by running

```
def testcq():
   x = [5,12,13]
   myq = cq(x)
   print myq.next()
   print myq.next()
   print myq.next()
```

whose output must be

```
5
12
13
5
```

4. (25) Here you will write a generator function to enable iterating through all elements of a Python list that match a given value. Your function will be declared as

```
def getnextmatch(x,m):
```

Here is the test code:

```
def testgnm():
   y = [5,12,13,3,4,5,6,7,8,5,12]
   g = getnextmatch(y,5)
   for j in g: print j
   g = getnextmatch(y,12)
   for j in g: print j
   g = getnextmatch(y,2)
   for j in g: print j
```

The output will be

```
0
5
9
1
10
```

Solutions:

1. GIL

2. hardware interrupt from timer or system call

3.

```
def cq(q):
    while 1:
        elt = q[0]
        q = q[1:] + [elt]
        yield elt

def testcq():
    x = [5,12,13]
    myq = cq(x)
    print myq.next()
    print myq.next()
    print myq.next()
    print myq.next()
```

4.

```
def getnextmatch(x,m):
    i = 0
    lx = len(x)
    while 1:
        if x[i] == m: yield i
        i += 1
        if i == lx: raise StopIteration

def testgnm():
    y = [5,12,13,3,4,5,6,7,8,5,12]
    g = getnextmatch(y,5)
    for j in g: print j
    g = getnextmatch(y,12)
    for j in g: print j
    g = getnextmatch(y,2)
    for j in g: print j
```