*[The following is an example of what your iRvQ2 will look like. The questions are similar (but not necessarily identical) to those in any Reading or Review Quiz (RdQ4.x, RdQ5.x, RdQ6.x, RdQ7.x,, x=1..;, RvQ4, RvQ5, RvQ6*) *from earlier in the course.]*

No computers allowed for this Review Quiz. **You are allowed one single-sided page of hand-written notes.** Answer each of the following essay questions. Each is similar to a Reading Quiz question from earlier in the course. Each question is worth 2 points. Writing something will earn you some credit > 0, but only if you write something...

**[iRvQ2-1]** What does the following code print?

**def one(a,b):**

**return a+b**

**def two(a,b):**

**return one(a\*b,a-b) - one(b-a,b\*a)**

**a=1**

**b=2**

**print (two(a,b))**

**[iRvQ2-2]** What are the sentinel values for input in the following code?

**sum = 0**

**input\_int = int(input("Enter an integer: "))**

**while input\_int % 2 == 0:**

**sum += input\_int**

**print(input\_int)**

**input\_int = int(input("Enter an integer: "))**

**print (sum)**

**[iRvQ2-3]** Complete the following code by adding one or more statements in each underlined area. Your resulting code should print **answer** as the sum of the odd integers from 1 through 47, inclusive.

You CANNOT change any code; just add new code in given areas.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**while n < 48:**

**answer = answer + n**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**print(answer)**

**[iRvQ2-4]** What does the following code print? Also, under each Boolean operator **and** and **or** in the **print()** statement,

write the number that gives the order in which that operator is applied in the evaluation of the expression with the given values. Write **1** under the operator that's applied first, **2** under the second, and so forth. Note that due to Boolean operator "short circuiting", you may have some operators that are never applied in evaluating the expression within **print()**.

**larry = 'curly'**

**moe = 'larry'**

**curly = 'moe'**

**print (larry > curly or larry < moe and curly < moe and len(larry) > len(moe) or len(moe) < len(curly))**

**[iRvQ2-5]** The following code has an infinite loop.  Explain why.

**n = 7**

**m = 5**

**count = 0**

**while n\*m != 0:**

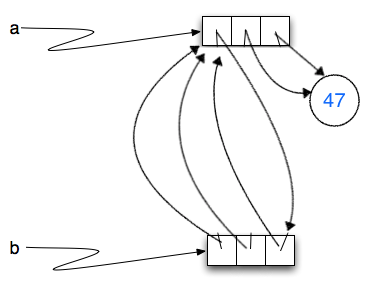
**count = count + n\*m**

**m = m - 2**

**n = n + 2**

**print(count)**

**[iRvQ2-6]** Write Python statements which assign values to **a** and **b**, such that their reference diagram looks like this:

****

**[iRvQ2-7]** What is the output when the following code is run?

**t="Moxie"**

**for index in range(len(t)-1,0,-1):**

**if index%2 == 0:**

**print (t[index])**

**[iRvQ2-8]** Define a function **check\_if\_entry(d,k,v)** for dictionary **d**, string **k,** and string **v**. It should return **True** if **k** is a string key in the dictionary **d** with string value **v**; otherwise, return **False**.

**[iRvQ2-9]** Write code that alters in place the following list **alist** so that every even integer in **alist** is replaced with **0**. "In place" means that list **alist** should reference the same list in memory after your code executes.

**alist = [[1,2],3,4,[[5,6],7]]**

**[iRvQ2-10]** Print out every character of string **s** that occurs exactly once within **s**, each on a different line of output. You may assume **s** has already been assigned a string value.