In Class Activity 4 – Time Complexity (16 points)

1. Express the running time of exercises 1-5 in 2.10 of your textbook using Big-O notation. In all cases assume that n is the input. (5 points)
2. List the running times from the exercises in order of fastest to slowest. (2 points)

(BONUS) Do the following in IDLE

   a. Copy the code from exercise 3 of chapter 2 in the textbook. Before the code create a counter set to 0 and assign n to 100. Increment the counter inside the loop to count how many times the while loop runs. Print out the counter after the loop.
   b. Assign the counter back to 0. Then update the line inside the loop so variable i is decreased by 2 on each iteration of the loop. (Keep incrementing the counter in the loop as before and print out the counter after the loop.)

   For your answer to this problem explain why results of your experiments above shows that the loop from exercise 3 uses $O(\log n)$ basic steps rather than than $O(n)$. Feel free to set n to different values initially. (2 points)

3. Write some code to explain why popping from end of a list $O(1)$ but popping from a particular index `pop(i)` is $O(n)$?
   Start with the list `a= [10,20,30,40,50,60,70,80,90]`.
   (a) What is the list after `a.pop()`?
   (b) What is the list after `a.pop(1)`?
   (c) Use the results to explain why `pop()` is and `pop(i)` is $O(n)$ (3 points)

4. Which of these is the more efficient way to build a list of n 100s? Explain your answer. (2 points)

   A:

```
mylist = []
for i in range(n):
    mylist.insert(0,100)
```

   B:

```
mylist = []
for i in range(n):
    mylist.append(100)
```

5. What is the Big-O running time of the following function that checks if a list contains the query?

```
def find(lst, query):
    for item in lst:
        if item == query:
            return True
    return False
```

   Explain your answer. (2 points)

6. Which algorithm is more efficient for finding the max value in a list with n items?
    - Algorithm #1: Walk through list, look at each item and remember the largest value seen so far
    - Algorithm #2: Sort list in increasing order, return the last item. Assume sorting takes $O(n\ log\ n)$ time

(a) Explain your answer.

(b) What do you think is the running time of python's max function? (2 points)