CS 1301 - Introduction to Computing
Spring 2017
Homework 8: Big O – Searching – Sorting

Rules:

- You must upload your submissions through gradescope.
  - Login into gradescope.
  - Select CS1301
  - Select Homework 08
  - You must select the "**SUBMIT PDF**" option.
  - Submit **HW08.pdf**
- This is an individual assignment. No collaboration is permitted.
- Due Date: **Thursday, April 6th 11:00PM.**

Name: _____ GTLogin: _____ Section_____

1. [20pts]: For each of the following pieces of code, write down the time complexity that the code will run in, choosing from `O(1), O(log n), O(n), O(n log n), O(n^2), O(n^3)`:

```
def something(n):
   for i in range(n):
       return n

for i in range(n):
   something(n)
```

Big-O:_____

```
for i in range(n):
   for j in range(5):
      print(i*j)
```

Big-O: _____

```
for i in range(n):
   for j in range(n, n/3, -9):
      print(i*j)
```

Big-O:_____

```
for i in range(521313*2213*11):
   for j in range(i ** i ** i):
      for y in range(j * i):
         print(i, j, y)
```

Big-O:_____

2. [10pts] You are going to have a dinner party where you invite N guests. If N=5, the guests will be numbered [0,1,2,3,4]. None of the guests know each other, so you write the following code to "introduce" each guest to every other guest.

```
def introduce(GuestA, GuestB):
    print(GuestA, "I'd like to introduce you to", GuestB)
    print(GuestB, "meet", GuestA)

def dinnerParty( listOfGuests):
    for guestX in listOfGuests:
        for guestY in listOfGuests:
            introduce(guestX, guestY)

dinnerParty( [0,1,2,3,4] )
```

Notice that the above code introduces the same guest to themselves, and also introduces a pair of guests twice (it introduces 0 to 1, and then 1 to 0). This is not exactly the same as a dinner party with real humans.

Your question: If you assume that a call to the introduce(...) function is your unit of work (i.e. just like a comparison in a sorting algorithm), what is the Big O complexity class of this problem? In other words, as the number of guests (N) increases, how quickly does the number of introductions increase?

Answer this question by filling in the following blanks:

If N = 2 the number of Introductions = _____

If N = 4 the number of Introductions = _____

If N = 8 the number of Introductions = _____

So therefore, the complexity class is: O(_____)

Also, if it takes 1 second to introduce each pair of guests, how many seconds will you spend doing introductions if you have 100 guests?

Answer: _____

3. [5pts] Given the following list, list the elements in the order in which binary search accesses them when searching for the number 4 (if an element is not accessed/compared then don't list it). Note: if necessary, the middle of an even sized list will be the lower index number e.g. the middle of [1, 2, 3, 4] would be 2.

[4, 5, 7, 8, 9, 11, 52]

4. [5pts] Would you use binary search or sequential search to search through the following list for some number? Why?

```
[5, 6, 3, 1, 14, 22]
```

5. [18pts] Identify the algorithm being used to sort each of the following lists, and finish sorting the list showing the new list at each step of the algorithm.

Algorithm A

Original List: `[4, 1, 3, 7, 2]`
First iteration: `[1, 4, 3, 7, 2]`
Second iteration: `[1, 3, 4, 7, 2]`

Third iteration: _____

Fourth iteration (if needed, otherwise leave blank): _____

Fifth iteration (if needed, otherwise leave blank): _____

Name of Algorithm A: _____

Algorithm B

Original List: `[4, 1, 6, 7, 2]`
First iteration: `[1, 4, 6, 7, 2]`
Second iteration: `[1, 2, 6, 7, 4]`

Third iteration: _____

Fourth iteration (if needed, otherwise leave blank): _____

Fifth iteration (if needed, otherwise leave blank): _____

Name of Algorithm B: _____

6. [7pts] Given a properly implemented merge sort algorithm and the list `[5, 3, 7, 8, 9, 11]` is it possible for the merge sort algorithm to eventually have to merge the following two lists? **Why or why not?**

`[5, 3, 7] [8, 9, 11]`

7. [20pts] Draw a diagram that illustrates how the merge sort algorithm would sort this list. Draw the contents of the list after each splitting and merging step of the algorithm:

[2, -3, 45, 10, -45, 100, 1000]

8. [15pts] Here is a sequence of numbers: `2, 13, 5, 10, 19, 1, 7, 3`

(a) [5 pts] Illustrate how a bubble-sort would sort the above list of numbers. After each pass, underline the numbers that are guaranteed to be in sorted order. Do all passes, do not make short-cutting optimizations.

(b) [5pts] Illustrate how an insertion-sort would sort the above list of numbers. After each pass, underline the numbers that are guaranteed to be in sorted order.

(c) [5 pts] Illustrate how a merge-sort would sort the above list of numbers.