All quiz rules from the course syllabus are in effect for the real quiz, in addition to what follows.

If you have questions about the test, please ask *on Piazza* as a **private** post, viewable only by you and the instructors. The instructors will make an announcement via a pinned errata post if something needs to be said to the entire class. **Do not make comments on any non-private posts on Piazza during the quiz for any reason. Making such comments may subject you to a grade penalty.**

This quiz is to be individual effort. Students are permitted to use notes, electronics, and bring textbooks. The work you submit for each quiz is expected to be produced by *you, alone and solely for this assessment*. You may not reuse or repurpose anything you wrote at another time, nor may you ever use anything written by someone else during this quiz. However, despite being allowed notes and electronics, you may *not* seek out the answer to a question in any way, nor may you communicate with anyone during the exam, *for any reason*, with the exception of asking a question on Piazza *set as instructors-only for visibility*.

You will have 40 minutes for this quiz, plus an additional ten minutes to enter your answers to the answer form. Please be careful when you enter your answers, as this will very likely be graded by a computer program and not a human. Students who have timed accommodations, such as through DSC, should use the 50 minute total as a baseline and apply their accommodations from there.

Submit your answers to this Google Form:

`https://docs.google.com/forms/d/e/1FAIpQLScZrDVJmTRrrfnLazsj3N51nlJx6p8YOvGFSVeRVo3gPx8MRw/viewform`

You may need to log in with your UCI credentials to access this. You may re-submit if you like, but your most recent submission is the one that will be graded. This means if you submit or re-submit after the end of the quiz, you will have a zero.

When 30 minutes have passed from the start of the quiz, check Piazza to see if there is a pinned thread that has clarifications to the questions. Anything on that thread by that time is something you are responsible for.

1. (1 point) For each of the following functions, determine whether it is $\mathcal{O}(n^2)$, $\Omega(n^2)$, or both. You do not need to provide proof or justification.
   Note that we are *not* asking you to provide the "best" $\mathcal{O}$-notation/Ω-notation; merely to describe it with one or both of the choices provided.

   Here, e is the base of the natural log.

   $a(n) = n^e$ ☐ $\mathcal{O}(n^2)$ ☐ $\Omega(n^2)$

   $b(n) = 4^{\log n}$ ☐ $\mathcal{O}(n^2)$ ☐ $\Omega(n^2)$

   $c(n) = n(\log n)^{10}$ ☐ $\mathcal{O}(n^2)$ ☐ $\Omega(n^2)$

2. (1.5 points) Rank the following functions in order from smallest asymptotic running time to largest. Additionally, identify a pair of functions x, y where $x(n) = \Theta(y(n))$. You do not need to show your work.

   For the ordering requirement, write the function identifier letters in order. Your submission on the "short answer" form should be exactly eight characters. For example, if you believe the functions are currently listed in asymptotic order, your answer should be "abcdefgh" (without the quotes).

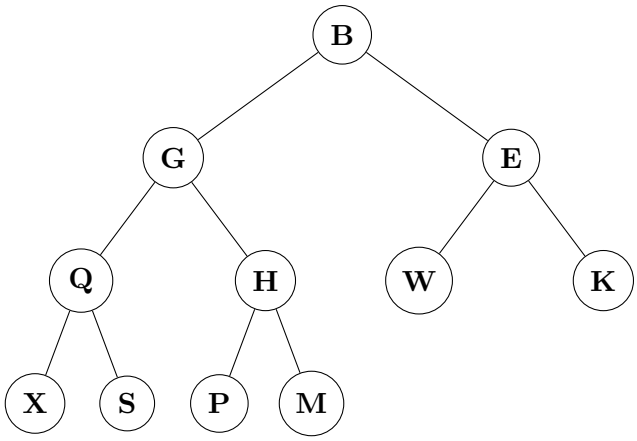
   For the Θ requirement, there are a series of checkboxes; check the ones that are Θ of one another.

   (a) $a(n) = 4n^6$
   (b) $b(n) = 0.07n \log n$
   (c) $c(n) = (1.5)^n$
   (d) $d(n) = \log^3 n$
   (e) $e(n) = \log(n!)$
   (f) $f(n) = 3 \log n$
   (g) $g(n) = 2^{42.42}$
   (h) $h(n) = 2^n$

3. (1.0 p**o**int) In lecture, we saw that an array can be considered a complete tree and, if the heap property applies, it is also a heap. In previous classes, y**o**u saw that a string (data type) is an array of characters.

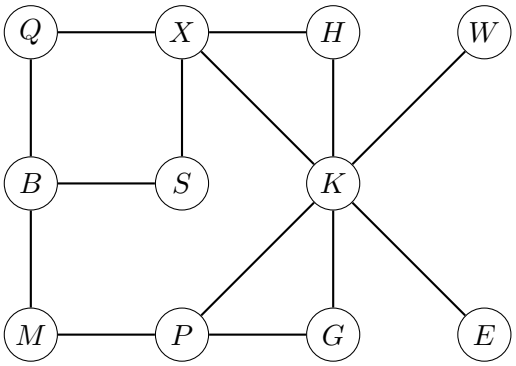
   Here, "alphabetical **o**rder" refers to standard English ordering of letters: ABCDEFGHIJKLMNOPQRSTUVWXYZ

   (a) Here is a binary heap, drawn as a tree. What is the character array/string representation of the heap? Write **only** the eleven characters in the f**o**rm.



   (b) Supp**o**se we perform a `remove-min` operation on this heap. What is the resulting array representati**o**n? You may wish t**o** do y**o**ur w**o**rk in terms of the heap drawn as a tree, above. Write only the ten characters in the form.

4. (1 point) Suppose we run a valid breadth-first order of the following graph, starting at vertex S. This means that vertex S is in layer zero on its own.

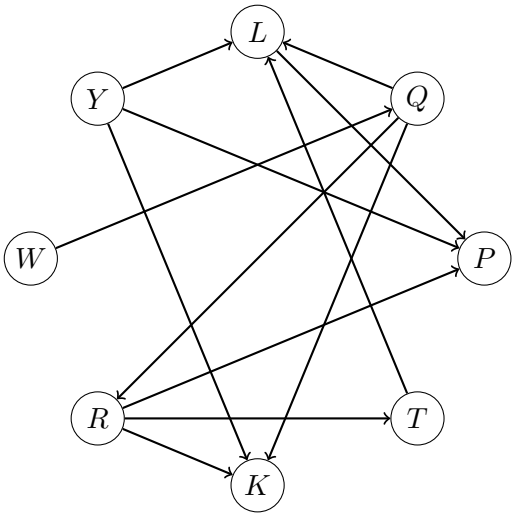


   (a) In which layer is vertex W?

   (b) In which layer is vertex K?

   (c) In which layer is vertex G?

5. (1 point) Give a valid topological order of the following graph. Write only the eight letters in the short-response answer form.



6. (0.5 points) This question deals with `InsertionSort`, the relevant code of which is reproduced for your convenience:

```
for j ← 2 to n do
    key ← A[j]
    i ← j − 1
    while i > 0 and A[i] >key do
        A[i + 1] ← A[i]
        i = i − 1
    A[i + 1] ← key
```

The f**o**llowing array is in the middle of being sorted by InsertionSort. Right now, the next line of c**o**de t**o** execute is for j to be incremented.

42	45	88	30	40	21	64	75	25	12	98	13	28	56	43

What is the maximum value that j can have right now (bef**o**re the increment**)?** Recall that when we discuss pseudo-code, we treat arrays as indexed $1 \ldots n$, so if your belief is that the outer loop has executed exactly once, your answer should be two.

7. (0.25 points) This question deals with `SelectionSort`, the relevant code of which is reproduced for your convenience:

```
for i ← 1 to n − 1 do
    min ← i
    for j ← i + 1 to n do
        if A[j] < A[min] then
            min ← j
    Swap A[i] and A[min]
```

The f**o**llowing array is in the middle of being sorted by SelectionSort. Right now, the loop conditi**o**n for the **o**uter loop (for l**oo**p indexed by i) is about t**o** be checked.

12	34	35	36	41	78	70	98	47	81	79	65	64	93	68

What is the maximum number of times the loop could have executed so far?

8. (0.25 points) This question deals with `BubbleSort`, the relevant code of which is reproduced for your convenience:

   for $i \leftarrow 1$ to $n-1$ **do**
     for $j \leftarrow 1$ to $n-i$ **do**
       if $A[j+1] < A[j]$ **then**
         Swap $A[j]$ and $A[j+1]$

   The f**o**llowing array is in the middle of being sorted by BubbleSort. Right now, the next line **o**f c**o**de to execute is f**o**r i t**o** be incremented.

26	28	51	16	56	18	41	55	58	12	21	59	77	92	97

   What is the maximum value that i can have right now (before the increment)? Recall that when we discuss pseudo-code, we treat arrays as indexed $1 \ldots n$.

9. (0.5 points) This question deals with `HeapSort`. Recall that this algorithm converts a vector into a max heap and then performs $n-1$ instances of the `extractMax` procedure.

   The f**o**llowing array is in the middle of being sorted by HeapSort. We are already past the p**o**rtion of the c**o**de that converts the vect**o**r to a max heap. We may have perf**o**rmed some number of the extractMax procedure.

54	39	35	27	32	21	55	57	60	67	75	79	89	90	92

   How many times has extractMax been performed?