Seat: ______________________

I&C SCI 46 Diagnostic Exam 1, Spring 2022

DO NOT OPEN EXAM UNTIL INSTRUCTED TO DO SO

SILENCE MOBILE PHONE AND OTHER DEVICES

This is a diagnostic exam intended to help you evaluate your readiness for the real exam.

Write the following information **clearly**. You may write **this information only** before the instructor calls to begin the exam. You **may not** write this information after the instructor calls to stop writing.

Name : ____

UCI Email Address : ____@uci.edu

UCI Student ID # : ____

Read and understand the following rules; failure to abide by these rules, or directions given by course staff during the exam, may result in disciplinary action, including but not limited to a failing grade in the class.

- This exam is solely for students enrolled in this lecture. Anyone not enrolled in this lecture may not take an exam.
- Keep your UCI ID readily accessible during the test. Proctors may request to see it.
- This exam is closed book, closed notes, and is individual effort. Once course staff begin passing out exams, you may not communicate with anyone other than proctors for any reason, nor may you have electronics, including calculators watches and phones, available to you during the test for any reason. **YOU DO NOT NEED A CALCULATOR!**
- If you leave your seat during the test for any reason, your instructor may collect it and deem you to have turned it in. Do not ask proctors for an exemption to this, they are not authorized to grant such.
- If you are still seated at 9:35 AM at the real quiz, you may not leave your seat until explicitly dismissed by the instructor. Leaving after 9:35 AM and before being dismissed may result in a penalty.
- You must take the exam in your assigned seat unless the professor (not a TA) tells you otherwise. You may not open the exam until explicitly told to do so by the professor. The instructor will call to cease writing at 9:45 AM, at which point you must immediately cease writing and close the exam. You may not write any further at that point, including finishing one's current sentence.
- If you believe a question is ambiguous, write at least two reasonable interpretations and indicate clearly which one you will be using. Then answer your question with that assumption. Unless your interpretation makes the problem much more trivial than intended, we will grade your response as if one of us had made that clarification.
- The purpose of the real exam is to evaluate how well *you* understand the material presented in the course. It is an academic integrity violation to do anything that subverts the goals of this assessment including, but not limited to, not doing your own work or submitting that of anyone else.
- Write your answers in the space provided for each question.
- Write your UCI email at the top of each answer page. You may not do this until the exam has begun. There is one point for doing this.

Nothing you write on this page will be graded. The next page in this booklet contains a spot to answer these questions. You may use this page as scratch paper if you would like, and room to do so exists.

1. (0.75 points) For each of the following functions, determine whether it is $\mathcal{O}(n)$, $\Omega(n)$, or both. You do not need to provide proof or justification. Make sure you clearly indicate, for each one, whether you believe it is $\mathcal{O}(n)$, $\Omega(n)$, or both. If the graders cannot determine your answer, you will not get credit for it.

   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.

   $a(n) = n(\log n)^{10}$ □ $\mathcal{O}(n)$ □ $\Omega(n)$

   $b(n) = n^{0.99}$ □ $\mathcal{O}(n)$ □ $\Omega(n)$

   $c(n) = 10^{100}n$ □ $\mathcal{O}(n)$ □ $\Omega(n)$

2. (1.25 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 should be exactly five characters. For example, if you believe the functions are currently listed in asymptotic order, your answer should be "abcde" (without the quotes).

   (a) $a(n) = 8^{(\log n)+1}$
   (b) $b(n) = n^8 \log n$
   (c) $c(n) = n \cdot (\log n)^8$
   (d) $d(n) = 10^4!$
   (e) $e(n) = 2^{6 \log \sqrt{n}}$

3. (4 points) Suppose I want to have a "Blue and Gold Stack" class. This is a class that has color-coded push, pop, top and size functions; for example, there is a "blue push" and a "gold push." Explain how you can implement this using a single array whose capacity is set at some value C, which will always be larger than the combined sizes of the blue and gold stacks. Your approach may only use the array of size C and $\mathcal{O}(1)$ additional space. You do not need to write code, but instead describe how you would implement the eight functions.

   You will earn no credit if you solve this by simply having two traditional stacks as private data members and redirecting the calls to `bluePush()` to the blue one, etc.

Write your answer for question 1 here

1(a)	1(b)	1(c)

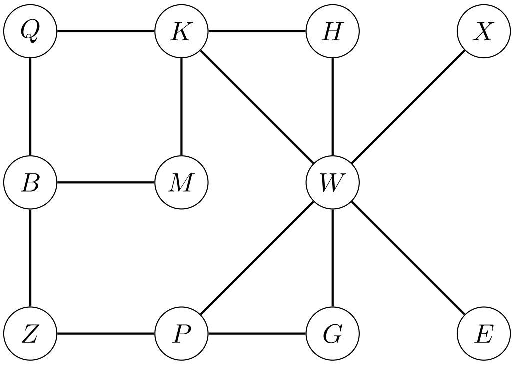
Write your answer for question 2 here.
You should write only the five letters in this box.

2

Write your answer for question 3 here. Be sure the grader can find each part.

Nothing you write on this page will be graded. The next page in this booklet contains a spot to answer these questions. You may use this page as scratch paper if you would like, and room to do so exists.

4. (2 point) Suppose we run a valid breadth-first order of the following graph, starting at vertex G. This means that vertex G is in layer zero on its own.



Write the contents of each layer.

5. (3 points) **The most disgusting recursive problem ever.**

The McDonald's near campus sells Chicken McNuggets in orders of 6, 9, or 20. Suppose you are ordering for a party and you know exactly how many McNuggets will be eaten by guests. It turns out that, for any integer $n \geq 44$, you can order exactly n Chicken McNuggets at this McDonald's.

For purposes of this problem, you cannot throw out McNuggets or allow them to go uneaten, such as by acquiring $n = 44$ by buying two twenty packs and a six pack, then discarding two. If the thought of this many Chicken McNuggets is too disgusting, you may pretend you are buying $n \geq 44$ celery sticks in bunches of 6, 9, or 20 (feel free to rename the function below in that case).

Finish the recursive function below to complete the ordering and return the counts by reference parameters. You may assume for this problem that there will be no overflow or underflow at any point in the problem and that stack space is not a concern. The code has been started for you and is part of a correct solution.

```
void buyChicken (unsigned n, unsigned & num6Packs, unsigned & num9Packs,
                  unsigned & num20Packs)
{
  if( 44 == n ){
    num20Packs = 1;  num6Packs = 4;  num9Packs = 0;
  }else if (45 == n ){
    num20Packs = 0; num6Packs = 3;  num9Packs = 3;
  }else if (46 == n ){
    num20Packs = 2; num6Packs = 1; num9Packs = 0;
  }
```

Write your answer for question 4 here. In each box, write only the given layer's contents. You may write each layer in any order. If a layer is empty, write nothing in the layer.

Layer 1	Layer 2	Layer 3
Layer 4	Layer 5	Layer 6

Write your answer for question 5 here. You do not need to copy over the starting code given, nor should you.

Nothing you write on this page will be graded. The next page in this booklet contains a spot to answer these questions. You may use this page as scratch paper if you would like, and room to do so exists.

6. (2 points) Suppose I am writing a program that will use a skip list with maximum height $h = 3\lceil \log n \rceil$ - that is, when I add an element to the skip list, if I get $3\lceil \log n \rceil$ "heads" in a row, I will stop adding height to the element. You cannot modify my code or select the input. You can, however, change the way the coin works from "50% heads, 50% tails" (current status) to another set of probabilities (which must still add up to 100%). You cannot introduce other results: the coin can still only land on heads or tails (no sideways landing, for instance).

   Your goal is to cause my Skip List to use the **maximum possible amount of space**. How would you bias the coin to achieve this effect? Explain briefly why this will cause the given problem.

7. (4 points) Consider the following definition for a node in a linked list:

```
struct Node
{
  Node(unsigned v, Node * n) : value(v), next(n) {}
  unsigned value;
  Node * next;
};
```

   Suppose we have a linked list of these and wish to answer a question: is there a subset of these Nodes such that the combined value equals a particular value 'target'?

   Complete the following function that returns a bool value to indicate the answer to the question. You may assume that the initial call gives a link to the front of the linked list and that no overflow or underflow will occur at any point in the run of a program.

   Hint: if there is a subset of the Nodes whose combined value adds to the target, then either the first element is in that subset or it isn't.

   This problem is substantially easier if you use recursion. For many students, this will be the hardest question on the test. You may wish to consider answering it last.

```
bool subsetSum(Node * front, unsigned target)
{
```

Write your answer for question 6 here.

Write your answer for question 7 here.