

IC312: Data Structures

6 Week Exam

September 27, 2021

Answer the questions in the spaces provided on the question sheets. **If you run out of room for an answer, continue on the last page of this exam.** Show all work, but please be legible.

Name: _____

Section: _____

Page	Points	Score
2	18	
3	11	
4	10	
5	8	
6	14	
7	9	
8	8	
9	22	
Total:	100	

The Naval Service I am a part of is bound by honor and integrity. I will not compromise our values by giving or receiving unauthorized help on this exam.

1. Fill in the following table with the correct run times (using Big-O notation and amortized solutions where appropriate) for each method of each ADT, as implemented with each data structure. Assume everything is implemented the best way we know how, and please indicate where amortization was applied with an asterisk (*). [10 pts]

		Array	Linked List
QUEUE	enqueue(e)		
	dequeue()		
	peek()		
STACK	push(e)		
	pop()		
	peek()		
LIST	get(i)		
	set(i,e)		
	add(i,e)		
	remove(i)		

2. Give a c and an n_0 to show that $100n + 10 \in O(n)$ [4 pts]

3. Give a c and an n_0 to show that $50n^2 - n + 50 \in O(n^3)$ [4 pts]

4. Suppose we are running a simple linear search as appears in pseudocode below, to find if an element appears in an unsorted array. What is n ? What circumstances make up this algorithm's "best case?" What about its "worst case?" [3 pts]

```

find(int[] array, int element):
  for (i=0; i<array.length; i++)
    if array[i]==element:
      return true
  return false

```

5. Rank the following in order from fastest to slowest, with a ranking of 1 being the fastest, and a ranking of 6 being the slowest: [4 pts]

- Logarithmic time
- $n\text{-log-}n$
- Exponential time
- Constant time
- Quadratic time
- Linear time

6. If $f(n) = 3n \log_2(n) + 5n^2 + 2$, circle ALL definitions of $g(n)$ such that $f(n) \in O(g(n))$. [2 pts]

- A. $g(n) = n^2$
- B. $g(n) = n$
- C. $g(n) = n \log(n)$
- D. $g(n) = 2^n$

7. True or False: It is possible for an algorithm to be $O(\log_2(n))$, yet NOT be $O(\log_{10}(n))$. [1 pts]

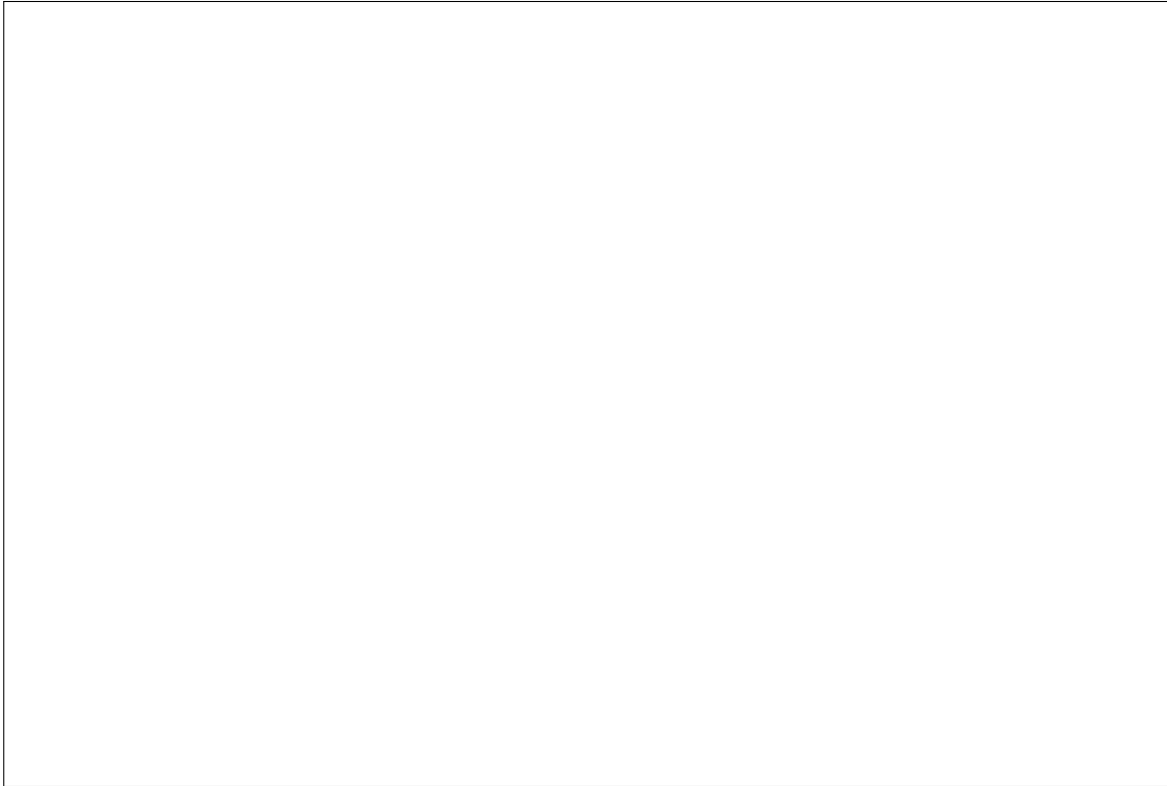
- A. True
- B. False

8. What is the runtime of adding an element to a sorted array? [1 pts]

- A. $O(n)$
- B. $O(1)$
- C. $O(\log(n))$
- D. $O(n^2)$

9. Complete the following function which creates a new Node which stores the input integer, such that the linked list will be in order from smallest to largest. You may assume the existing list is already in order. You may choose to write a second private function. Your solution must be recursive. [8 pts]

```
public void addInOrder(int theInt){
```



10. Which of the following is the runtime of this algorithm? [2 pts]

```
f(n){  
    sum=0  
    for (i=0; i<n; i++){  
        sum++  
    }  
    for (i=0; i<n; i++){  
        for (j=0; j<i; j++){  
            sum++  
        }  
    }  
    print(sum)  
}
```

- A. $O(n \log(n))$
- B. $O(\log(n))$
- C. $O(n^2)$
- D. $O(n^3)$

11. Which of the following is the runtime of this algorithm?

[2 pts]

```
f(n){
  sum=0
  for (j=-10; j<n; j+=20) {
    for (i=1; i<n; i*=2){
      sum++
    }
  }
  print(sum)
}
```

- A. $O(n \log(n))$
- B. $O(\log(n))$
- C. $O(n^2)$
- D. $O(n^3)$

12. Write the recurrence relation for the following algorithm:

[6 pts]

```
void f(n) {
  if n==1:
    print "hello!"
    return
  for (int i = 0; i < n/2; i++)
    print "goodbye!"
  f(n-1)
}
```

13. Write the recurrence relation for the following algorithm:

[6 pts]

```
void f(int[] array, int beg, int end) {  
    if (end-beg)==1:  
        return  
    mid=(beg+end)/2  
    f(array,beg,mid)  
    f(array,mid,end)  
}
```



14. Find the big-O for the following recurrence relation. **SHOW YOUR WORK.**

[8 pts]

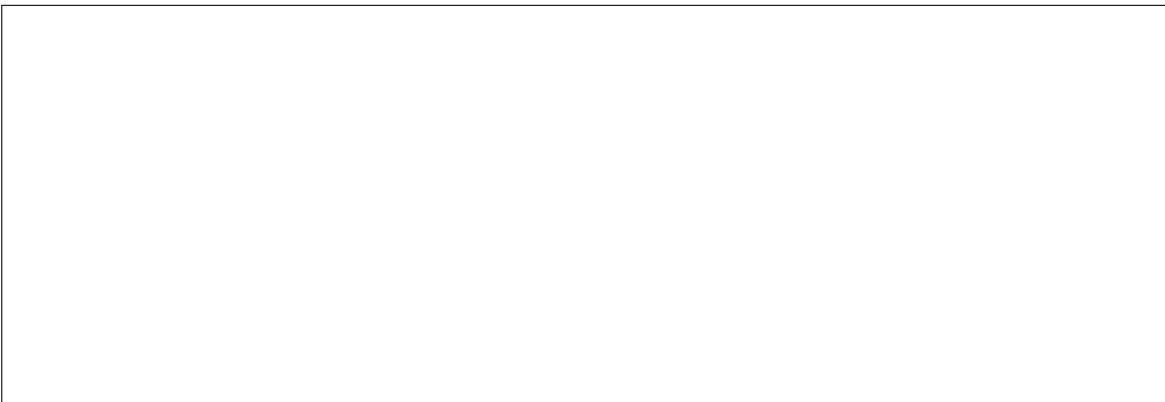
$$T(n) = \begin{cases} 7, & n \leq 1 \\ 2n + 2T(n/2), & n \geq 2 \end{cases}$$

15. Justify why when adding elements to an array, if the array becomes full, one must double the size of the array rather than increase its size by a fixed amount. [5 pts]



16. What is the output of the following sequence of operations: [4 pts]

```
s.push('A')
s.push('B')
print(s.peak())
print(s.pop())
s.push('C')
print(s.pop())
print(s.pop())
```



17. What is the output of the following sequence of operations:

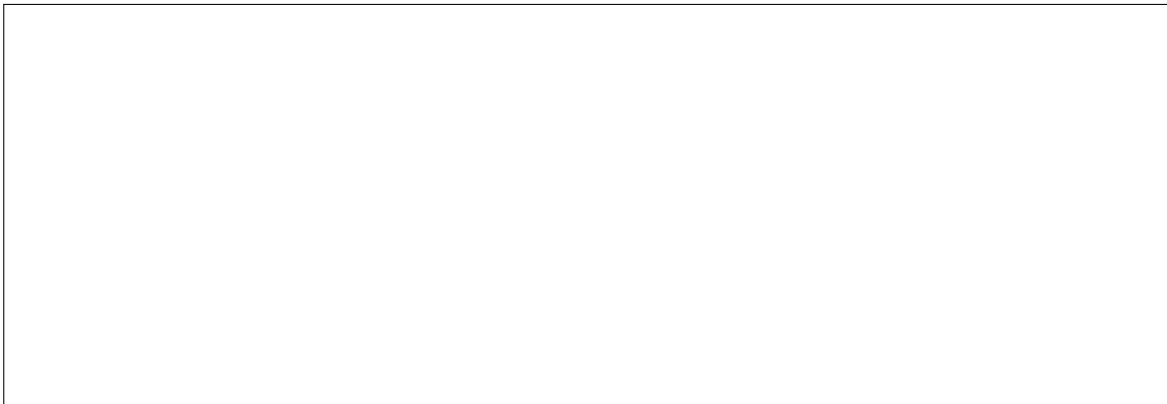
[4 pts]

```
q.enqueue('A')
q.enqueue('B')
print(q.peak())
print(q.dequeue())
q.enqueue('C')
print(q.dequeue())
print(q.dequeue())
```



18. Draw a tree with a height of 2.

[2 pts]

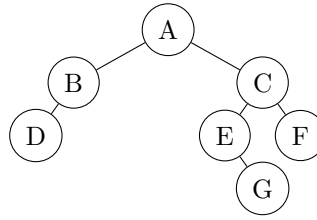


19. What is the height of a tree with 0 nodes?

[2 pts]



20. Consider the following tree:



(a) Give the POST-ORDER traversal of the tree [4 pts]

(b) Give the IN-ORDER traversal of the tree [4 pts]

(c) Give the PRE-ORDER traversal of the tree [4 pts]

(d) Give the LEVEL-ORDER traversal of the tree [2 pts]

21. Given a standard definition of a Binary Tree Node with left and right child pointers and data consisting of ints, complete the following function recursively so that it returns a boolean indicating whether an element appears in the tree. Feel free to make a private function to perform your recursion. [8 pts]

```
public boolean find(int theInt){
```

This is a scratch sheet for additional work.

Arithmetic Sum:

$$\sum_{i=1}^n i = \frac{n(n+1)}{2}$$

Geometric Sum:

$$\sum_{i=0}^{n-1} r^i = \frac{1-r^n}{1-r}$$