Q1 Honor Code

0 Points

I understand that I must not talk to anyone else about this exam before it has been graded. I also understand that talking to someone else during the exam would be not only an academic integrity violation, but also be a violation of the trust placed in me by the course staff.

Q2

1 Point

What is the time complexity of the following function?

```
def foo(n):
    for i in range(n**2 - 2*n + 100):
        for j in range(n + n**0.5 - 20):
            print(i + j)
```

○ $\Theta(1)$

○ $\Theta(\log n)$

○ $\Theta(\sqrt{n})$

○ $\Theta(n)$

○ $\Theta(n \log n)$

○ $\Theta(n\sqrt{n})$

○ $\Theta(n^2)$

● $\Theta(n^3)$

○ $\Theta(2^n)$

Q3

1 Point

What is the time complexity of the following function?

```python
def foo(n):
    for i in range(n):
        for j in range(n**2):
            for k in range(j):
                print(j)
```

○ $\Theta(\sqrt{n})$

○ $\Theta(n)$

○ $\Theta(n \log n)$

○ $\Theta(n\sqrt{n})$

○ $\Theta(n^2)$

○ $\Theta(n^3)$

○ $\Theta(n^4)$

● $\Theta(n^5)$

○ $\Theta(n^6)$

Q4

1 Point

What is the time complexity of the following function?

```python
def foo(arr):
    """arr is an array of size n"""
    n = len(arr)
    for i in range(n):
        r = sum(arr) * sum(arr)
        print(r)
```

○ $\Theta(1)$

○ $\Theta(\log n)$

○ $\Theta(\sqrt{n})$

○ $\Theta(n)$

○ $\Theta(n \log n)$

○ $\Theta(n\sqrt{n})$

● $\Theta(n^2)$

○ $\Theta(n^3)$

○ $\Theta(2^n)$

Q5

1 Point

What is the time complexity of the following function?

```
def foo(n):
    for i in range(n):
        for j in range(n):
            for k in range(i): # <--- this is range(i), not range(n)!
                print(i + j + k)
```

○ $\Theta(1)$

○ $\Theta(\log n)$

○ $\Theta(\sqrt{n})$

○ $\Theta(n)$

○ $\Theta(n \log n)$

○ $\Theta(n\sqrt{n})$

○ $\Theta(n^2)$

● $\Theta(n^3)$

○ $\Theta(2^n)$

Q6

1 Point

What is the time complexity of the following function?

```
def foo(n):
    i = 1
    while i < n:
        j = 0
        while j < n:
            j += 1
        i *= 2
```

○ $\Theta(1)$

○ $\Theta(\log n)$

○ $\Theta(\sqrt{n})$

○ $\Theta(n)$

● $\Theta(n \log n)$

○ $\Theta(n\sqrt{n})$

○ $\Theta(n^2)$

○ $\Theta(n^3)$

○ $\Theta(2^n)$

Q7

1 Point

What is the *best case* time complexity of the following function?

```
def foo(arr):
    """arr is an array of size n"""
    for x in arr:
        for y in arr:
            if (x + y) == 5:
                return sum(arr)
    return False
```

○ $\Theta(1)$

○ $\Theta(\log n)$

○ $\Theta(\sqrt{n})$

● $\Theta(n)$

○ $\Theta(n \log n)$

○ $\Theta(n\sqrt{n})$

○ $\Theta(n^2)$

○ $\Theta(n^3)$

○ $\Theta(2^n)$

Q8

1 Point

What is the *worst case* time complexity of the following function?

```
def foo(arr):
    """arr is an array of size n"""
    if 5 in arr:
        for i in range(n**0.5):
            print(i)
    else:
        mergesort(arr)
```

○ $\Theta(1)$

○ $\Theta(\log n)$

○ $\Theta(\sqrt{n})$

○ $\Theta(n)$

● $\Theta(n \log n)$

○ $\Theta(n\sqrt{n})$

○ $\Theta(n^2)$

○ $\Theta(n^3)$

○ $\Theta(2^n)$

Q9

1 Point

What is the *best case* time complexity of the following function?

```
def foo(arr):
    """arr is an array of size n"""
    if 5 in arr:
        for i in range(n**0.5):
            print(i)
    else:
        mergesort(arr)
```

○ $\Theta(1)$

○ $\Theta(\log n)$

● $\Theta(\sqrt{n})$

○ $\Theta(n)$

○ $\Theta(n \log n)$

○ $\Theta(n\sqrt{n})$

○ $\Theta(n^2)$

○ $\Theta(n^3)$

○ $\Theta(2^n)$

Q10

1 Point

What is the *expected* time complexity of the following function?

```
import random

def foo(n):
    """arr is an array of size n"""
    # randomly draw a number from 0, 1, 2, ..., n-1
    x = random.randint(0, n-1)
    if x > n // 2:
        for i in range(n**2):
            print(i)
    else:
        print("lucky!")

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

○ $\Theta(1)$

○ $\Theta(\log n)$

○ $\Theta(\sqrt{n})$

○ $\Theta(n)$

○ $\Theta(n \log n)$

○ $\Theta(n\sqrt{n})$

● $\Theta(n^2)$

○ $\Theta(n^3)$

○ $\Theta(2^n)$

Q11

1 Point

State (but do not solve) the recurrence relation describing this function's run time.

```python
import random

def foo(n):
    if n <= 3:
        return

    for i in range(n**2):
        print(i)

    return foo(n//2) + foo(n//2)
```

T(n) = 2 T(n/2) + n^2

Q12

1 Point

What is the time complexity of the following function?

```
import random

def foo(n):
    """arr is an array of size n"""
    print(n)
    if n <= 1:
        return
    foo(n//2)
```

○ $\Theta(1)$

● $\Theta(\log n)$

○ $\Theta(\sqrt{n})$

○ $\Theta(n)$

○ $\Theta(n \log n)$

○ $\Theta(n\sqrt{n})$

○ $\Theta(n^2)$

○ $\Theta(n^3)$

○ $\Theta(2^n)$

Q13

1 Point

Consider the function $f(n) = 5n^2 - 100n + 100$. Which of the following asymptotic bounds are true? Choose **all** that apply.

☐ $\Theta(n)$

☐ $O(n)$

☑ $\Omega(n)$

☑ $\Theta(n^2)$

☑ $O(n^2)$

☑ $\Omega(n^2)$

☐ $\Theta(n^3)$

☑ $O(n^3)$

☐ $\Omega(n^3)$

Q14

1 Point

Let $f(n) = (n + \log n) \times \frac{5n^2+3n-100}{n(n-1)}$. Which of the following asymptotic bounds on f is true?

○ $\Theta(1)$

○ $\Theta(\log n)$

○ $\Theta(\sqrt{n})$

● $\Theta(n)$

○ $\Theta(n \log n)$

○ $\Theta(n\sqrt{n})$

○ $\Theta(n^2)$

○ $\Theta(n^3)$

○ $\Theta(2^n)$

Q15

1 Point

Let $f(n) = \log_2(n^3 + n(n+1))$. Which of the following asymptotic bounds on f is true?

○ $\Theta(1)$

● $\Theta(\log n)$

○ $\Theta(\sqrt{n})$

○ $\Theta(n)$

○ $\Theta(n \log n)$

○ $\Theta(n\sqrt{n})$

○ $\Theta(n^2)$

○ $\Theta(n^3)$

○ $\Theta(2^n)$

Q16

1 Point

True or False. If $f = O(n^5)$ and $f = \Omega(n^4)$ then f must be either $\Theta(n^4)$ or $\Theta(n^5)$.

○ True

● False

Q17

1 Point

True or False. If $f_1 = \Theta(g_1(n))$ and $f_2 = \Omega(g_2(n))$ then $\frac{f_1}{f_2} = \Theta(g_1/g_2)$.

○ True

● False

Q18

1 Point

The code below performs a binary search. It is the same as the pseudocode given in lecture but has been modified to print each value encountered in the query.

```
def binary_search(arr, start, stop, target):
    if stop <= start:
        return None

    middle = math.floor((start + stop) / 2)

    print(arr[middle])

    if arr[middle] == target:
        return middle
    elif arr[middle] < target:
        return binary_search(arr, middle + 1, stop, target)
    else:
        return binary_search(arr, start, middle, target)
```

Suppose this query method is called with a target of 62, and that the array it is called on is sorted. Which one of the following sequences of keys could possibly have been printed?

○ 80, 45, 72, 40, 60, 62

○ 80, 90, 72, 55, 60, 62

○ 62, 80, 12, 15, 20, 50

● 80, 45, 72, 49, 60, 62

○ 80, 62, 50, 20, 70, 62

Q19

1 Point

Suppose binary search is called on an *unsorted* array. Which one of the following will happen?

The code of binary search is reproduced below for convenience.

```
def binary_search(arr, start, stop, target):
    if stop <= start:
        return None

    middle = math.floor((start + stop) / 2)

    if arr[middle] == target:
        return middle
    elif arr[middle] < target:
        return binary_search(arr, middle + 1, stop, target)
    else:
        return binary_search(arr, start, middle, target)
```

○ The right answer will always be returned, but the code will take longer to run.

● The code will finish running without error, but the answer may be incorrect.

○ An error will be returned because the array is indexed out-of-bounds.

○ The function will run forever.

Q20

1 Point

Consider the below code, which is the same as in lecture except that the `<` has been changed to `>` in the line marked by `<---- HERE`.

```
def mergesort(arr):
    if len(arr) > 1:
        middle = math.floor(len(arr) / 2)
        left = arr[:middle]
        right = arr[middle:]

        mergesort(left)
        mergesort(right)

        merge(left, right, arr)

def merge(left, right, out):
    left.append(float('inf'))
    right.append(float('inf'))
    left_ix = 0
    right_ix = 0

    for ix in range(len(out)):
        if left[left_ix] > right[right_ix]: # <---- HERE
            out[ix] = left[left_ix]
            left_ix += 1
        else:
            out[ix] = right[right_ix]
            right_ix += 1
```

Suppose `mergesort` with the modified `merge` is called on an array. Which of the following will happen?

● The new `mergesort` will correctly sort its input, but now in *descending* order.

○ The new `mergesort` will run forever.

○ The new `mergesort` will raise an error because an array is accessed out-of-bounds.

○ The new `mergesort` will complete without raising an error, but the resulting array may not be in sorted order (ascending or descending).

Q21

1 Point

True or False. `binary_search` assumes that its input is in sorted order.

● True

○ False

Q22

1 Point

True or False. `quickselect` assumes that its input is in sorted order.

○ True

● False

Q23

1 Point

Suppose the `partition` function from `quickselect` has been called on the array `arr` with an unknown pivot. Which of the following arrays could be the result?

○ 3, 6, 9, 10, 4, 5

● 5, 2, 4, 6, 10, 9

○ 10, 6, 9, 3, 4, 5

○ 5, 6, 10, 3, 4, 9

Q24

1 Point

Suppose a binary search tree contains 10 keys, all unique. True or False: it is possible for the root of the tree to contain the minimum key.

● True

○ False

Q25

1 Point

Suppose a binary search tree with n nodes is balanced. What is the *worst* case time complexity of any efficient algorithm that finds the minimum key? That is, what is a theoretical lower bound for this problem?

○ $\Theta(1)$

● $\Theta(\log n)$

○ $\Theta(\sqrt{n})$

○ $\Theta(n)$

○ $\Theta(n \log n)$

○ $\Theta(n\sqrt{n})$

○ $\Theta(n^2)$

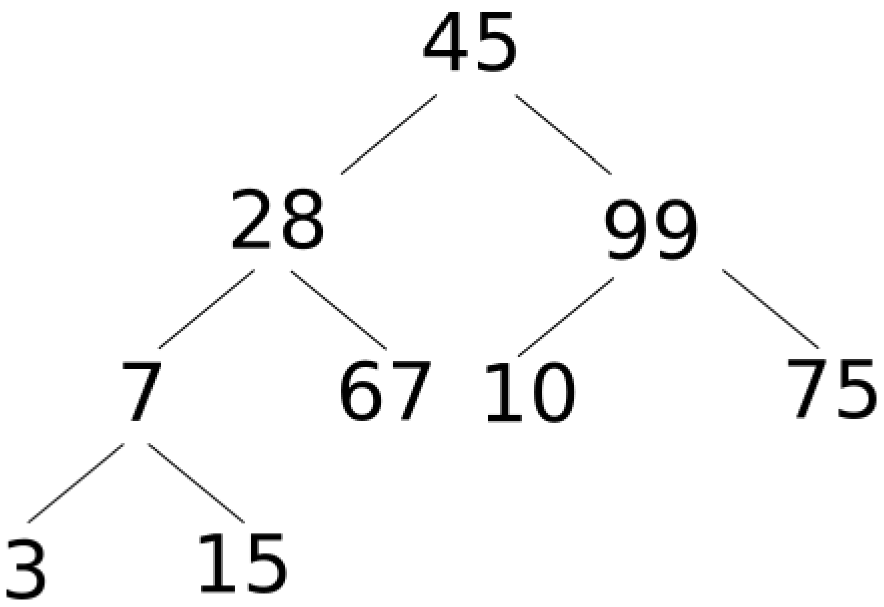
○ $\Theta(n^3)$

○ $\Theta(2^n)$

Q26

1 Point

True or False: the tree shown below is a binary search tree.



○ True

● False