Name: _____ Section: _____

INSTRUCTIONS:

(1) DO NOT DISTRIBUTE THIS EXAMINATION.

(2) DO NOT OPEN YOUR EXAM BOOKLET UNTIL YOU HAVE BEEN TOLD TO BEGIN.

(3) The total for the exam is 100 points

(4) Use the backside of the paper as your scratch paper. The backside is not graded.

(5) If you make a mistake, cross it out or erase it. Otherwise, it will be graded, for better or for worse.

(6) No electronic devices (e.g. phones, calculators, mp3 players, etc.) are allowed.

(7) You are allowed one sheet of notes on Letter paper or smaller.

(8) Write your name on the upper-right corner of each page of the exam.

(9) **To receive full credit, you must write legibly and your answers must be clear.** You may need to sketch out answers to less straightforward problems on the scratch side before you write on the front side.

(10) Do not leave classroom until you have handed in the exam.

(11) The exam lasts 90 minutes

Problems 1-15: Multiple-choice. Circle the letter of the best response.

(1) For the following program segment, give the best analysis of the running time in Big-Oh notation

```
sum = 0
for i = 1 to n^2:
    for j = 1 to n:
        sum = sum + 1
```

(a) $O(1)$
(b) $O(\log n)$
(c) $O(n)$
(d) $O(n^2)$
(e) None of the above.

(2) Below is a function from a `LinkedList` class which determines if the list is empty. Give the best analysis of the running time in Big-Oh notation.

```
def is_empty(L):
    return L.head == None
```

(a) $O(1)$
(b) $O(\log n)$
(c) $O(n)$
(d) $O(n^2)$
(e) None of the above.

(3) Determine whether the following statement is true or false.

$$\frac{n(n+1)}{2} \in \Theta(n^3)$$

(a) True
(b) False
(c) Not enough information

(4) For the following pair of functions, determine whether the first function has a lower, same, or higher order of growth (to within a constant multiple) than the second function.
First: 2^{n-1}
Second: 2^n

(a) lower
(b) same
(c) higher

(5) What does the following function on list `L` do?

```
def MysteryFunction(L):
for i in range(len(L) - 1):
    for j in range(i + 1, len(L)):
        if L[i] == L[j]:
            return False
return True
```

(a) Returns False if all elements of `L` are swapped.

(b) Returns True if all elements are sorted in ascending order.

(c) Returns True after counting the occurances of an entry.

(d) Returns True if all elements in `L` are distinct.

(e) None of the above.

(6) If L has length n, what is the running time of the `MysteryFunction`?

(a) $O(1)$

(b) $O(\log n)$

(c) $O(n)$

(d) $O(n^2)$

(e) None of the above.

(7) Which is the following is the best *stable* sort (Hint: A sorting function is stable if $A_i = A_j$ and $i < j$ then A_i always comes before A_j in the sorted list)?

(a) Bubble Sort

(b) Insertion Sort

(c) Selection Sort

(d) Merge Sort

(e) Quick Sort

(8) What is worst-case number of items that must be checked to determine if an element k exists in a *sorted* list of 64 items?

(a) 1

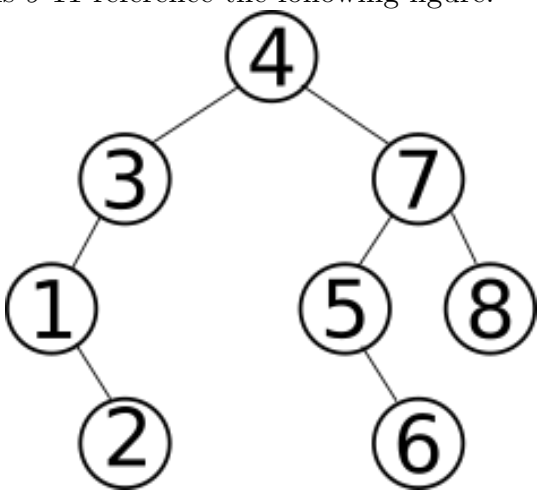
(b) 6

(c) 7

(d) 63

(e) 64

Problems 9-11 reference the following figure.



(9) Is the above tree a valid AVL tree?

(a) Yes

(b) No. The node containing 3 is unbalanced.

(c) No. The node containing 4 is unbalanced.

(d) No. The node containing 5 is unbalanced.

(e) No. A different node is unbalanced.

(10) What is the height of the tree?

(a) 2

(b) 3

(c) 4

(d) 5

(e) None of the above

(11) What is the pre-order traversal of the tree?

(a) [1, 2, 3, 4, 5, 6, 7, 8]

(b) [2, 1, 3, 4, 6, 5, 8, 7]

(c) [4, 3, 1, 2, 4, 5, 6, 8]

(d) [2, 1, 3, 6, 5, 8, 7, 4]

(e) None of the above

(12) What is the running time of Insertion Sort if the elements of the list are originally in *decending* order?

(a) $O(\log n)$

(b) $O(n)$

(c) $O(n \log n)$

(d) $O(n^2)$

(e) None of the above.

(13) `L` is initially the list [0, 1, 2, 3, 4, 5, 6, 7, 8, 9] with a `retain_if` statement like the one in Project 2. What is the contents of `L` after executing the statement `L.retain_if(lambda x: x % 2 == 0)`?

(a) $[0, 2, 4, 6, 8]$

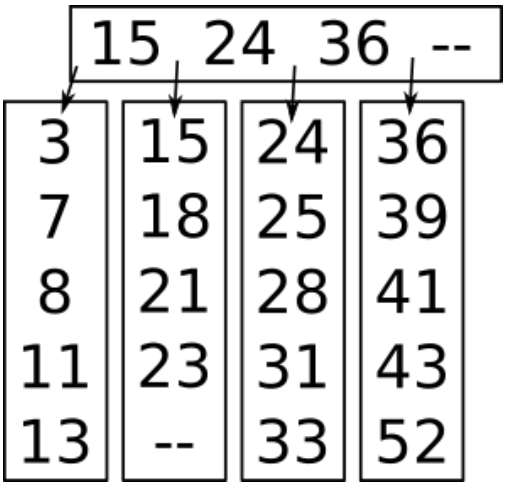
(b) $[1, 3, 5, 7, 9]$

(c) [0]

(d) [2]

(e) None of the above.

(14) What is the contents of the root node after inserting 35 into the following B-Tree with M = 5 and L = 5?



(a) [15, 24, 36]

(b) [15, 24, 28, 36]

(c) [15, 24, 31, 36]

(d) [15, 24, 35, 41]

(e) [15, 24, 36, 41]

(15) Which of the following can be an in-order traversal of a binary search tree containing three keys?

(a) [8, 6, 12]

(b) [12, 8, 6]

(c) [6, 12, 8]

(d) [6, 8, 12]

(e) None of the above.

(16) Prove that $n^3 + n + 7$ is $\Theta(n^3)$.

(17) Let `Q` be a deque represented by a circular array with capacity 30, which does not change during this problem. Initially 14 elements are enqueued, starting at physical index 7. The following steps are performed in order.

- `PushBack` is called 11 times.
- `PopFront` is called 8 times.
- `PeekFront` is called 3 times.
- `PushFront` is called 9 times.
- `PopBack` is called 6 times.

Identify the physical indices of the new first and last elements of the deque.

(18) Write a function `OrderReverse(A, B)` that returns `True` if `A` comes *after* `B` according to their natural ordering and `False` otherwise. (Hint: A and B are the same type and have a $<$ operator.)

```
def OrderReverse(A, B):
```

(19) Use Quick-Sort to sort the list [2, 3, 7, 4, 6, 5, 8, 1]. Show each step.

(20) Considering the following function that sorts a list `L`.

```
def OddEvenSort(L):
    sorted = False
    while not sorted:
        sorted = True
        for i in range(1, len(L) - 1, 2): # Odd indices
            if L[i] > L[i+1]:
                L[i], L[i+1] = L[i+1], L[i] # Swap
                sorted = False
        for i in range(0, len(L)-1, 2): # Even indices
            if L[i] > L[i+1]:
                L[i], L[i+1] = L[i+1], L[i] # Swap
                sorted = False
```

(a) Identify the worst-case runtime in Big-Oh notation. Justify your answer.

(b) Identify the best-case runtime in Big-Oh notation. Justify your answer.

(c) Identify the average-case runtime in Big-Oh notation. Justify your answer.

(d) What does `L` look like after one iteration of `OddEvenSort` if `L` is initially [2, 3, 7, 4, 6, 5, 8, 1]?

(21) Let `S` be an unsorted stack. Using a second stack and a fixed amount of additional memory, write a function `SortStack` that sorts `S` in $O(n^2)$ time. The smallest item should end up on the top of `S` and the largest item on the bottom of `S`.

```
def SortStack(S):
    T = Stack()
```

(22) Consider the list [2, 3, 7, 4, 6, 5, 8, 1].

(a) Create a binary search tree by inserting each of the elements in the list in the given order.

(b) Find the post-order traversal of your tree.