CISC 220 – *Regular* section

Midterm

Friday, October 15, 2010

This exam is worth 15% of your grade. Read all instructions before answering—some questions are multi-part. Partial credit will be given in 0.25-point gradations; show your work where possible. Write your answers on both sides of exam pages as necessary.

Q1 (3 points)

In C/C++ code, write a `dequeue()` member function for a class `Queue` which is implemented with a singly-linked list of `Node` objects (each with member variables `int data` and `Node *next`). Assume that `Queue` has member variables `Node *back` and `Node *front`. The return type of `dequeue()` should be `int`, and your function should include appropriate error-checking.

Q2 (1 point)

A *binary heap* is a binary tree with two additional properties. Name and explain those properties.

Q3 (2 points)

Consider the following function:

```
void FunctionX(int n, int a, int b) {
  for (int i = 0; i < n; i++)
    for (int j = i; j < n; j++)
      if (a < b) FunctionA(n);
      else FunctionB(n);
}
```

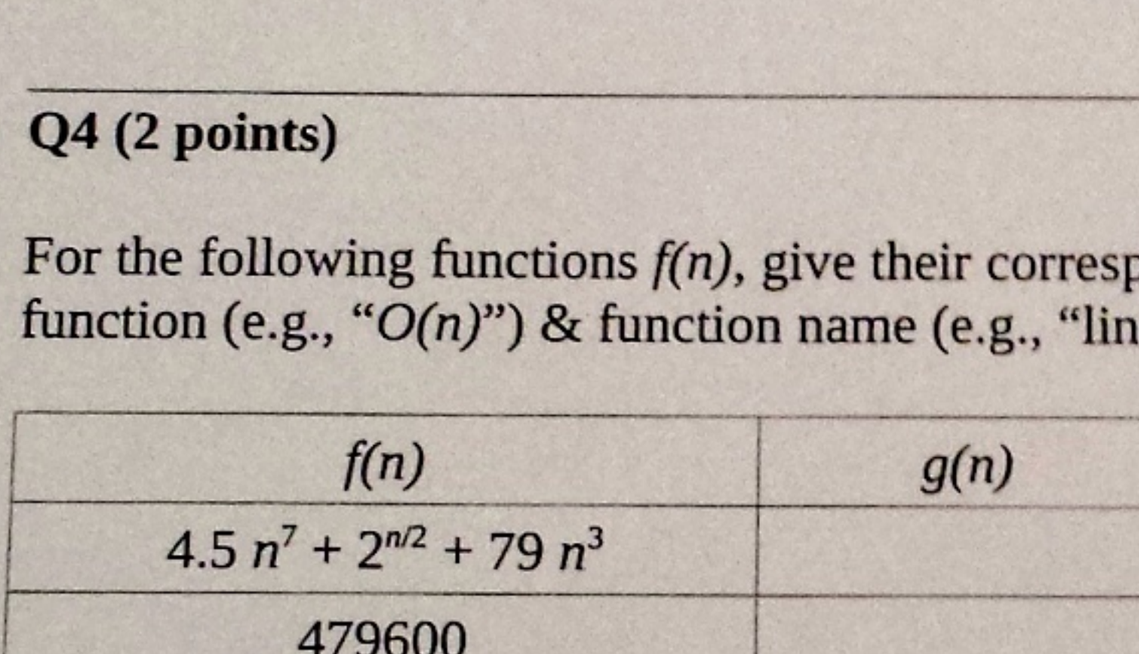
Assuming that `FunctionA` takes $3n + 1$ primitive operations and `FunctionB` takes $5 \log n$ operations, how many operations does `FunctionX` require in terms of n in the worst case? Show the steps you take to get your answer. What is its corresponding big-O running time?

Q4 (2 points)

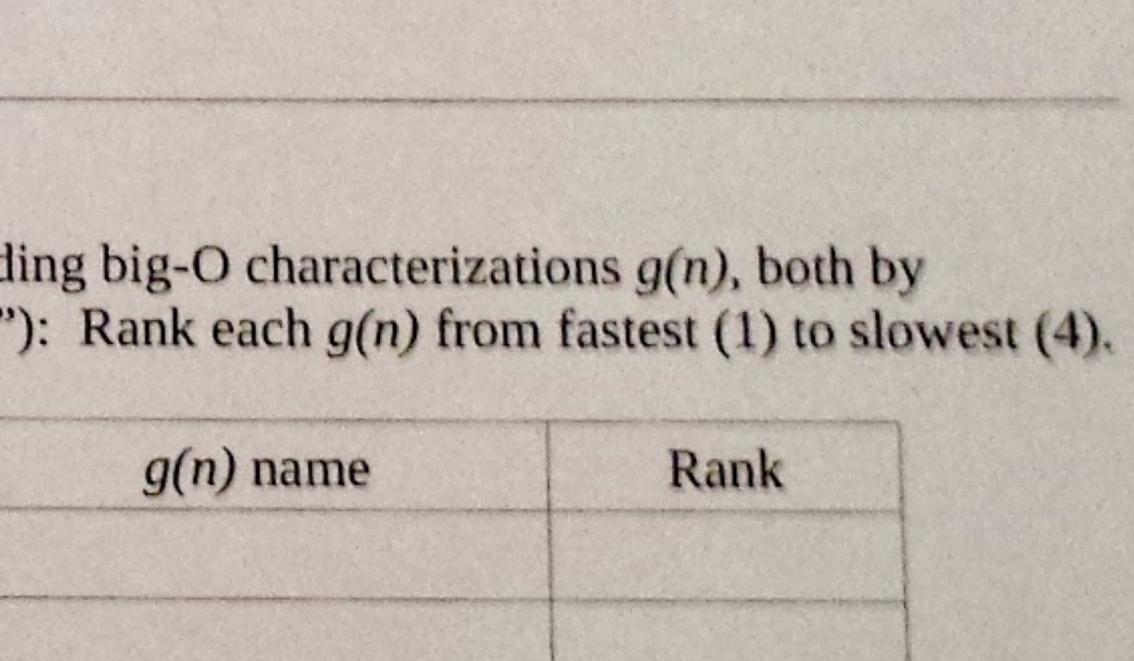
For the following functions *f(n)*, give their corresponding big-O characterizations *g(n)*, both by function (e.g., "*O(n)*") & function name (e.g., "linear"): Rank each *g(n)* from fastest (1) to slowest (4).

f(n)	*g(n)*	*g(n)* name	Rank
$4.5\ n^7 + 2^{n/2} + 79\ n^3$			
479600			
$27\ \sqrt{n}\ + 400\ n \log n + 150{,}201$			
$3\ n^2 + 47\ n + n^5$			

Q5 (2 points)



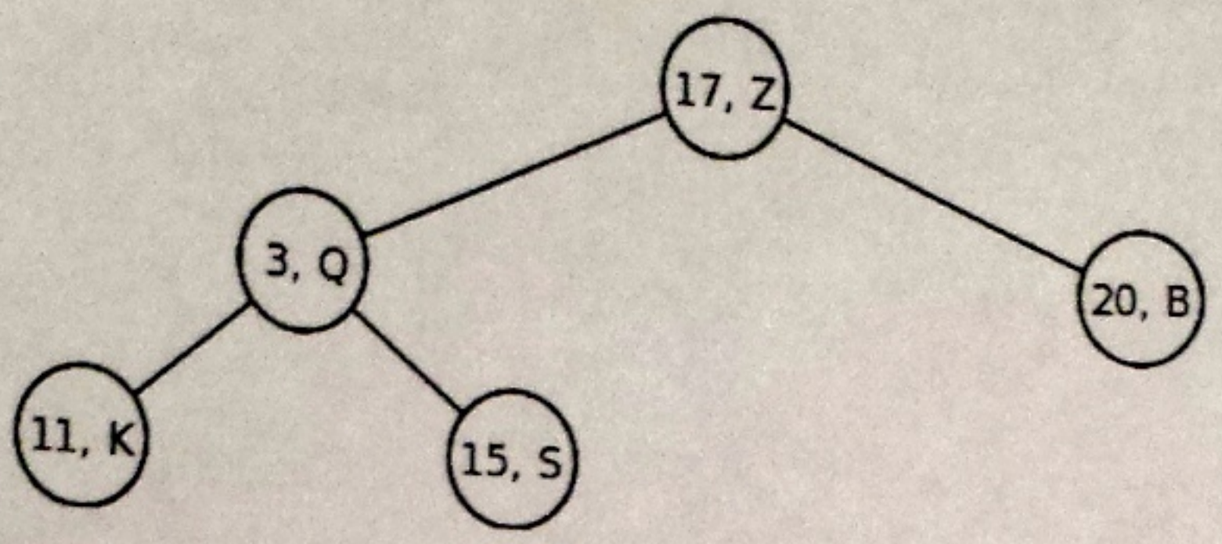
In-order



Post-order

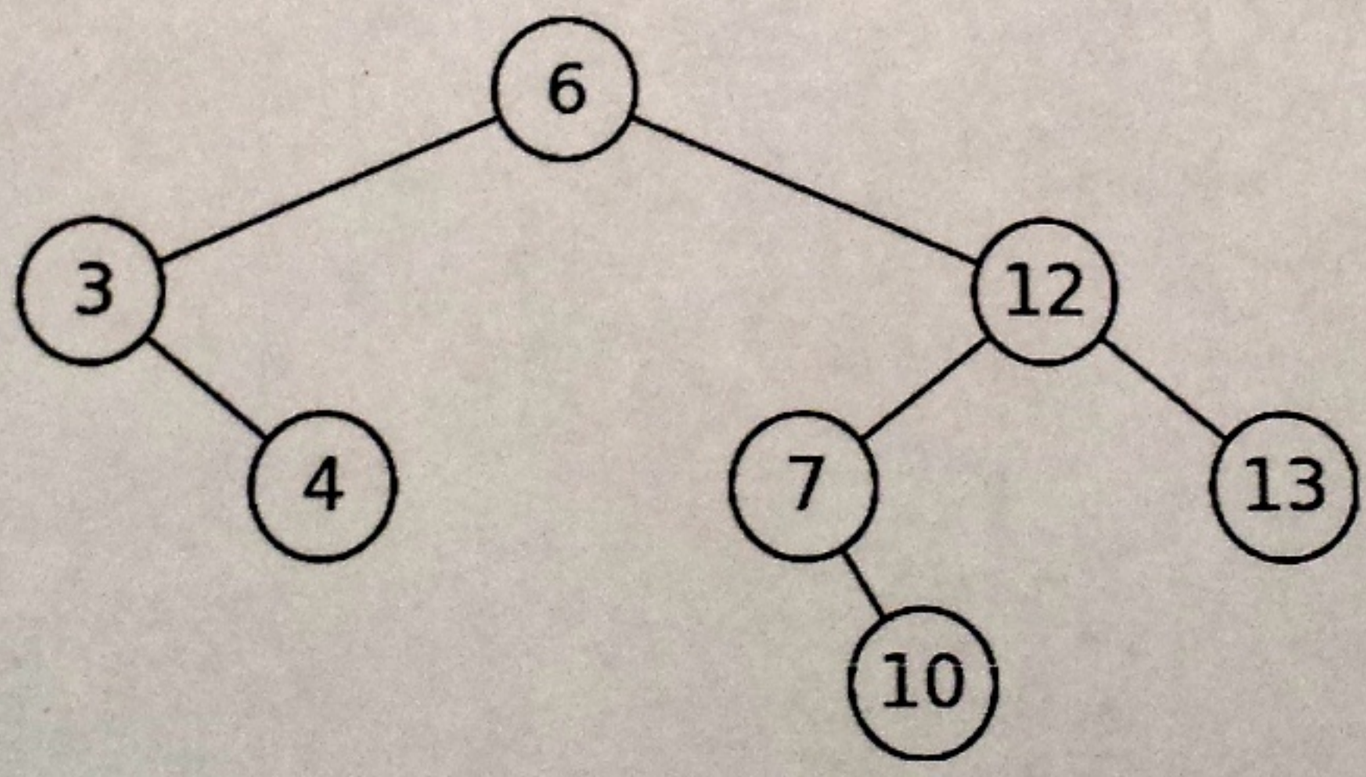
Label the nodes of the identical expression trees above with the order in which they would be visited by the traversal indicated by each caption, starting from 1. What does the expression evaluate to?

Q6 (2 points)



Consider the *k*-d tree data structure in the figure above, where $k = 2$ and the splitting key rotates by level. Explaining briefly why each item goes where it does, draw the tree after each of the following items is inserted in order: (18, M), (2, T), (17, B). Treat the number variable as key 0 and the letter variable as key 1, and use the convention that equal keys go to the right.

Q7 (3 points)



Consider the AVL tree in the figure above. Showing all intermediate steps and balance notation, what would be the resulting trees if each of the following operations were done *independently*?

(a) `remove(6)`

(b) `insert(5)`

(c) `remove(4)`