1. The node with key=6 has just been added to the AVL binary search tree below so that the node key=8 is now unbalanced with left-height minus right-height equal to 2:

```
     8
    / \
   4   9
  / \
3   5
```

Re-balance the tree with the proper rotation or rotations.

2. You have a long list of n records which are kept in memory sorted according to some type of key. New records are being added all the time. What are both the strong and weak points of each of the following data storage options with regards to: lookup, addition of an element, full sort?

   2a. Keep the records in a sorted linked-list.
   2b. Keep the records in a sorted array.
   2c. Keep the records in a binary search tree (not necessarily balanced) implemented with pointers.
   2d. Keep the records in an AVL balanced binary search tree implemented with pointers.
   2e. Keep the records in a heap implemented with an array.
   2f. Keep the records in a B(or B+)-tree of order m where \( \log_{\lceil m/2 \rceil} (n+1)/4 \) is small (i.e. under 10).

3. For each of the following functions classify their complexity as one of: constant, linear, quadratic, cubic, \( \Theta(n \log n) \), polynomial of degree greater than three, exponential, worse than exponential, or none of the previous.

   i. \( h(n) = \sum_{k=1}^{n} 4^k \)
   ii. \( j(n) = \sum_{k=1}^{n} k^2 \)
   iii. \( g(n) = (2n + 2) \log_2 n^2 \)
   iv. \( r(n) = 2(n^2) \)
   v. \( s(n) = n! \)

4. Mathematical complexity requires us to specify two quantities: the size \( n \) of the problem and the basic operation \( c_{op} \), where \( c_{op} \) could be an arithmetic or logical operation, a memory read, a memory swap, or even a function call. Give some examples to show that this is not always straightforward.

5. Suppose that a function \( M \) has initial value \( M(1) = 1 \) and recursion equation

\[
M(n) = M(n-1) + n
\]

Find the value of \( M(n) \) as a formula.

6. Solve the system of linear equations:

\[
\begin{align*}
x + y &= 24 \\
(0.88)x + (1.12)y &= 0
\end{align*}
\]
using Gaussian elimination.

7. Is the closed hash table with linear probing a good choice for lookups if there are lots of both additions and deletions occurring and \( n \) is only slightly smaller than \( m \)? Why or why not?

8. All three sorts: \texttt{merge\_sort}, \texttt{heap\_sort}, and \texttt{quick\_sort} have \( C_{\text{ave}}(n) = \Theta(n \log n) \). What factors would determine which you would use?

9. Suppose that \( n \) data items are indexed by an ordered pair of two unsigned integer keys \((k_1, k_2)\). In choosing a hash function, which of the following would probably be a better choice and why?
   
   9a. Let \( m = 1024 \) and define \( h(k_1, k_2) = 2^{(k_1+k_2)} \mod m \).
   
   9b. Let \( m = 1021 \) and define \( h(k_1, k_2) = (k_1 \times 71 + k_2) \mod m \).

10. If \( n \) data items are put into an open hash table with \( m \) entries in such a way that the keys are distributed as evenly as possible, what will be the expected \( C_{\text{ave}}(n) \) and why?

11. You have a large data set of \( n \) records which is \textbf{too large} to fit in physical memory (so that most of it will reside on the disk – or in virtual memory which has been swapped to the disk). Why are the time constraints in reading information from the records much more serious in this case?

12. In a primarily disk-resident data set we could use a B (or B+) tree to index the data, based on some key. If the number \( n \) of records is \( n = 2^{26} \) (roughly 64,000,000), given that

\[
h \leq \lceil \log_{\lceil m/2 \rceil} (n + 1)/4 \rceil + 1
\]

what should the order \( m \) of the tree be (and the maximum number of keys per node) to ensure that the height of the tree is 4 or less? What role will binary search play in finding a key?

13. Can you see a reason why most operating systems cache frequently accessed disk blocks in the system part of main memory?