1. __True or false: There cannot be a comparison based algorithm for finding the maximum and minimum of an array of n numbers using fewer than $5n/4$ comparisons.

2. For the problem of finding the maximum and the second largest of an array of n numbers, the best algorithm we discussed uses about _____ comparisons.
   (a) $n - 1$
   (b) $n \lg(n)$
   (c) $n + \lg(n)$
   (d) $3n/2$

3. Algorithm A manipulates an array of n items. It requires $\Theta(n^2)$ time. For arrays of 1000 items algorithm A takes about 1 millisecond. On an array of 1,000,000 items how long do you expect algorithm A to take?
   (a) about 17 milliseconds
   (b) about 17 seconds
   (c) about 17 minutes.
   (d) about 17 hours.

4. __True or False: Nobody knows whether or not there might be a comparison based sorting method that uses $\Omega(n)$ comparisons to sort an array of n numbers.

5. __True or False: Nobody knows whether or not there might be a comparison based sorting method that uses $O(n)$ comparisons to sort an array of n numbers.

6. __True or False: The recurrence $T(n) = T(n/2) + O(1)$, for $n \geq 2$ with $T(1) = 1$ has the solution $T(n) = O(\lg(n))$. 
7. Professor Moriarty has four algorithms whose runtime functions (call them $T_2, T_3, T_4, T_5$) satisfy the four recurrences

$$T_k(n) = \begin{cases} 
1, & \text{for } n = 1, \\
kT_k(n/k) + O(n), & \text{for } n > 1,
\end{cases}$$

for $k = 2, 3, 4, 5$.

Which best describes the situation?

(a) $T_k(n) = O(n)$, for each $k$ in 2..5.
(b) $T_k(n) = O(n \lg(n))$, for each $k$ in 2..5.
(c) $T_k(n) = O(n \lg(5))$, for some $k$ in 2..5.
(d) none of the above

8. Devious Professor Moriarty has developed four additional algorithms whose runtime functions (we’ll also call them $T_2, T_3, T_4, T_5$, satisfy the four recurrences,

$$T_k(n) = \begin{cases} 
1, & \text{for } n = 1, \\
kT_k(n/2) + O(n), & \text{for } n > 1,
\end{cases}$$

for $k = 2, 3, 4, 5$.

Which best describes the situation?

(a) $T_k(n) = O(n)$, for each $k$ in 2..5.
(b) $T_k(n) = O(n \lg(n))$, for each $k$ in 2..5.
(c) $T_k(n) = O(n \lg(5))$, for some $k$ in 2..5.
(d) none of the above

9. Which is true of the recurrence

$$T(n) = \begin{cases} 
1, & \text{for } n = 1, \\
7T(n/2) + O(n^2), & \text{for } n > 1
\end{cases}.$$

(a) $T(n)$ describes the runtime of a matrix multiplication algorithm.
(b) $T(n) = O(n \lg(7))$.
(c) $T(n) = O(n^{2.81})$.
(d) all of the above

10. Which recurrence relation describes the runtime of FFT (the fast Fourier transform).

(a) $T(n) = 2T(n/2) + O(n)$
(b) $T(n) = 2T(n/2) + n/2T(2)$
(c) both of the above
(d) none of the above

11. When the FFT is called on $n = 2^k$ points, a primitive root of unity, $\omega$, is used. This $\omega$ must be a primitive $m$th root of unity, for which $m$?

(a) $m = k$
(b) $m = n$
(c) $m = n/2$
When the FFT is called on \( n \) points a primitive root of unity, \( \omega \), is used. For certain \( X \) and \( Y \), the two recursive calls work with \( X \) of the points and with the \( Y \)-th power of \( \omega \).

(a) \( X = n/2 \) points, \( Y = 2 \) is exponent of \( \omega \)
(b) \( X = n \) points, \( Y = 1/2 \) is exponent of \( \omega \)
(c) \( X = n/2 \) points, \( Y = 1/2 \) is exponent of \( \omega \)
(d) \( X = n \) points, \( Y = 2 \) is exponent of \( \omega \)

13. True or false: The recurrence \( T(n) = 2T(n/2) + O(n^2) \) describes the worst case number of comparisons used in mergesort.

14. True or false: The leaves of a full binary tree are all on the same level.

15. A graph is a DAG if, with respect to it’s depth first search forest, it has no ______ edges.
   (a) tree
   (b) forward
   (c) back
   (d) cross

16. RSA public key encryption is hard to crack, because
   (a) Modular arithmetic is a total mystery
   (b) testing for primality is difficult
   (c) factoring large composite numbers is difficult
   (d) The Fermat test is fooled by some non-primes called Carmichael numbers.

17. Let \( n \) be the number of days since the beginning of the earth’s rotation until your date of birth. What is \( 3^n \mod 2 \)?

18. What is \( 2^{2^{2^2}} \mod 7 \)?
19. (6 points)

(a) A directed acyclic graph can be linearized. Explain what this means.

(b) Draw a linearization of this graph: \( V = \{A, B, C, D\} \), \( \text{nbr}[A] = C, D; \text{nbr}[B] = A, D; \text{nbr}[C] = D; \text{nbr}[D] = \text{null} \).

(c) How may a linearization of a DAG be constructed from the results of depth first search that computes previsit and postvisit times for each node?
20. (10 points)

(a) Using \texttt{modmul}(a, b, m) which computes \(a \times b \mod m\), write \texttt{modexp}(a, e, m), which computes \(a^e \mod m\), for \(n\) bit numbers \(a, e, m\). Your program should use no more than \(2n\) calls to modmul, that is, no more than \(2n\) multiplications or squarings \(\mod m\).

(b) Up to big-O, what is the runtime of your \texttt{modexp}?

21. (5 points)

(a) Write \texttt{mergesort}(a[0..n-1]). You may use (without defining it) \texttt{merge}(a[0..i-1], b[0..j-1]), which merges the two sorted array segments \(a\) and \(b\), putting the result in \(c[0..i+j-1]\).

(b)
22. (10 points)

(a) Let \( f(x) = \sum_{i=0}^{n-1} f_i x^i \) be a \( n \) term polynomial. Explain how \( f(a) \) and \( f(-a) \) can be computed by combining \( f_e(a^2) \) and \( f_o(a^2) \), where \( f_e \) and \( f_o \) are the even part and the odd part of \( f \), respectively.

(b) When \( \omega \) is an \( n \)-th root of unity, what power of \( \omega \) is equal to \( -\omega \)?

23. (10 points)

(a) Write \( \text{explore}(G, v) \), which does a depth first search of the graph \( G = (V, E) \) beginning at vertex \( v \). You may use the adjacency list representation, with \( \text{nbr}[u] \) being the list of neighbors of \( u \) in \( G \).

(b) Up to big-O, what is the run time of \( \text{explore} \)?

24. Have a good spring break.