1) Given the binary tree below, where \( n \) is the number of nodes and \( m \) the number of edges, transforming it into a valid max heap in the best running time possible requires:

- a. Finding the max value and swapping it with the root. It takes \( O(n) \)
- b. Traversing all the tree, adding the elements into a valid max heap. It takes \( O(n \log n) \)
- c. Calling the upheap function on each one of the leaves. It takes \( O(\log n) \)
- d. Ordering the tree and then swapping the rightmost leave with the root. It takes \( O(n \log n) \)
- e. Running DFS starting on the root, swapping elements when required. It takes \( O(n+m) \)
2) We can detect quickly if a graph is not connected by:

a. Running DFS once and checking if we visited all the vertices of the graph
b. Running Dijkstra’s algorithm and checking if we got all the vertices in the path
c. Multiplying the adjacency matrix of the graph by itself and checking if no cell in the matrix is zero
d. Checking if all MST starting on each vertex in the graph contains all the vertices in the graph
e. None of the above

3) Which of the following is always unique for a given graph?

a. Euler cycle
b. Minimum Spanning Tree
c. Adjacency matrix
d. BFS and DFS traversal
e. None of the above

4) Given $\Sigma = \{a, b, d, k\}$ and the Last-Occurrence function below. It is true that:

<table>
<thead>
<tr>
<th></th>
<th>c</th>
<th>a</th>
<th>b</th>
<th>r</th>
<th>d</th>
<th>k</th>
</tr>
</thead>
<tbody>
<tr>
<td>L(c)</td>
<td>13</td>
<td>11</td>
<td>12</td>
<td>8</td>
<td>6</td>
<td></td>
</tr>
</tbody>
</table>

a. We cannot infer any substring of the pattern
b. Character d is not in the text
c. The pattern finishes with the substring "bra"
d. Options a, b and c are correct
e. Options b and c are correct

5) Which of the following patterns generate the Failure-Function below:

<table>
<thead>
<tr>
<th>j</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>F(j)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>
6) Is the following a valid restructuring transformation when inserting a key into a 2-3 tree?

a. Yes, since it cannot exist nodes with 3 elements
b. No, since key b is the one needed to be pushed up instead of key c

c. Yes, since each node now has 2 keys and 3 children
d. No, since subtrees T4 and T5 have to be merged
e. None of the above

7) Consider an array implementation of a stack. You are asked to add the following additional operations to the stack API:

- Min(): returns the minimum element currently in the stack
- Search(x): returns true if the element x is somewhere in the stack, and false otherwise
- SearchTop3(x): returns true if element x in one of the three elements in the top three stack positions, and false otherwise

We know that Push() and Pop() run in O(1). You can use additional constant space for the implementation. Then, it is true that:

a. Min() can run in O(1). Search(x) can run in O(1). SearchTop3(x) can run in O(1)
b. Min() can run in O(n). Search(x) can run in O(1). SearchTop3(x) can run in O(n)
c. Min() can run in O(n). Search(x) can run in O(n). SearchTop3(x) can run in O(n)
d. Min() can run in O(1). Search(x) can run in O(n). SearchTop3(x) can run in O(n)
e. Min() can run in O(n). Search(x) can run in O(n). SearchTop3(x) can run in O(1)
8) There is a common command in Linux called “tree”, which works as a sort of recursive ls command. The output of the command “tree temp/” is shown below

```
  temp
  ├── dir1
  │   ├── a.txt
  │   ├── b.txt
  │   │   ├── dir1a
  │   │   │   └── c.txt
  │   └── dir1a
  │       └── c.txt
  ├── dir2
  │   └── d.txt
  └── dir3
       ├── e.txt
       └── f.txt
```

What type of tree traversal did the tree command do in this example?

a) Preorder  
b) Postorder  
c) Inorder  
d) Sorted order  
e) Eulerian order

9) Imagine that you are asked to write a program that works with a very large and sparse undirected and unweighted graph. It has roughly 500 million vertices, but each vertex only has at most 5 edges adjacent to it. You will be writing this to run on a machine that has 32GB of RAM available. Which of the following would NOT be an appropriate graph representation to use?

a) Adjacency list  
b) Edge list  
c) **Adjacency matrix**  
d) Vertex list  
e) All of the above are well-suited for this situation

10) Which of the following is not a property of a red-black tree?

a) A red node cannot have a red child  
b) The root is black  
c) **All leaves are equidistant from the root**  
d) All leaves are black  
e) None of the above
11) How many unique minimum spanning trees does the following graph have?

![Graph Image]

a) 1  
b) 2  
c) 4  
d) 6  
e) 10

12) After running Dijkstra’s algorithm on a graph with some source vertex S, you notice that the distances to some vertices are listed as infinity. What can you say about the original graph?
   a) The graph is bipartite  
b) The graph has an Eulerian tour  
c) The graph is a forest  
d) The graph is complete  
e) The graph is disconnected

13) A good hashing function for a hash table has which of the following properties?
   a) It maps to the indices uniformly  
b) It is invertible  
c) It is deterministic  
d) b and c  
e) a and c

14) Which of the following must be true to be able to use Dijkstra’s algorithm?
   a) The graph is acyclic
b) The graph is directed
c) The graph has a unique MST
d) The graph has nonnegative weights
e) All of the above

15) If you wanted to create a resizeable array with $O(1)$ amortized insert time, which of the following strategies should you be sure to implement?
   a) Resize by a fixed amount when the array becomes full
   b) Resize by multiplying by a constant factor when the array becomes full
   c) Only allow the user to insert at the end of the array
   d) Do not allow random deletions from the array
   e) All of the above

16) What is the lookup time for a random element in a max heap, given no additional data structures?
   a) $O(1)$
   b) $O(\log n)$
   c) $O(n)$
   d) $O(n \log n)$
   e) $O(n^2)$

17) Which of the following sorting algorithms uses a divide and conquer approach?
   a) Bubblesort
   b) Insertion sort
   c) Shell sort
   d) Selection sort
   e) Quick sort

18) What is the asymptotic relationship between $n^k$ and $c^n$? Assume $k \geq 1$ and $c > 1$.
   a) $n^k$ is $O(c^n)$
   b) $n^k$ is $\Theta(c^n)$
   c) $n^k$ is $\Omega(c^n)$
   d) b and c
   e) a, b, and c
19) Which recurrence relation describes the following pseudocode? (It does not matter what the pseudocode actually does).

```c
void nonsense(int* array, int start, int end) {
    int len = end - start;
    if (len <= 2) return;

    nonsense(array, start + len / 3, end);
    nonsense(array, start, end - len / 3);

    for (int i = end-1; i >= start; i--) {
        for (int j = start; j < end; j++) {
            array[j] = 2 * array[i];
        }
    }
}
```

a) \( T(n) = 2 \ T(n / 3) + 2n; \ T(2) = 1 \)
b) \( T(n) = 2 \ T(n / 2) + n^2; \ T(2) = 1 \)
c) \( T(n) = T(n / 3) + n; \ T(2) = 1 \)
d) \( T(n) = 2 \ T(2 \ n / 3) + n^2; \ T(2) = 1 \)
e) None of the above

20) Decompress the following sequence using LZ78:

```
(0, l) (0, u) (0, k) (0, e) (0, _) (1, u) (0, c) (3, _) (1, i) (7, k) (0, s) (5, l) (0, a) (3, e) (11, '')
```

a) luke_luck_likes_lakes
b) lukes_duck_likes_lakes
c) **luke_luck_lucks_lakes**
d) lucks_duck_lucks_lakes
e) none_of_the_above

21) From a complete graph with number of edges \( E \) and number of Vertices be \( N \), by removing how many edges, we can construct a spanning tree?

a) \( E-N+1 \)
b) \( N-E+1 \)
c) \( N+E-1 \)
d) \( E-N-1 \)
22) If we implement Recursive Squaring naively by calling the subproblem for half the exponent \((n/2)\) TWICE instead of once (i.e., by not storing it to a variable), the time complexity would change to
   a) Exponential in terms of \(n\)
   b) Linear in terms of \(n\)
   c) Would not change
   d) Logarithmic in terms of \(n\)
   e) None of the above

23) Knapsack problem that weights are positive integers in power of 2: \(1, 2, \ldots 2^n\), can be solved by
   a) Greedy algorithm
   b) A combination of greedy and dynamic programming
   c) By Dynamic programming in linear time
   d) Can be solved not more efficiently than general knapsack problem
   e) None of the above

24) Suffix trie of input string consisting of \(n\) times character \(a\) (aaa...a) can be constructed in \(O(n)\) time using \(O(n)\) space using
   a) Without any compression
   b) No algorithm has been found
   c) Using Huffman trie encoding
   d) Using Run-Length encoding
   e) Either Huffman or Run-Length encoding

25) The fastest implementation of Dijkstra with adjacency matrix (without list) runs in
   a) \(O(V^2)\)
   b) \(O(V * E)\)
   c) \(O((E + V)\log V)\)
   d) \(O(V \log V)\)
   e) \(O(V^3)\)
26) What is the time complexity of the following Kmerge algorithm?
   a) O(n)
   b) O(nlogk)
   c) O(n^2)
   d) O(nk^2)

kMerge(int k, followed by k lists):
    Create k pointers pointing to the start of the k lists
    Open an output file
    While not done:
        Write the smallest of the items pointed to to the output file
        increment the pointer that was just used

28) Suppose you correctly implement searching within a large sorted file in O(logN). If the file
    has size N and said search runs in t seconds, in how many seconds do you expect the same
    program to run on file of size N^2 :
   a) 2 * t
   b) t^2
   c) t * logt
   d) t ^ 1.5
   e) 4 * t

29) Let G be an undirected connected graph with distinct edge weight. Let emax be the edge
    with maximum weight and emin the edge with minimum weight. Which of the following
    statements is false?
   a) Every minimum spanning tree of G must contain emin 
   b) If emax is in a minimum spanning tree, then its removal must disconnect G
   c) No minimum spanning tree contains emax
   d) G has a unique minimum spanning tree
30) Consider the directed graph shown in the figure below. There are multiple shortest paths between vertices S and T. Which one will be reported by Dijkstra’s shortest path algorithm? Assume that, in any iteration, the shortest path to a vertex v is updated only when a strictly shorter path to v is discovered.

a) SDT
b) SBDT
c) SACDT
d) SACET

31) Let G be a directed graph whose vertex set is the set of numbers from 1 to 100. There is an edge from a vertex i to a vertex j iff either j = i + 1 or j = 3i. The minimum number of edges in a path in G from vertex 1 to vertex 100 is

a) 4
b) 7
c) 23
d) 99
32) Which of the following standard algorithms is not Dynamic Programming based.
   a) Bellman–Ford Algorithm for single source shortest path
   b) Floyd Warshall Algorithm for all pairs shortest paths
   c) 0-1 Knapsack problem
   d) Prim’s Minimum Spanning Tree

33) Which of the following is true about Huffman Coding.
   a) Huffman coding may become lossy in some cases
   b) Huffman Codes may not be optimal lossless codes in some cases
   c) In Huffman coding, no code is prefix of any other code.
   d) All of the above

34) Consider the undirected graph below:

Using Prim’s algorithm to construct a minimum spanning tree starting with node A, which one of the following sequences of edges represents a possible order in which the edges would be added to construct the minimum spanning tree?

   a) (E, G), (C, F), (F, G), (A, D), (A, B), (A, C)
   b) (A, D), (A, B), (A, C), (C, F), (G, E), (F, G)
   c) (A, B), (A, D), (D, F), (F, G), (G, E), (F, C)
   d) (A, D), (A, B), (D, F), (F, C), (F, G), (G, E)
35) In a directed graph, where all edges have integral weights of zero or 1, we can find shortest path with which variants of the following algorithm:
   a) Bellman-ford algorithm
   b) Dijkstra
   c) BFS
   d) a and b
   e) All the above

Note: variation to BFS is to use double ended queue or simply two queues; by first push neighbors that are zero apart from the current node and then nodes that are one distance apart.

36) Consider the following Red-Black tree:

Suppose we want to insert the element 14. Which of the following is true? For reference:

a) We must perform a recoloring only
b) We must perform a left rotation only
c) We must perform a right rotation only
d) We must perform a left rotation followed by a right rotation
e) The element can be inserted without any further action
37) Here is an array of 10 elements:

\[
8 \ 2 \ 7 \ 4 \ 1 \ 3 \ 0 \ 6 \ 9 \ 5
\]

What is the order of the elements after THREE iterations of the outer loop of selection sort, when sorting from low to high?

a) 0 1 2 4 7 3 8 6 9 5  
b) 2 4 7 8 1 3 0 6 9 5  
c) 2 1 3 0 4 6 5 7 8 9  
d) 0 1 2 3 4 6 7 8 5 9  
e) None of the above

38) Suppose we have a hash table of size N = 9, with hash function \( h(k) = k \mod N \). Collision resolution is done by way of quadratic probing, where the \( i^{th} \) probe for a key \( k \) is \( h(k, i) = (h(k) + i^2) \mod N \). Insert the following elements into the table in this order:

14  4  22  8  6  18

Which option below represents the resulting table?

a)

<table>
<thead>
<tr>
<th>Index</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key</td>
<td>18</td>
<td>4</td>
<td>22</td>
<td>14</td>
<td>6</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

b)

<table>
<thead>
<tr>
<th>Index</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key</td>
<td>8</td>
<td>18</td>
<td>4</td>
<td>14</td>
<td>6</td>
<td>22</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

c)

<table>
<thead>
<tr>
<th>Index</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key</td>
<td>18</td>
<td>4</td>
<td>14</td>
<td>22</td>
<td>6</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

d)

<table>
<thead>
<tr>
<th>Index</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key</td>
<td>18</td>
<td>14</td>
<td>22</td>
<td>4</td>
<td>6</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

e) None of the above
39) Which of the following sequences corresponds to a BFS of the below graph, starting from element 5, in which the adjacent nodes are visited in numerical order?

- a) 5 7 1 2 8 3 9 4 6 0
- b) 5 1 3 8 7 2 9 4 0 6
- c) 5 1 3 9 4 0 6 8 2 7
- d) 5 1 7 2 3 8 9 4 0 6
- e) None of the above

40. Which of the following sorting algorithms guarantees to run in O(n log n) time?
   - a) Quicksort
   - b) Heapsort
   - c) Insertionsort
   - d) Selectionsort

41. Which of the following sorting algorithm does not always run in O(n log n) time?
   - a) Quicksort
   - b) Heapsort
   - c) Mergesort
   - d) None of the above

42. Which of the following is an in-memory sorting algorithm with O(n log n time complexity)?
   - a) Heapsort
   - b) Mergesort
   - c) Randomized Quicksort
   - d) Binary tree sort
43. How many strongly connected components are there in the following graph?

![Graph](image)

a) 1  
b) 3  
c) 4  
d) 5  
e) None of the above

44) Consider the following graph and find the number of different topological ordering of the vertices of the graph:

![Graph](image)

a) 1  
b) 2  
c) 4  
d) 6  
e) 8

Answer: 6; a-b-c-d-e-f, a-d-e-b-c-f, a-b-d-c-e-f, a-d-b-c-e-f, a-b-d-e-c-f, a-d-b-e-c-f

45) Consider the DAG with V = \{1,2,3,4,5,6\}, shown below:

![Graph](image)
Which of the following is **NOT** a topological ordering:

a) 1,2,3,4,5,6  
b) 1,3,2,4,5,6  
c) 1,3,2,4,6,5  
d) 3,2,4,1,6,5  
e) All of the above are in topological order

**Answer:** 1 appears after 2 and 3.

46) Which vertex may be removed to make the following graph no longer biconnected?

a) 4  
b) 7  
c) 6  
d) 8  
e) None of the above
47) What is the Big-Oh expression for the following function in C++:

```c++
int enigma(int arr[], int size, int target) {
    int foo = 0;
    int bar = size - 1;
    int uga = 0;
    while(foo <= bar) {
        uga = (foo + bar) / 2;
        if(target == arr[uga]) {
            return uga;
        } else if(target > arr[uga]) {
            foo = uga + 1;
        } else { //target < arr[uga]
            bar = uga - 1;
        }
    } //while
    return -1;
}
```