Instructions:

- Write your name and student number in the space provided
- Please check that your exam is complete. It should have 19 pages in total
- The examination has a total of 100 marks
- This is an open textbook/lecture notes exam. You can consult your notes/textbook but you cannot talk/communicate with other students. Exam/homework solutions are not allowed during the exam.
- University regulations do not allow you to leave the room in the last 30 min of the exam.
- When you are done, raise your hand and we will pick your exam up
<table>
<thead>
<tr>
<th>TOTAL:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part 1</td>
</tr>
<tr>
<td>Part 2:</td>
</tr>
<tr>
<td>14 /4</td>
</tr>
<tr>
<td>15 /4</td>
</tr>
<tr>
<td>16 /5</td>
</tr>
<tr>
<td>17 /4</td>
</tr>
<tr>
<td>18 /5</td>
</tr>
<tr>
<td>19 /5</td>
</tr>
<tr>
<td>20 /5</td>
</tr>
<tr>
<td>21 /5</td>
</tr>
<tr>
<td>22 /5</td>
</tr>
<tr>
<td>23 /8</td>
</tr>
<tr>
<td>24 /8</td>
</tr>
<tr>
<td>25 /3</td>
</tr>
</tbody>
</table>
PART 1: MULTIPLE CHOICE

Instructions: Enter your answers on the Scantron sheet. We will not mark answers that have been entered on this sheet. Unless stated otherwise, assume all logarithms are base 2. Each multiple choice question is worth 3 marks.

Answers: EBBBBDAEAECDDDE

1. Consider the algorithm Multiply(A,n) below. A is an array of size n storing integer values. What is the best characterization of the best and worst case asymptotic time complexity of the following algorithm?

   Algorithm Multiply(A,n)
   result = 0
   for i = 0 to \( \frac{n}{2} \) do
     j = 0
     while A[j] < 0 and j < \( \frac{n}{2} \) do
       result = result + A[i]*A[j]
       j = j + 1
   return result

   (a) Best case \( O(1) \), worst case \( O(n) \).
   (b) Best case \( O(\log n) \), worst case \( O(n) \).
   (c) Best case \( O(n) \), worst case \( O(n) \).
   (d) Best case \( O(1) \), worst case \( O(n^2) \).
   (e) Best case \( O(n) \), worst case \( O(n^2) \).

2. Suppose \( T \) is a binary search tree storing \( n \) entries. Which of the following statements is true?

   (a) Inserting into \( T \) is \( O(1) \) in the best case and \( O(\log n) \) in the worst case.
   (b) Inserting into \( T \) is \( O(1) \) in the best case and \( O(n) \) in the worst case.
   (c) Inserting into \( T \) is \( O(\log n) \) in the best case and \( O(\log n) \) in the worst case.
   (d) Inserting into \( T \) is \( O(\log n) \) in the best case and \( O(n) \) in the worst case.
   (e) Inserting into \( T \) is \( O(n) \) in the best case and \( O(n) \) in the worst case.
3. Let $T$ be a full binary tree with root $r$ where each node $v$ has a variable $\textit{var}$ that can be accessed with $v.\textit{var}$. Consider the algorithm below. What does the algorithm return when called on the root $r$? Here $v.\textit{left}$ and $v.\textit{right}$ are the left and the right children of $v$, respectively.

\begin{verbatim}
Algorithm TreeAlg($v$)
    if $v$ is external
        $v.\textit{var} = 0$
        return 0
    else
        $v.\textit{var} = 1 + \max\{v.\textit{left}.\textit{var}, v.\textit{right}.\textit{var}\}$
        $l = \text{TreeAlg}(v.\textit{left})$
        $r = \text{TreeAlg}(v.\textit{right})$
        if $v.\textit{left}.\textit{var} < v.\textit{right}.\textit{var}$
            return $l + r + 1$
        else
            return $l + r$
\end{verbatim}

(a) the number of nodes $v$ in the tree s.t. the left child of $v$ is at a smaller depth than the right child of $v$

(b) the number of nodes $v$ in the tree s.t. the left subtree of $v$ has a smaller height than the right subtree of $v$

(c) the number of nodes $v$ in the tree s.t. the left subtree of $v$ has less nodes than the right subtree of $v$

(d) the number of nodes $v$ in the tree s.t. the left subtree of $v$ has less internal nodes than the right subtree of $v$

(e) the number of nodes $v$ in the tree s.t. the left subtree of $v$ has less external nodes than the right subtree of $v$

4. Let $T$ be a (2,4) tree storing 200 entries (recall that leaves do not store anything). What is the smallest possible height of $T$?

(a) 3

(b) 4

(c) 5

(d) 6

(e) 7
5. For which data structure the order of insertion does not matter, i.e. the resulting data structure is identical regardless of the order the elements were inserted?

(a) Unsorted sequence.
(b) Sorted sequence.
(c) Heap.
(d) AVL tree.
(e) (2-4) tree.

6. Consider the following directed graph. Which of the following is a valid order of node visitation during depth first search (DFS) traversal?

(a) A D B C F E
(b) B F E A D C
(c) D B F C E A
(d) C F D A E B
(e) E A D B C F
7. Consider the following undirected graph. Which of the following is not a valid order of node visitation during breadth first search (BFS) traversal?

(a) A D E C B F
(b) B D F C E A
(c) C E B F A D
(d) D A B F E C
(e) E C B A F D

8. Suppose we use Dijkstra’s algorithm to find the shortest paths from vertex A in the graph below. In which order the vertices could be inserted into the ‘blue cloud’?

(a) A D E C B F
(b) A F D E C B
(c) A E C D B F
(d) A E D C F B
(e) A E D B C F
9. Suppose we use Prim’s algorithm on the weighted graph below. In which order the vertices could be inserted into the ‘blue cloud’?

(a) A F E B C D
(b) A F D E C B
(c) A E F B D C
(d) A E F D B C
(e) A E B F D C

10. Suppose that you are implementing a DFS graph traversal, but the class implementing graph nodes does not support node marking. You decide to use an AVL-tree to implement node marking. You start with an empty tree. To mark a node as visited, you insert it into the tree. A node is unmarked if it is not in the tree, and marked if it is in the tree. Assume graph is represented as an adjacency list, you mark only nodes, not edges. Let $m$ be the number of edges, $n$ the number of nodes. What is the best characterization of the worst-case asymptotic complexity of DFS in this case?

(a) $O(n + m)$
(b) $O(n \log n)$
(c) $O(n \log n + m \log n)$
(d) $O(n^2 + m \log n)$
(e) $O(n^2 + m)$
11. Let \( G \) be an undirected, weighted, connected graph. Let \( w(u, v) \) denote the weight of edge \((u, v)\). What does the following algorithm return when called on a graph node \( u \)? Assume that before the call to \( \text{GraphAlg}(G, u) \) all vertices are unmarked.

**Algorithm** \( \text{GraphAlg}(G, u) \)

mark \( u \)

\( c = 0 \)

for each edge \((u, v)\) incident on \( u \) do

if \( v \) is not marked then

\( c = c + w(u, v) + \text{GraphAlg}(G, v) \)

return \( c \)

(a) Sum of all edge weights.
(b) Sum of all edge weights plus the number of nodes.
(c) Sum of the degree of each node multiplied by the weight of the edges incident on that node.
(d) Sum of weights of the discovery edges.
(e) Sum of weights of the back edges.

12. Let \( G \) be strongly connected directed graph with 100 nodes. Let \( G^* \) be the transitive closure of \( G \). How many edges does \( G^* \) have?

(a) 99
(b) 2475
(c) 4950
(d) 9900
(e) 10000

13. Let \( A \) be an array storing \( n \) elements. Which of the following statements is true?

(a) If \( A \) is almost sorted, selection sort should be used.
(b) Mergesort is always the fastest algorithm in practice to sort \( A \).
(c) Quicksort is always the fastest algorithm in practice to sort \( A \).
(d) If \( A \) is so large that it does not fit into the main memory, insertion sort should be used.
(e) If \( A \) is so large that it does not fit into the main memory, merge sort should be used.
14. [4 marks] Insert 6 into the following (2,4) tree. Use exactly the same algorithm as we studied in class. Draw the intermediate trees.

SOLUTION:
15. [4 marks] Delete 1 from the following (2,4) tree. **Use exactly the same algorithm as we studied in class.** Draw the intermediate trees.

\[
\begin{array}{c}
3 \\
\downarrow \\
1 & 5 \ 7 \\
\downarrow & \downarrow \\
0 & 2 & 4 & 6 & 8 \\
\end{array}
\]

**SOLUTION:**

\[
\begin{array}{c}
3 \\
\downarrow \\
2 \ 5 \ 7 \\
\downarrow \\
0 & 4 & 6 & 8 \\
\end{array}
\]

\[
\begin{array}{c}
3 \\
\downarrow \\
5 \ 7 \\
\downarrow \\
0 \ 2 & 4 & 6 & 8 \\
\end{array}
\]

\[
\begin{array}{c}
5 \\
\downarrow \\
3 \ 7 \\
\downarrow \\
0 \ 2 & 4 & 6 & 8 \\
\end{array}
\]
16. [5 marks] Let $G$ be an undirected, weighted, connected graph. Let $w(u,v)$ denote the weight of edge $(u,v)$. Let $n$ denote the number of vertices and $m$ the number of edges in $G$. Suppose $G$ is implemented with an adjacency matrix. What is the best characterization of asymptotic worst-case complexity of algorithm $\text{GraphAlg}$ below, as a function of $n$, or $m$, or both $n$ and $m$? Assume that before the call to $\text{GraphAlg}(G,u)$ all vertices are not marked.  
Show all the work you do.

\underline{Algorithm $\text{GraphAlg}(G,u)$}

\begin{algorithm}
\hspace{1em} for each edge $(u,v)$ incident on $u$ do \\
\hspace{2em} $w(u,v) = w(u,v) + 1$ \\
\hspace{2em} mark $u$ \\
\hspace{1em} for each edge $(u,v)$ incident on $u$ do \\
\hspace{2em} if $v$ is not marked then \\
\hspace{3em} $\text{GraphAlg}(G,v)$
\end{algorithm}

SOLUTION: We count the total number of invocation of $\text{GraphAlg}(G,u)$ and how much time is spent at each invocation, not including the recursive call. $\text{GraphAlg}(G,u)$ is invoked exactly one time for each vertex $u$. At each invocation for vertex $u$, time spent is $1 + 2 \cdot n + 3 \cdot \deg(u)$, as indicated above. Adding it up over all vertices $u$, we get

\[
\sum_{u \in V} (1 + 2 \cdot n + 3 \cdot \deg(u)) = \sum_{u \in V} 1 + \sum_{u \in V} 2 \cdot n + \sum_{u \in V} 3 \cdot \deg(u) = n + 2n^2 + 6m,
\]

which is $O(n^2 + m)$. If the graph is simple, it is $O(n^2)$ since the number of edges is less than $n^2$ in a simple graph.
17. [4 marks] Let $G$ be an undirected weighted graph shown below. Show the clouds that vertices belong to after 4 iterations of the Kruskal’s minimum spanning tree algorithm.

SOLUTION:
18. [5 marks] Compute the topological ordering for the following graph $G$ and list the vertices in the order consistent with the topological ordering, starting with the vertex with smallest topological number. Is your topological sorting unique?

SOLUTION:

19. [5 marks] Draw a connected graph with 6 vertices where both DFS and BFS can visit nodes in the same order if started from an appropriate vertex. Mark this vertex with \( s \).

SOLUTION: Several possibilities, here is one:

![Graph Diagram]

20. [5 marks] The mergesort tree of a sequence of integers is a binary tree in which each node contains a sequence of integers passed as a parameter to a call to mergesort (the sequence contents before and after execution), and each child node represents a recursive call to mergesort. This is exactly as in the lecture notes. Draw the mergesort tree of the following sequence: 8,5,2,0,6,4,5,1. Note that you can draw just one final tree, you do not have to show all the steps in the tree expansion as we did in class.

SOLUTION:

![Mergesort Tree Diagram]
For the following 2 questions, you can use any of the algorithms and data structures that we have studied in class. Describe your algorithm in words mentioning any data structures/algorithms we have studied in class. You do not have to write pseudo code. For example, suppose the question is as follows:

Given an array $A$ of size $n$, find the median element of $A$ ($x$ is a median element if half of the elements of $A$ are larger than or equal to $x$, and half of the elements of are smaller than or equal to $x$). The worst case time complexity of your algorithm should be $O(n \log n)$.

You answer might be: sort $A$ using heap sort and return the element of $A$ at rank $n/2$.

21. [5 marks] Suppose you have two text files, $A$ and $B$, and you need to find all the words that occur twice as often in file $A$ than in file $B$. For example, if file $A$ contains words `run, run from me me` and file $B$ contains words `run to me`, the output should be `run me`. Describe how you would solve this problem in $O(n \log n)$ worst case time, where $n$ is the total number of words in both text files.

SOLUTION: Create two empty AVL trees storing entries, with string key, and integer value. Let us call them $treeA$ and $treeB$. Read words from file $A$ and insert them in $treeA$ as follows. The first time the word is read (not in $treeA$), insert it into $treeA$ with integer value set to 1. If the word is read again (already in $treeA$), increase its integer value by 1. After file $A$ is read, all words that occur in file $A$ are stored in $treeA$ with their counts. Do the same for file $B$ but inserting into $treeB$. Now get an iterator over $treeB$. For each word in the iterator, search for it in $treeA$. If present, compare integer values. If the value in $treeA$ is twice larger than value in $treeB$, output this word.
22. [5 marks] Suppose that you are developing a flight reservation application for a travel agency. You have a list of all cities and all the flights between these cities. As a part of the application, the travel agent wants the following feature: given any city $A$, he wants to be able to find all the other cities which can be reached from $A$ with no more than $k$ plane changes, where $k$ is a positive integer. How do you implement this as efficiently as possible?

SOLUTION: Create a graph with each city represented by a node. If there is a flight between two cities, put the directed edge between the corresponding tree nodes. Given node for city $A$, run directed $BFS$ with level marking. Return all the cities corresponding to the nodes whose levels are less than or equal to $k$. 


23. [8 marks] Write an algorithm \textit{Unique}(A, B, n, m) that takes as an input array \(A\) of size \(n\) and array \(B\) of size \(m\). Assume that both arrays store integers and are sorted in increasing order. You can also assume that \(A\) has no repeating elements and also that \(B\) has no repeating elements. Your algorithm should count and return the number of elements of \(A\) that do not occur in \(B\) plus the number of elements of \(B\) that do not occur in \(A\). For example, if \(A = \{1, 3, 5, 7, 8\}\) and \(B = \{3, 4, 7, 9\}\), your algorithm should return 4 since there are four elements that occur just in \(A\) or just in \(B\) (namely 1, 4, 5, 8, 9). The running time should be \(O(n+m)\), and your algorithm should be in-place. Recall that in-place means that you can declare only a few additional variables, you cannot use any container data structures.

\textbf{SOLUTION:}

\begin{verbatim}
Algorithm Unique(A, B, n, m)

    count = 0
    i = 0
    j = 0
    while i < n and j < m do
            i = i + 1
            count = count + 1
        else if A[i] = B[j] do
            i = i + 1
            j = j + 1
        else
            j = j + 1
            count = count + 1
    return count
\end{verbatim}
24. [8 marks] Let $G$ be a weighted undirected connected graph. Write, in pseudo-code, an algorithm $\text{sumCrossEdges}(G)$ that performs BFS of $G$ and returns the sum of weights of edges that get labeled as ‘cross’ edges.

SOLUTION:

```
Algorithm $\text{sumCrossEdges}(G)$

$sum = 0$

for $e \in G.edges()$
  mark $e$ unvisited

for $v \in G.vertices()$
  mark $v$ unvisited

$s = G.vertices().getFirst()$
$Q = \text{new empty sequence}$
$Q.insertLast(s)$
mark $s$ visited

while $Q$ is not empty
  $v = Q.first()$

  for all $e \in G.incidentEdges(v)$ do
    $w = \text{opposite}(v, e)$
    if $w$ is not visited
      mark $w$
      set label of $e$ to DISCOVERY
      $Q.insertLast(w)$
    else if $e$ is not visited
      set label of $e$ to CROSS
      $sum = sum + e.getWeight()$

return $sum$
```

25. [3 marks] Compute the best characterization of the asymptotic time complexity of your algorithm above in the worst-case. Assume that the graph is implemented as adjacency list. Assume \(n\) is the number of vertices and \(m\) is the number of edges.

SOLUTION: First for loop runs in \(O(m)\) time, second for loop in \(O(m)\) time. Next four line take constant time. While loop is executed \(O(n)\) times, one time for each vertex \(v\) in the graph. The time to execute for loop for vertex \(v\) is \(1 + \text{deg}(v)\) with adjacency list graph representation. Summing up over all graph vertices \(v\), we get \(n + m\) running time for the while loop, so the total time is \(O(n + m)\).