DURATION: 3 HOURS

Department Name & Course Number: Computer Science COMP 3804B
Course Instructor: Michiel Smid

Authorized memoranda: NONE

Students MUST count the number of pages in this examination question paper before beginning to write, and report any discrepancy to the proctor. This question paper has 12 pages (not including the cover page).

This examination question paper MAY NOT be taken from the examination room.

In addition to this question paper, students require:

- an examination booklet: no
- a Scantron sheet: no
Student Name:
Student Number:

<table>
<thead>
<tr>
<th>Question</th>
<th>Maximum Number of Marks</th>
<th>Marks Received</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$11 \times 3 = 33$</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>100</td>
<td></td>
</tr>
</tbody>
</table>

Instructions

1. All questions must be answered on this examination paper. You may use both sides of the answer sheets for answers. There are some extra sheets attached at the end of this examination paper.

2. This is a closed book examination.

3. Please try not to ask questions during the exam. If you find a question to be ambiguous or unclear, then make, and state, whatever assumptions you feel are necessary; your mark will then be partly based on the reasonableness of your assumptions.

4. Please answer to the point and do not write everything you know on the topic. Substantial marks will be deducted if your answer is not precise.
Question 1: (3 marks for every correct answer)
Answer TRUE or FALSE:

(A) \(1^3 + 2^3 + 3^3 + 4^3 + \ldots + n^3 = O(n^4)\).

Answer:

(B) You are given two algorithms that solve the same problem. The running time of algorithm I satisfies the recurrence \(T(n) = T(n - 1) + n\), whereas the running time of Algorithm II satisfies the recurrence \(T(n) = 2 \cdot T(n/2) + n\).

The running time of algorithm I is smaller than that of algorithm II for large values of \(n\).

Answer:

(C) Given a sequence of \(n\) numbers, a min-heap for these numbers can be constructed in \(O(n)\) time.

Answer:

(D) Heapsort is an asymptotically optimal comparison-based sorting algorithm.

Answer:

(E) Dynamic programming is a technique to solve \(\text{NP}\)-complete problems in polynomial time.

Answer:

(F) Given a max-heap storing \(n\) numbers, and given a number \(x\), we can determine in \(O(\log n)\) time if \(x\) is stored in this max-heap.

Answer:

(G) In \(O(n)\) time, we can find the 100-th smallest number in a set consisting of \(n\) real numbers.

Answer:

(H) Let \(G\) be a connected graph with positive weights on its edges. There cannot be more than one minimum spanning tree of \(G\).

Answer:

(I) Let \(G\) be a connected graph with positive weights on its edges, and let \(u\) and \(v\) be two vertices of \(G\). Dynamic programming cannot be used to compute the longest path between \(u\) and \(v\).

Answer:

(J) \(\text{NP}\) stands for “non-polynomial time”.

Answer:
(1.k) The following algorithm computes a minimum spanning tree of an undirected connected graph $G = (V, E)$ with positive weights on its edges:

$m = \text{the number of edges in } E$;
put the edges of $E$ in an arbitrary order $e_1, e_2, \ldots, e_m$;
$T = \emptyset$;
for $i = 1$ to $m$
do if $T \cup \{e_i\}$ does not contain a cycle
then $T = T \cup \{e_i\}$
endif
endfor;
return $T$

Answer:

**Question 2: (15 marks)** Let $A$ and $B$ be two sets of real numbers, each set containing $n$ elements. Describe an algorithm (in plain English) that returns true if there is an element $a \in A$ and an element $b \in B$ such that $|a - b| \leq 1$. If such elements $a$ and $b$ do not exist, then the algorithm returns false. The running time of your algorithm must be $O(n \log n)$. Explain why your algorithm is correct and why its running time is $O(n \log n)$.

(You may use any result that was discussed in class as part of your solution.)
Question 3: (5 marks) Give the definition of $L \leq_p L'$, i.e., $L$ is polynomial-time reducible to $L'$.

Question 4: (7 marks) What is wrong with the following argument?

If

- $L'$ is an NP-complete language,
- the language $L$ is in NP, and
- $L \leq_p L'$,

then the language $L$ is NP-complete.

It suffices to explain your answer in plain English.
Question 5: (20 marks) The set cover problem is defined as follows:

\[ \text{SetCover} := \{ (B, S_1, S_2, \ldots, S_m, K) : \]
\[ B \text{ is a set of size } n; \ S_1, S_2, \ldots, S_m \text{ are sets with } \bigcup_{i=1}^{m} S_i = B; \ K \text{ is an integer; } \]
\[ \text{there exists a subset } I \subseteq \{1, 2, \ldots, m\} \text{ of size } K, \text{ such that } \bigcup_{i \in I} S_i = B \}. \]

The (0-1)-integer programming problem with \(K\) ones is defined as follows:

\[ \text{IntProgKOnes} := \{ (A, K) : \]
\[ A \text{ is an integer } n \times m \text{ matrix all of whose entries are in } \{0, 1\}; \ K \text{ is an integer; } \]
\[ \text{there exists a binary vector } x \text{ of length } m \text{ with exactly } K \text{ ones, such that } Ax \geq \mathbf{1} \]
\[ \text{(componentwise)} \}, \]

where \(\mathbf{1}\) denotes the vector of length \(n\), all of whose entries are equal to 1.

Prove that \(\text{IntProgKOnes}\) is \(\text{NP}\)-complete. (You may use the fact that \(\text{SetCover}\) is \(\text{NP}\)-complete. You do not have to show that \(\text{IntProgKOnes}\) is in \(\text{NP}\).)
Question 6: (20 marks) You are given two binary strings $A = a_1a_2\ldots a_m$ and $B = b_1b_2\ldots b_n$. A common substring of $A$ and $B$ is a binary string that occurs consecutively both in $A$ and in $B$.

Give a dynamic programming algorithm (in pseudocode) that computes, in $O(mn)$ time, a longest common substring of $A$ and $B$. Argue why your algorithm is correct.

For example, for the input strings $A = 0001010111$ and $B = 01010101000$, the output will be the string 010101.

Hint: Define $L[i, j]$ to be the length of a longest common substring of $A$ and $B$ that ends at $a_i$ and $b_j$. Observe that $L[i, j] = 0$ if $a_i \neq b_j$. 