Answer THREE questions with at least ONE from each Section.

Section 1..Answer at least ONE question from this section

1. a. i. Briefly explain what is meant by an algorithm taking time in the order of T (where T is some function).

ii. Give four reasons why one needs to exercise caution when describing an algorithm taking time in the order of T.

iii. What is an optimization problem?

[10 marks]

b. Consider a recursive program that loops through the input to eliminate half the input each cycle. As an example, consider a program that searches a sorted list. Assume the algorithm takes time in the order of T. This recurrence can be described by the following equation for $n \geq 2$, and $T(1) = 0$:

$$T(n) = T(n/2) + 1$$

What mathematical function is T equivalent to? Give a proof and explanation with your answer.

[10 marks]

c. i. Is Quicksort based on a greedy, divide and conquer, or dynamic programming, approach? Explain why.

ii. What are the essential similarities and differences of how the Quicksort and Mergesort algorithms work?

iii. Give an example of an array containing 9 items that would be handled inefficiently by the Quicksort algorithm. Explain how the inefficiency would arise.

[13 marks]

[Total 33 marks]

TURN OVER

2. a. i. Draw a tree corresponding to the following heap:

$$\langle a, h, g, f, j, c, d, r, t, b \rangle$$

ii. For the tree generated in part (i), give the sequence of nodes visited in a depth-first search.

iii. For the tree generated in part (i), give the sequence of nodes visited in a breadth-first search.

[6 marks]

b. i. Define the travelling salesperson's problem (also called the travelling salesman's problem).

ii. Explain why the travelling salesperson's problem is a type of optimization problem.

iii. Briefly explain how a greedy algorithm can be defined for getting a solution (not necessarily an optimal solution) to the travelling salesperson's problem.

[16 marks]

c. Give the pseudocode for a version of depth-first search based on a stack data structure. Use the usual operations Push and Pop. Explain any other data structures or functions that you require.

[9 marks]

[Total 33 marks]

3. a. i. What is the principle of optimality?

ii. What features of an algorithmic problem make dynamic programming a useful approach?

iii. What is the key difference between the greedy and dynamic programming approaches to designing algorithms?

[15 marks]

b. What is the purpose of each of the following algorithms: (i) Floyd's algorithm; and (2) Warshall's algorithm.

[4 marks]

c. Consider the following adjacency matrix for a graph with the nodes $\{a, b, c, d, e\}$.

	a	b	c	d	e
a	0	7	∞	1	∞
b	2	0	∞	∞	4
c	∞	6	0	∞	∞
d	∞	1	∞	0	∞
e	∞	∞	7	∞	0

i. Draw a graph corresponding to this adjacency matrix.

ii. Use this adjacency matrix as an input to Floyd's algorithm. Give the intermediate and final matrices that would be generated by the algorithm. Ensure that you present your answer clearly.

[14 marks]

[Total 33 marks]

Section 2..Answer at least ONE question from this section

4. Let A consist of all instances of the axiom schemes

   1. $(\phi \to \neg\neg\phi)$
   2. $(\neg\neg\phi \to \phi)$
   3. $(\theta \to (\phi \to \theta))$
   4. $((\theta \to (\phi \to \psi)) \to ((\theta \to \phi) \to (\theta \to \psi)))$
   5. $((\neg\theta \to \neg\phi) \to (\phi \to \theta))$
   6. $((\phi \to \theta) \to (\neg\theta \to \neg\phi))$

   a. Define the 'provability relation' $\vdash$. Explain what we mean when we write $\Gamma \vdash \phi$, where Γ is a set of sentences and ϕ is a formula. State clearly which inference rule(s) are to be used.

   [9 marks]

   b. Let us restrict to the propositional logic of formulas using the connectives $\neg$ and $\to$ only. Explain what we mean when we say that the provability relation $\vdash$ (from previous part) is (i) sound and (ii) complete.

   [8 marks]

   c. Write down a proof that $\neg(p \to q) \vdash \neg q$.

   [16 marks]

[Total = 33 marks]

CONTINUED

5. a. Write down a formal definition of the set of formulas of propositional logic.

[5 marks]

b. Explain what it means when we say that two propositional formulas are *equivalent*.

[5 marks]

c. Explain how structured induction works for propositional formulas.

[8 marks]

d. Prove, by structured induction, that for every propositional formula ϕ there is an equivalent formula ϕ^* such that ϕ^* does not use the connectives $\neg, \wedge, \vee$, but it may use the propositional constant $\bot$. That is, ϕ^* uses the connectives $\bot$ and $\rightarrow$ only.

[15 marks]

[Total = 33 marks]

6. a. Define *disjunctive normal form* and *conjunctive normal form* for propositional logic.

[6 marks]

b. Use tableaus to convert each of the following into disjunctive normal form.

1. $(p \to (q \to r))$
2. $((p \to q) \to r)$
3. $\neg(\neg(p \to q) \to \neg(q \to \neg p))$

[9 marks]

c. Explain how a tableau can be used to convert a propositional formula into conjunctive normal form.

[9 marks]

d. Use a tableau to convert the following formula into conjunctive normal form.

$$\neg(p \vee \neg(q \to ((\neg p \wedge q) \to r)))$$

[9 marks]

[Total = 33 marks]

END OF PAPER