This is a closed book test. You may not refer to any resources.

This is a 50 minute test.

Please write your answers in ink. Pencil answers will be marked, but will not be reconsidered after the test papers have been returned.

The test will be marked out of 50.

<table>
<thead>
<tr>
<th>Question 1</th>
<th>/8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Question 2</td>
<td>/10</td>
</tr>
<tr>
<td>Question 3</td>
<td>/30</td>
</tr>
<tr>
<td>Question 4</td>
<td>/2</td>
</tr>
<tr>
<td><strong>TOTAL</strong></td>
<td>/50</td>
</tr>
</tbody>
</table>
Question 1 (8 Marks)

When implementing a Branch and Bound algorithm, we typically use a min-heap to store the partial solutions. Why?

Solution: In each iteration of the algorithm we need to find the partial solution with the lowest lower bound. Suppose we have $k$ partial solutions. If we store the partial solutions in an unsorted array or list, we would have to search the entire set of partial solutions to find the smallest, which will take $O(k)$ time. If we keep the set of partial solutions in a sorted array or list, we can find the smallest in $O(1)$ time but every time we add a new partial solution it takes $O(k)$ time to keep the set in sorted order. However, with a min-heap we can find the smallest in $O(1)$ time (and then fix the heap in $O(\log k)$ time), and adding each new partial solution takes only $O(\log k)$ time.

Since $k$ can be exponentially large with respect to $n$, any operation that requires $O(k)$ time is infeasible. However, even if $k$ is approximately $2^n$, $O(\log k)$ is only $O(n)$.

Thus we use a min-heap because it allows us to do the essential operations in feasible time.

Marking: Their discussion does not need to give all the comparisons to other data structures.

"It’s faster": 4/8
"It’s faster for finding the minimum value": 6/8
"It lets us do the operations in logarithmic time": 8/8
Question 2 (10 Marks)

Let P be a partial solution with lower bound $l$ and upper bound $u$.

Suppose we extend P one step and get a new partial solution $P'$ with lower bound $l'$ and upper bound $u'$.

(a) Is it possible that $u' < l$? Explain your answer.

Solution: No. The lower bound $l$ is based on guaranteed costs in $P$. No expansion of $P$ can avoid these costs, so we must have $l \leq l' \leq u'$.

Marking:
- "No": 2/5
- "No, because upper bounds have to be higher than lower bounds": 3/5
- "No, because $l$ is based on guaranteed costs": 5/5
- "Yes": 1/5
- "Yes, because ...": 2/5

(b) Is it possible that $l' > u$? Explain your answer.

Solution: Yes. The value $u$ is an upper bound on the best expansion of $P$. This particular expansion $P'$ may be a very bad choice with very high guaranteed costs.

Marking:
- "Yes": 3/5
- "Yes, because $P'$ may be a bad extension": 5/5
- "No": 1/5
- "No, because ...": 2/5
Question 3: (30 marks)

T. Ronald Dump has just been elected Permanent Secretary General of the United Nations and you are in charge of evacuating the few remaining members of the League of Rational People to their new home on Titan. You have a large fleet of one-way ships at your disposal, and your task is to assign all the people to ships in such a way as to minimize the total launch cost.

There are \( n \) ships and \( m \) people. You may assume that \( n \geq m \). You are not required to launch all the ships (only the ones you assign people to).

Each ship \( S_i \) has a maximum load value \( m_i \) that specifies the maximum number of kilograms the ship can carry.

Each ship \( S_i \) has a cost/kg value \( c_i \) for launching, so if ship \( S_i \) is carrying a load of \( k_i \) kilograms, the cost to launch the ship is \( k_i \cdot c_i \).

Each person \( P_i \) has a mass \( g_i \) which is their mass in kilograms. You may assume that no person has a mass that is greater than the capacity of any ship.

So the goal is to minimize \( C \), where \( C = \sum_{i=1}^{n} k_i \cdot c_i \) where \( k_i \) is the sum of the masses of the people assigned to ship \( S_i \) with the constraints:
\[
    k_i \leq m_i \quad \forall \ i \quad \text{and} \quad \text{each person is assigned to a ship}
\]

You have been instructed to create a Branch & Bound algorithm to solve this problem.

(question continues on next page)
In this question, correct answers that demonstrate careful thought will receive more marks than answers that are correct but trivial.

(a) [5 marks] Describe the sequence of decisions your algorithm will make

Solution: I will iterate through the list of passengers, assigning each one to a ship.

Marking: Other solutions are just as good, such as “Iterate through the list of ships, choosing a set of passengers for each one”. Give full marks for anything that is plausible.

(b) [8 marks] Describe how you will compute the initial value of the Global Upper Bound. (If you decide to sort the data, explain your reasons for doing so.)

Solution: I will sort the passengers into descending order by mass, and sort the ships into ascending order by cost/kg value. My reason for doing this is to try to pay as little as possible to launch the passengers with the greatest mass. I will construct a feasible solution by iterating through the sorted list of passengers, assigning each passenger to the first ship that has room for the passenger.

Marking: Sorting makes this Greedy heuristic possible. They don’t have to sort on the same criteria as I do, but they should sort ships, passengers or both.

“Assign people to ships (or ships to people) without sorting anything”: 5/8
“Sort the values then assign people to ships (or ships to people)”: 8/8

(c) [3 marks] Describe how you will compute Cost So Far for each partial solution

Solution: I will compute the total launch cost for all passengers who have been assigned to ships, which is just the sum of the product of the passenger’s mass and the cost/kg value for the ship they are assigned to.
(d) [8 marks] Describe how you will compute Guaranteed Future Cost for each partial solution

Solution: For each remaining passenger, determine the first ship that has room for the passenger. The product of this passenger’s mass and this ship’s cost/kg value is a guaranteed future cost because assigning this passenger to any other ship will cost at least as much. The sum of these products is the guaranteed future cost.

Marking: their answer should incorporate some analysis of the remaining passengers and ships. For example, they could multiply the sum of the masses of all remaining passengers by the lowest cost/kg of all the ships. This is less effective than the method outlined above but it is acceptable. This is the hardest part of the test, and some leniency can be applied in grading it. If they show or state an understanding of Guaranteed Future Cost (ie, a value x such that all expansions of the partial solution cost \( \geq x \)), they should get at least 4/8

(e) [6 marks] Describe how you will compute Feasible Future Cost for each partial solution

Solution: I would use exactly the same technique as was described for computing the initial value of the Global Upper Bound, applied to the remaining passengers.

Marking: This question is straightforward, but it is based on their answer for computing the Global Upper Bound. If their answer for that part doesn’t work and they use it again here, they should not lose marks for it twice.

If they show that they know what Feasible Future Cost means but they cannot compute it, give at least 3/6.
Question 4: (2 marks)

Choose the answer that best completes this sentence:

Properly designed Branch and Bound algorithms ....

(a) ... are guaranteed to run in polynomial time.

(b) ... will always find an optimal solution.

(c) ... can only be applied to minimization problems.

(d) ... require quantum computing.

Solution:  (b)