**Exercises**

**1.1-1**

**Describe your own real-world example that requires sorting. Describe one that**

**requires finding the shortest distance between two points.**

Real-world sorting example: I would image a having over 150 songs in a music application requires sorting of the songs, probably from the most played to the least played. In that one can easily access his/her favourite song than stressing while looking it up in the music list.

Shortest distance example: The best I can think of is using google maps to look up for directions from one’s location to one’s destination. Normally a variety of routes are given including the directions with the shortest distance. And as humans we tend to prefer the directions with the shortest distance to optimize time and resources.

**1.1-2**

**Other than speed, what other measures of efficiency might you need to consider in**

**a real-world setting?**

Measures of efficiency in real-world setting could refer to the standards used to evaluate the performance of a solution. Therefore, other standards include

* Scalability (ability for a solution to work on a various data size without depreciation in performance)
* Ease of Implementation
* Resource utilization (how much resources does the solution consume)

**1.1-3**

**Select a data structure that you have seen, and discuss its strengths and limitations.**

***Linked Lists***

Strengths:

* Dynamic Size: Linked lists can easily grow or shrink in size without requiring reallocation or resizing of the entire structure.
* Efficient Insertions/Deletions: Adding or removing elements, especially in the middle of the list, is O(1) if the node reference is known.

Limitations:

* Sequential Access: Accessing elements requires traversal from the head of the list, resulting in O(n) time complexity for lookups.
* Complex Implementation: Managing pointers and ensuring memory safety can complicate implementation and debugging.

**1.1-4**

**How are the shortest-path and traveling-salesperson problems given above similar?**

**How are they different?**

**Def:**

Shortest-path problem seeks the shortest path or minimal distance between two points (nodes) in a graph, which can have weighted or unweighted edges.

Traveling salesperson problem (TSP) involves finding the shortest possible route that visits a set of nodes/cities exactly once and returns to the starting node/city.

**Similarities:**

* Graph-based: Both problems are formulated on graphs, where nodes represent locations and edges represent connections between them.
* Optimization: The goal in both cases is to find the best possible path based on a certain criterion (e.g., shortest distance, minimum cost).

**Differences:**

Endpoints:

* Shortest-path: Finds the shortest path between two specific nodes in the graph.
* Traveling-salesperson: Finds the shortest possible route that visits every node in the graph exactly once and returns to the starting node.

Scope:

* Shortest-path: Focuses on finding the optimal path between a pair of nodes.
* Traveling-salesperson: Aims to find the optimal route that covers all nodes in the graph.

Complexity:

* Shortest-path: Can be solved efficiently using algorithms like Dijkstra's algorithm or the A\* search algorithm.
* Traveling-salesperson: Is generally NP-hard, meaning there is no known efficient algorithm to solve it for large instances.

**1.1-5**

**Suggest a real-world problem in which only the best solution will do. Then come**

**up with one in which “approximately” is the best solution is good enough.**

Problem which requires the best solution:

In most cases designing a plane’s flight path to prevent plane collusions, minimize fuel consumption and avoid restricted airspace like the Russian airspace. Any slight deviation can lead to a catastrophe; therefore, the best solution is the most preferable.

Problem that can take an approximate solution:

Registration of first-year students into campus halls while other students also report. Due to limited room availability and time constraints, finding the perfect match for every student isn’t feasible. Instead, an approximate approach, such as assigning rooms based on availability, priority rules, and flexible allocations, ensures that most students get acceptable accommodations quickly. This approach balances efficiency, fairness, and practicality without needing a perfect match for each student.

**1.1-6**

**Describe a real-world problem in which sometimes the entire input is available**

**before you need to solve the problem, but other times the input is not entirely**

**available in advance and arrives over time.**

Online retailers (like Jumia) need to *manage their inventory efficiently* to ensure products are available when customers place orders. Inventory management involves keeping track of stock levels, forecasting demand, and replenishing items to avoid stockouts or overstocking.

Entire Input Available Before Solving:

* At the start of a sales event or a new season, the retailer has complete inventory data, including stock levels of all items. This allows the retailer to plan restocking and promotional strategies based on the full initial input.

Input Arrives Over Time:

* As sales occur, inventory data is updated in real time. Unexpected factors like sudden spikes in demand, supplier delays, or returns happen continuously, requiring the system to adjust stock levels and availability information on the fly.