1.1-1. Describe your own real-world example that requires sorting. Describe one that requires ending the shortest distance between two points.

Sorting Example: A real-world scenario requiring sorting is organizing a list of students' names alphabetically for registration purposes at a school. The list needs to be sorted so that it's easier for the administrators to search and locate individual names quickly.

Shortest Distance Example: A real-world scenario that requires finding the shortest distance is navigating through a city using GPS. If you want to drive from point A to point B, the navigation system will calculate the shortest route to reduce travel time and fuel consumption.

1.1-2. Other than speed, what other measures of efficiency might you need to consider in a real-world setting?

Memory usage: In some scenarios, minimizing memory usage is critical, especially for devices with limited resources.

Scalability: Efficiency in handling larger datasets is important. For example, a solution might work fast with a small dataset but degrade as the data grows.

Energy consumption: Especially relevant in battery-powered devices where energy-efficient algorithms are needed.

User experience: In some cases, providing a result in a consistent and predictable amount of time is more important than being the fastest.

1.1-3. Select a data structure that you have seen, and discuss its strengths and limitations.

How are the shortest-**Selected Data Structure: Hash Set**

- **Strengths:**
  - Fast lookups, insertions, and deletions (average O(1) time complexity).
  - Great for checking membership or eliminating duplicate entries.
- **Limitations:**
  - Cannot maintain order of elements.
  - Not suitable when you need to access elements in a specific sequence.
  - Relatively inefficient in terms of memory compared to other structures like arrays or lists due to hash function overhead.

1.1-4. path and traveling-salesperson problems given above similar? How are they different?

Similarities: Both the shortest-path problem and the traveling salesperson problem (TSP) deal with finding optimal routes between points or nodes in a graph.

Differences: The shortest-path problem is concerned with finding the quickest or least costly route from a starting point to a single destination, while the TSP is about finding the most efficient way to visit multiple locations and return to the starting point.

Shortest-path algorithms (like Dijkstra's) focus on optimizing a route between two specific points, whereas TSP requires visiting all points, making it a more complex, NP-hard problem.

1.1-5. Suggest a real-world problem in which only the best solution will do. Then come up with one in which <approximately= the best solution is good enough.
Best Solution Example: In the design of a pacemaker, only the best solution will do. The device needs to regulate heartbeats with precise timing to ensure the patient's safety, so there is no room for approximation or error.

Approximate Solution Example: In food delivery services, such as Glovo or Café Javas, an approximate solution is often good enough. The algorithm doesn't need to find the absolute fastest route every time; as long as the food arrives in a reasonable time frame, it is considered satisfactory.

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.
Problem with Entire Input Available: Scheduling final exams for students at a university, where the complete list of exams and available time slots are known in advance. In this scenario, the input (exam data) is available before the scheduling process begins.

Problem with Partial Input Available Over Time: Online shopping order processing: as new orders come in, they need to be processed and shipped. The input (orders) arrives over time, and the system needs to continuously handle new orders while managing existing ones.