OPIFUDRIRA TIMOTHY

S23B23/086

B24775

Assignment for Design and Analysis of Algorithms

1.1-1: Real-world examples:

- **Sorting example:** Imagine you're a DJ like DJ Flixton, organizing your song library. You might need to sort your music tracks by BPM (beats per minute) to create a seamless flow between tracks in a mix. Sorting them from slow to fast can help in building energy throughout a set.
- **Shortest distance example:** You're navigating to a new gig venue. You use a GPS app to find the shortest route between your current location and the destination, minimizing travel time and fuel.
-

1.1-2: Other measures of efficiency:

- **Memory usage:** Especially when dealing with large datasets, it's essential to consider how much memory an algorithm uses.
- **Scalability:** How well does the algorithm perform as the size of the data grows?
- **Power consumption:** In embedded systems, efficiency in terms of energy consumption is crucial.
- **Ease of implementation and maintenance:** An efficient algorithm is also one that is not overly complex and is easy to debug and modify.
-

1.1-3: Data structure example:

- **Arrays:**
    - **Strengths:** They allow for fast access to elements by index, and they are great for storing data when the size is fixed.
    - **Limitations:** Inserting or deleting an element can be slow because it might require shifting elements. Also, their size is static, so if you need a flexible structure, arrays might not be the best choice.

1.1-4: Similarities and differences between shortest-path and traveling-salesperson problems:

- **Similarities:** Both problems aim to minimize the total distance or cost between points. They require analyzing multiple paths to find the optimal one.
- **Differences:** The shortest-path problem focuses on finding the shortest route between two specific points. The traveling salesperson problem (TSP) requires finding the shortest path that visits a set of points exactly once and returns to the starting point. TSP is more complex because it involves visiting all points rather than just connecting two.

1.1-5: Real-world problem examples:

- **Best solution required:** A hospital emergency room might use an algorithm to prioritize patients based on the severity of their condition. In this case, only the best solution (choosing the most critical patient first) is acceptable.
- **Approximate solution acceptable:** When compressing video files for streaming, a small loss of quality is acceptable as long as the file size is significantly reduced. You don't need perfect compression, just one that balances quality and size.

1.1-6: Problem where input availability varies:

- **Available input:** When planning a road trip with a fixed schedule and destinations, you have all the data (locations, distances, stops) beforehand.
- **Unavailable input:** If you're delivering food orders, new orders might come in while you're already delivering, requiring real-time updates to your route. The input (new orders) arrives over time, so you need to adjust on the go.