

### 1.1-1

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

#### ● Sorting:.

When I have to-do list each task with a number represent its priority and I want to sort them to work in most important tasks first.

#### ● Shortest path:.

I want to go to my university with the shortest road so that I will not be late.

### 1.1-2

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

I might consider how much storage and hardware technologies this algorithm require.

### 1.1-3

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

Arrays is very basic data structure. It can access and modify any element very efficiently but its size is fixed.

### 1.1-4

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

How are they different?

## shortest path

wants to find the shortest path between two points

have efficient algorithms

## traveling-sales person

wants to find shortest path between set of points

doesn't have efficient algorithm yet

### 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.

## ● needs best solution:.

when we enter a password, computers hash it and match it with the correct one. If even the difference is only one character this is considered wrong so we need the exact solution.

## ● approximate solution is enough:.

when you use a map app and want to find a fast route. if the app needs 10 minutes to find a route faster than the one he find in 0.5 seconds by 10 seconds, it's better to choose the second route even if it's not the fastest one.

### 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.

In competitive programming contests, we sort the participants according to number of problems solved and need time. If the contest is still running the input arrives overtime but when it ends we already have the entire input.