Module 1 Prompt Discussion

Noboru Hayashi

1. The design value in writing an ADT is to diverse implementation and functionality. The users can use the program with the knowledge about the input and output of the program, but knowing how it was implemented is not required. Also ADT is abstract, so when the programmers or creators design a new ADT, as a primary approach, they can start having high-level designing based on the functionality, instead of coding the specific implementations.

2. We measure the runtime and space consumption as indicators of the program's complexity, mainly using the information of input data size or number.

3. An upper bound for a function is applied when the input number is over specific value. So it's possible that if the input value is small for the function, the output exceeds the upper bound.

4. Upper bounds indicate the worst case time or space the algorithms use while handling and operating data and functions. Lower bounds are for the optimistic calculation. I think these indicators can be important while choosing one algorithm from multiples.

5. If the algorithm is simple and uses constant resources, I think there's no need to consider upper and lower bounds.

6. Since there are multiple resources the algorithm consumes such as time and memory space. For different algorithms the usage of the resources are different. Plut, these resource consumption also depends on the size or the scale of the problem. So there's no "best" algorithm, need to compare and pick one for each question or task.

7. Even though memory is cheap nowadays, the amount of data is also increasing. So if the algorithm or program is not using memory resources efficiently due to high space complexity, it would lead to more disk access. Reading and writing to disk is slower than to memory, so I think eventually high complexity in space would impact on the runtime performance.