How would you describe ...

Linked Lists

A structure where each element has at least two parts.

Parts

- 1. the value of the element
- 2. The link (also know as a pointer) which
  - either points to the next element,
  - or points to an indicator for the end of the list (e.g. `NULL`)

Extra

- Since each list element points to the next, a linked list doesn't need sequential memory space.
- Retrieving any but the first item in the list, requires traversing the list from the beginning, to find that item.

Stacks

Work exactly how they sound: You can only add *(push)* items to the "top" (end) of the stack. You only remove *(pop)* items from top (end) of the stack.

- Properties
  - Newly added items are dealt with first: called Last-In-First-Out order.
  - To get to the oldest items (items that were *pushed* first), you must remove one by one, from the top of the stack.
  - The pointer always points the most recent item in the stack.

Hash Maps

Match or *map* a key, like "Elon_Musk"; to a value, like "Tesla".

A hash map is like a sandwich. Between the key and value, is the hash-value/hash function. Every key-value pair, has its own _hash_ between it.

- Ideal situation
  - Every `key` maps to exactly one `value`.
  - Keys in the hash map are unique.
- Example (Python)
  - ```
    { "Elon_Musk": "Tesla",
      "Steve_Ballmer": "Microsoft",
      "Satya_Nadella": "Microsoft"}
    ```
- Extra
  - A hash function may not provide a unique value for every key.
  - A value is accessed/retrieved by its key.

Trees

A nested structure, where each element/node has two parts.

- Parts
  - 1. the value of the node
  - 2. the branches - links to subsequent nodes
- Properties
  - 1. Every node has a fixed, maximum number of branches.
  - 2. The node that is not a branch, is the root node.
  - 3. Branches nodes are "children" of their parent node.
  - 4. Any search for any value starts at the root node.
- Example
  - In a binary tree, nodes can have a max of 2 branches.

Heaps

A Tree where the value of every parent node is the maximum or minimum of all of child nodes (and their children, etc.) connected to that parent.

- Examples
  - `Max value heap`: A parent node has value 255. Its child nodes must have values 255 or less. Parent holds the max value.
  - `Min value heap`: A parent node has value 255. Its child nodes must have values 255 or more. Parent holds the min value.
- The "greater/less than or equal to" rule, applies to every node, not just the parent.
    ```