General Assembly

INTRODUCTION TO DATA STRUCTURES

What Is a Data Structure?

A **data structure** is a container that stores data in a specific, organized way. It holds data points so that they can be accessed, used, and analyzed. There are five basic operations you can perform on a data structure:

1. **Access**: Retrieving a specific data point
2. **Insert**: Adding a data point
3. **Delete**: Removing a data point
4. **Search**: Determining whether a specific value is in the dataset
5. **Sort**: Ordering data points by one or more of their characteristics

Choosing a Data Structure

Before we can perform operations on a data structure, we have to choose the type of data structure we want to use. It's critical to structure your data well at the beginning of a project; no one data structure is equally efficient for all five operations, and some data structures don't support certain operations at all. Here are some common data structures:

- **Arrays**: Ordered collections of items added one by one
- **Linked Lists**: Lists that store data inside of nodes that point to each other
- **Stacks**: Lists of ordered values in which the first item in is the last item out
- **Queues**: Lists of ordered values in which the first item in is the first item out
- **Hash Tables**: List-like structures that use a hash function to generate keys for values
- **Sets**: Unordered collections of unique elements
- **Binary Trees**: Node-based structures that store values using left and right pointers
- **Tries**: Tree structures in which nodes can store lists of child nodes (almost always used for strings)
- **AVL Trees**: Binary trees that are balanced (i.e., the left and right branches of the tree are the same height)
- **Graphs**: Collections of nodes and edges in which nodes store data and edges represent connections between data

The Array

The **array**, an ordered collection of items, is the most commonly used data structure across all programming languages. Because arrays are so common, their specifications (the ways in which they are defined) are the same across most programming languages. Arrays have several key characteristics:

- They are **indexed**: each item in an array has an index associated with it, which corresponds to the item's position in the array.
- Items can be accessed individually via their index.
- In most programming languages, the index starts at 0 (so the index of the first item is 0, the index of the second item is 1, and so on)
- Their size is either **fixed** (Java, C) or **variable** (JavaScript, Ruby), depending on the programming language.
- They are typically restricted to one data type (e.g., integers, strings, Booleans).