Hash Maps

Learn to Code with Rust / Section Review

Hash Maps

- A **HashMap** is a collection type consisting of unordered key-value pairs.
- A key is a unique identifier for a value.
- The values can contain duplicates.
- A hash map establishes mappings/connections between two pieces of data.
- The length of a hashmap is a count of its key-value pairs.

Creating a HashMap

- The **std::collections** submodule contains the **HashMap** type.
- The **new** constructor function instantiates an empty **HashMap**.
- If the code does not add a key-value pair, we must provide a type declaration.
- The **HashMap** defines 2 generic types: **K** and **V**. The keys will be of type **K** and the values of type **V**.

HashMap Methods

- The **insert** method adds a new key-value pair. If the key exists, Rust will replace the existing value.
- The **remove** method removes a key-value pair using the key. It returns an **Option** where the **Some** variant will store the value.
- The **entry** method returns an **Entry** struct with an **or_insert** method that adds a key-value pair only if the key is not present.

Accessing a Value

- Use square brackets to access a value by a key.
- The program will panic if the key does not exist.
- The **get** method accepts a key and returns an **Option**.
  - The **Some** variant will store a reference to the value.
  - The **None** variant stores no associated data.