Hashmap

- **Ownership:** HashMaps take ownership of keys and values unless you use references
- **Keys:** Must implement `Eq` and `Hash` traits
- **Performance:** Average O(1) time complexity for operations

```rust
use std::collections::HashMap;

fn main() {
    // Create a new HashMap
    let mut scores = HashMap::new();

    // Insert key-value pairs
    scores.insert(String::from("Blue"), 10);
    scores.insert(String::from("Yellow"), 50);

    // Access values
    let team_name = String::from("Blue");
    let score = scores.get(&team_name); // Returns Option<&V>
    println!("Blue team score: {:?}", score);
}


```

HashMap Methods

	Method	Description
1	`insert(K, V)`	Insert key-value pair
2	`get(&K)`	Get value for key
3	`get_mut(&K)`	Get mutable reference to value
4	`remove(&K)`	Remove key-value pair
5	`contains_key(&K)`	Check if key exists
6	`len()`	Get number of elements
7	`is_empty()`	Check if empty
8	`keys()`	Get iterator over keys
9	`values()`	Get iterator over values
10	`entry(K)`	Get Entry API for complex updates