QUIZ

What is *separate chaining*?

- Converting keys to integers with the `.charCodeAt()`
- Using linked lists to store values in a hash map array in order to avoid collisions

👏 You got it!

- Chaining multiple method calls with the `.` notation to an initial method call
- Storing many different values in the same data structure like an array

Finish this `.hash()` method:

```
class HashMap {
  constructor(size = 0) {
  this.hashmap = new Array(size)
    .fill(null);
}

hash(key) {
  let hashCode = 0 ;

  for (let i = 0; i < key.length ; i++) {

    hashCode += hashCode + key.charCodeAt(i) ;
  }
    return hashCode % this.hashmap.length ;
  }
}
```

👏 You got it!

Fill in the code for the `HashMap` constructor method so we can use separate chaining:

```
class HashMap {
  constructor(size = 0) {
    this.hashmap = new Array(size)
      .fill(null)
      .map(() => new LinkedList());
  }
}
```

You got it!

Fill in the code to finish `.retrieve()`:

```
retrieve(key) {
  const arrayIndex = this.hash(key);
  let current = this.hashmap[arrayIndex].head ;

  while (current) {
    if ( current.data.key === key ) {
      return current.data.value ;
    }
    current = current.next ;
  }
  return null;
}
```

You got it!

Will running this code result in a collision between keys?

```
class HashMap {
  constructor(size = 0) {
    this.hashmap = new Array(size)
      .fill(null);
  }

    hash(key) {
    let hashCode = 0;
    for (let i = 0; i < key.length; i++) {
      hashCode += hashCode + key.charCodeAt(i);
    }
    return hashCode % this.hashmap.length;
  }

  assign(key, value) {
    const arrayIndex = this.hash(key);
    this.hashmap[arrayIndex] = value;
  }
}
```

Sometimes. If key-value pairs store the same value, there will be a collision between the keys.

Yes, this hash map implementation will result in a collision between keys. Any value stored at an index in the hash map array will be overwritten if another key is assigned to that index.

Correct! The `.assign()` method only overwrites values, it doesn't check that keys are matching and doesn't make use of separate chaining.

Fill in the code regarding the `current` variable in `.assign()`:

```
assign(key, value) {
  const arrayIndex = this.hash(key);
  const linkedList = this.hashmap[arrayIndex];

  if (linkedList.head === null) {
    linkedList.addToHead({ key, value });
    return ;
  }

  let current = linkedList.head ;

  while (current) {
    if (current.data.key === key) {
      current.data = { key, value } ;
    }

    if (!current.getNextNode()) {
      let tail = new Node({key, value});
      current.setNextNode(tail) ;
      break;
    }
      current = current.getNextNode() ;
    }
  }
}
```

You got it!