Collisions

1 min

We have a hash map implementation, but what happens when two different keys generate the same index? Run the code in **collision.js** to see a collision in action.

Instead of returning 'marsh plant' and 'forest animal' we retrieve 'forest animal' twice. This is because both key-value pairs are assigned to the same index 0 and the first value, 'marsh plants' was overwritten.

When two different keys resolve to the same array index this is called a collision. In our current implementation, all keys that resolve to the same index are treated as if they are the same key. This is a problem because they will overwrite one another's values.

Instructions

1. Checkpoint 1 Passed

1.

Run the code in the text editor to see the result of a collision between two keys.

collision.js

```
const LinkedList = require('./LinkedList');

const Node = require('./Node');

class HashMap {

  constructor(size = 0) {

    this.hashmap = new Array(size);

  }


  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;

  }
  retrieve(key) {
    const arrayIndex = this.hash(key);
    return this.hashmap[arrayIndex];
  }
}

module.exports = HashMap;

const parkInventory = new HashMap(2);
parkInventory.assign('reed', 'marsh plant');
parkInventory.assign('deer', 'forest animal');

console.log(parkInventory.retrieve('reed'));
console.log(parkInventory.retrieve('deer'));
```