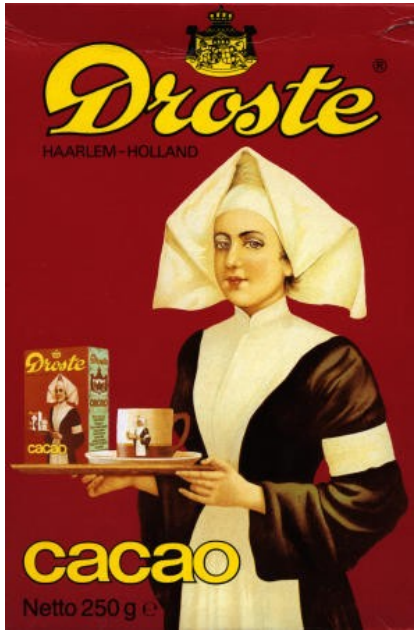
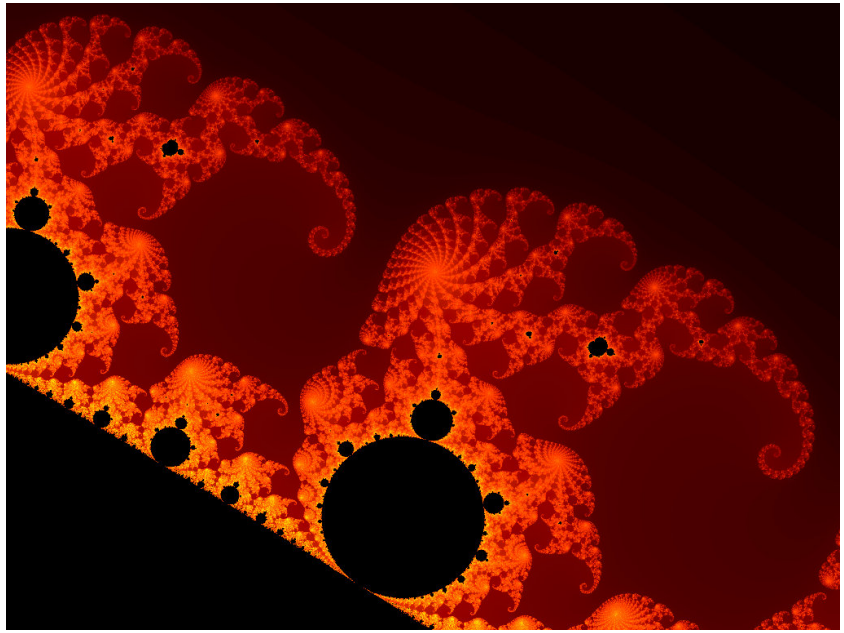
Hash Maps

Daniel Archambault

Previously in CS-115





Multirecursion twice the fun, right? :(

Previously in CS-115

- What is multirecursion?

Previously in CS-115

- What is multirecursion?
- What are the two main parts of any recursive algorithm?

Previously in CS-115

- What is multirecursion?
- What are the two main parts of any recursive algorithm?
- What ADT is useful for tracing multirecursive algorithms?

Previously in CS-115

- Now it's time for the most giggle inducing lecture of CS-115!

Hash Maps

Philosophical Pause

Based on CC Flicker Image: *Colton Witt: As I Look, I wonder*

Philosophical Pause

Based on CC Flicker Image: *Colton Witt: As I Look, I wonder*

- Is there a way to use non-integer keys can be used as array indexes?

A Dreamland!

- A data structure with fast look-ups, like an array.
- Fast inserts into a specific location, like an array.
- But, does not take integer keys as indexes
  - Oh, no, it can take weird non-integer things like strings
  - And maybe, it can have a facility to write a method so it can take custom classes as a key!
- But, maybe we are living in a dreamworld?

A Dreamland!

- A data structure with fast look-ups, like an array.
- Fast inserts into a specific location, like an array.
- But, does not take integer keys as indexes
  - Oh, no, it can take weird non-integer things like strings
  - And maybe, it can have a facility to write a method so it can take custom classes as a key!
- But, maybe we are living in a dreamworld?
  - Yes, Virginia, there is such a data structure! :)

Hash Map ADT

- Is usually expressed as a big array.
- Should be able to take non-integer keys (String, Double, Date, maybe even ClosedShape...)
- Should have fast access and overwrite of elements like an array.
- Not built for iteration like a list, but it would be nice.
- `https://docs.oracle.com/javase/8/docs/api/java/util/HashMap.html`

Hash Map ADT

- Is usually expressed as a big array.
- Should be able to take non-integer keys (String, Double, Date, maybe even ClosedShape...)
- Should have fast access and overwrite of elements like an array.
- Not built for iteration like a list, but it would be nice.
- `https://docs.oracle.com/javase/8/docs/api/java/util/HashMap.html`
- But, what's really awesome is that no recursion is involved.

Motivation

- We want to do crazy things like this

```
myHashMap["Dan!"] = 27;
myHashMap["Swansea!"] = 33;
```

- This might be cool too...

```
myHashMap[3.14] = "Circle time!";
myHashMap[2.71] = "Natural Log";
```

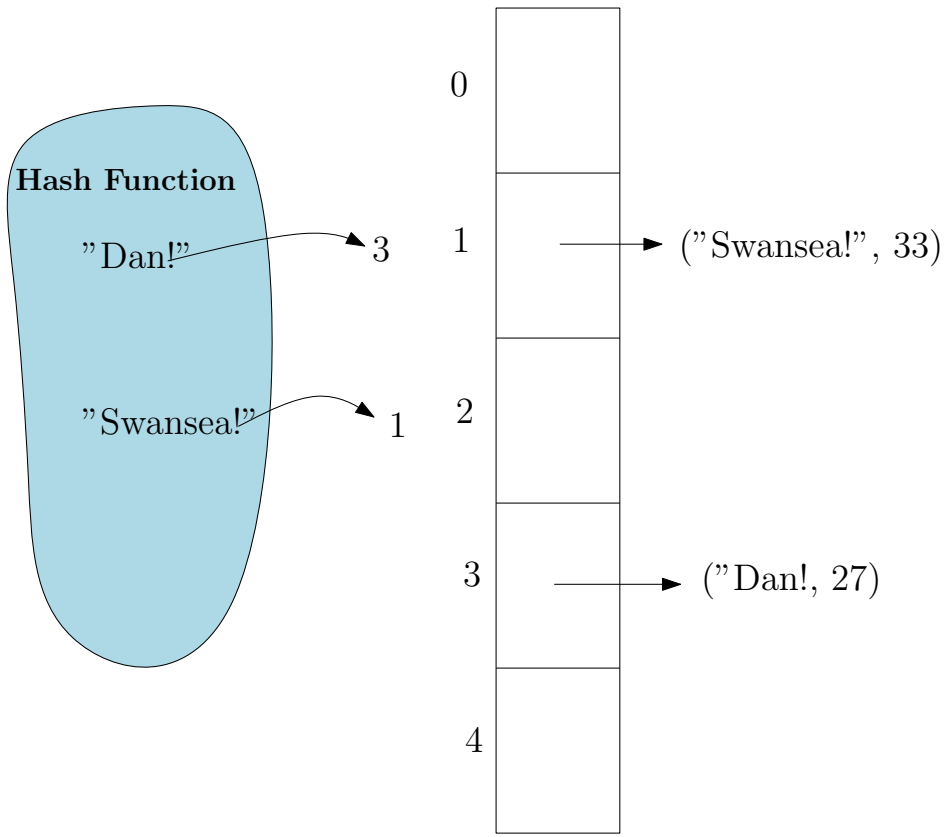
- But how can we do such things? Syntax is a bit different.

Approach

- Create a big array. Elements are called buckets.
- Write a hashing function for the non-integer keys
  - First example, write one that converts strings to ints
  - In the second example, write one that converts doubles to ints
- Whenever presented with a key, use function to look up
  - Use hashing function to convert it into an integer
  - The modulo operator (“%”) tells which bucket to look in

What a Hash Map Looks Like

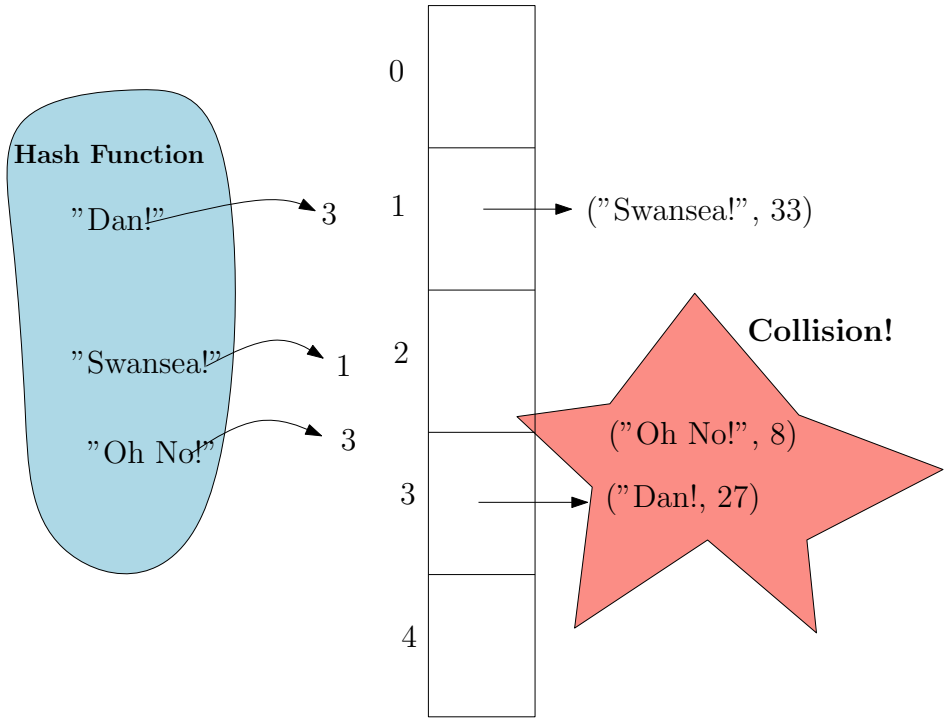
```
myHashMap.put ("Dan!", 27);
myHashMap.put ("Swansea!", 33);
```



Collisions Can Occur

- There are no guarantees that two different keys won't has to the same index
  - This is what is known as a collision

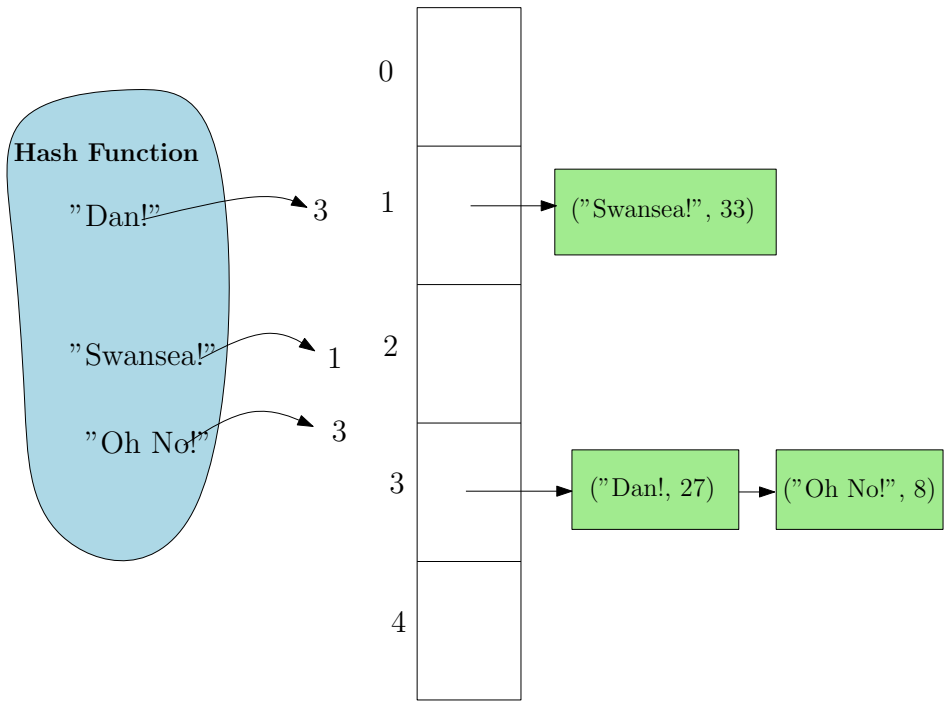
```
myHashMap.put("Dan!", 27);
myHashMap.put("Swansea!", 33);
myHashMap.put("Oh No!", 8);
```



Chaining Solution

- To handle collisions, Java API does chaining for you
  - It constructs a linked list for buckets
  - Assuming number of collisions is small, it won't be too long

```
myHashMap.put("Dan!", 27);
myHashMap.put("Swansea!", 33);
myHashMap.put("Oh No!", 8);
```



How do I get my data out?

- You can use the `get` method

```
myHashMap.put("Dan!", 27);
myHashMap.put("Swansea!", 33);
myHashMap.put("Oh No!", 8);
myHashMap.get("Dan!"); //returns 27.
```

What happens if I hash stuff to the same key?

- Just like an array, the value gets overwritten.

```
myHashMap.put("Dan!", 27);
myHashMap.put("Dan!", 33); //Overwrite with 33
myHashMap.put("Dan!", 8);  //Overwrite with 8
myHashMap.get("Dan!"); //returns 8.
```

Declaring own Hash Map

- HashMap is a class in the Java API
  - `https://docs.oracle.com/javase/8/docs/api/java/util/HashMap.html`

```
HashMap<String, Integer> myHashMap =
    new HashMap<String, Integer> ();
```

- HashMap is a generic
  - Both the key and the value must be class types
  - This means you need to use Integer for ints...

Do I have to write a hash function?

- Good news, many hash functions become pre-implemented with the API
  - `String`, `Double`, etc.
- Okay great, I'll just go ahead and do this!

```
HashMap<MyClass, String> myMap = ...
```

- Believe it or not this will work (but the hashing will be sub-optimal)
  - Consider overriding `hashCode` for a better spread of keys
  - `https://docs.oracle.com/javase/8/docs/api/java/util/AbstractMap.html#hashCode--`
- Often, consider adding the hash codes of non-static attributes
- Think about collisions... how do we avoid them occurring?

Summary and Review

- Introduction to Hash Maps and how to use them
- How does a Hash Map work?
- Collisions, what are they again?
- How are collisions resolved?
- What is a hashing function?