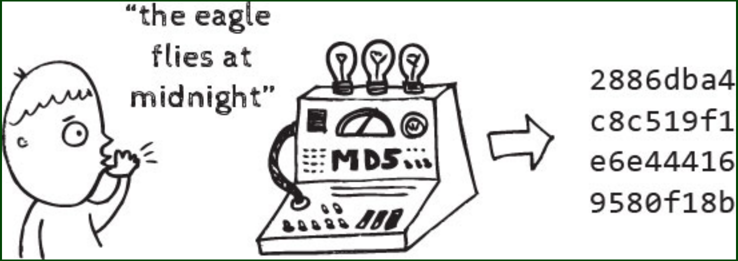
CSE 143

Computer Programming II

Hashing


"the eagle flies at midnight"
MD5
2886dba4
c8c519f1
e6e44416
9580f18b

IntSet

Today, we will consider multiple new implementations of `IntSet`:

```java
public interface IntSet {
   public void add(int value);
   public void remove(int value);
   public boolean contains(int value);
}
```

A New Data Structure

Design a class `RangeSet` that represents a set which only allows numbers inside a **fixed range**.

You should have a constructor:

`RangeSet(`**`max`**`)`	This constructor initializes a new `RangeSet` which only allows elements between 0 (inclusive) and **max** (exclusive).

And the following **public** methods:

`add(`**val**`)`	Adds **val** to the `RangeSet` if it is a valid value and throws an `IllegalArgumentException` otherwise.
`remove(`**val**`)`	Removes **val** to the `RangeSet` if it is a valid value in the set and does nothing otherwise.
`contains(`**val**`)`	Returns true if **val** is in the `RangeSet` and false otherwise.

`add`, `remove`, and `contains` must all be $\mathcal{O}(1)$

RangeSet

```
public class RangeSet implements IntSet {
   private boolean[] data;

   public RangeSet(int max) { this.data = new boolean[max]; }

   public void add(int value) {
      if (value >= this.data.length || value < 0) {
         throw new IllegalArgumentException();
      }
      this.data[value] = true;
   }

   public boolean contains(int value) {
      if (value >= this.data.length || value < 0) {
         return false;
      }
      return this.data[value];
   }

   public void remove(int value) {
      if (value < this.data.length && value >= 0) {
         this.data[value] = false;
      }
   }
}
```

Generalizing to an Arbitrary Range

In `RangeSet`, when we got the number n, we mapped it to the index n. What if we had a function that took an input and mapped it to an index?

Definition (HashCode)

A **hash code** is a function that takes in a piece of data and maps it to an array index.

If we have an array of size 8, consider the following hashcode:

```java
public int hashCode(int value) {
   return value % 8;
}
```

Now, let's insert the following data: 1, 4, 13

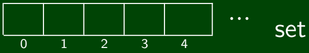
set[0]	set[1]	set[2]	set[3]	set[4]	set[5]	set[6]	set[7]

	1			4	13		
set[0]	set[1]	set[2]	set[3]	set[4]	set[5]	set[6]	set[7]

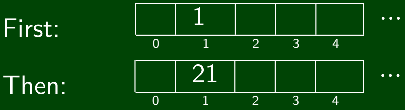
HashSet Attempt #1

```
public class IntHashSet implements IntSet {
   public final int DEFAULT_SIZE = 20;
   public Integer[] data;

   public IntHashSet() {
      this.data = new Integer[DEFAULT_SIZE];
   }

   private int hashCode(int value) {
      return value % data.length;

   public void add(int value) {
      this.data[hashCode(value)] = value;
   }

   public boolean contains(int value) {
      return this.data[hashCode(value)] == value;
   }

   public void remove(int value) {
      this.data[hashCode(value)] = null;
   }
```

HashSet Attempt #1 Problem!



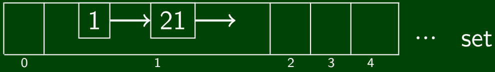
Consider the following insertions: 1, 21



Uh oh! We've overwritten the one!

How can we fix this?

Instead of storing an integer, let's store a list of integers



String HashCode!

```
public int hashCode() {
   int h = hash;
   if (h == 0 && value.length > 0) {
      char val[] = value;

      for (int i = 0; i < value.length; i++) {
         h = 31 * h + val[i];
      }
      hash = h;
   }
   return h;
}
```