Before class

• Play around with the hashCode method in the Double class

    Double d = new Double(.00000003);
    System.out.println(d.hashCode());

• See if you can find the largest hash code for a number between 0-1

• Submit your answer here:

    http://goo.gl/acA46
How I did it

Double d = new Double(0.0);
int max = d.hashCode();
Double maxD = new Double(d.doubleValue());
for(int i=0; i < 1000000000; i++)
{
    d = d + 0.000000001;
    int temp = d.hashCode();
    if(temp > max)
    {
       max = temp;
       maxD = d.doubleValue();
    }
}
System.out.println(maxD + " has the hash value " + max);
public int compareTo(ThreeInts other)
{
    int mySum = myOne + myTwo + myThree;
    int otherSum = other.myOne + other.myTwo + other.myThree;
    return mySum - otherSum;
}
Create a class ComplexNumber

- use ThreeInts as your guide

ComplexNumber objects should have only two instance variables, myR and myI

Write a compareTo method.

- complex numbers should be compared using magnitudes

\[ \sqrt{r^2 + i^2} \]
ComplexNumber a = new ComplexNumber(1,7);
ComplexNumber b = new ComplexNumber(1,7);

if(a.equals(b))
{
    System.out.println("The complex numbers are equal");
}
else
{
    System.out.println("The complex numbers are not equal");
}

The complex numbers are not equal
Hash Codes

- “cat” hashes to 98262
- “bat” hashes to 97301
- “act” hashes to 96402
- [4] hashes to 35
- [4,6] hashes to 1091
When you create a class

• every object should have a hash code

• the hash code should not change unless an instance variable changes value

• two objects are equals() if they have the same hash code

• two objects are !equals() if they have different hash codes
Hints for making hash codes

• Don’t write your hash code method from scratch
  • Use existing Java hashCode methods in creative ways

• Computing hashCodes is SLOW
  • save your hash code as in instance variable
  • only recompute your hash code if you need to
Code time

• Add a \texttt{hashCode()} and \texttt{equals()} to \texttt{ThreeInts}
Your turn

• Add equals() and hashCode() to ComplexNumbers
  
  • two complex numbers are equal if their real and imaginary parts are equal
  
  • the hash code MUST be equal if the real and imaginary parts are equal
  
  • Order matters: 5 + 3i != 3 + 5i
  
  • test your code!
Questions?

- `compareTo`: You will need to write one for the next homework assignment.

- `hash codes`