Structured Data in Java
An Introduction

Sample problem

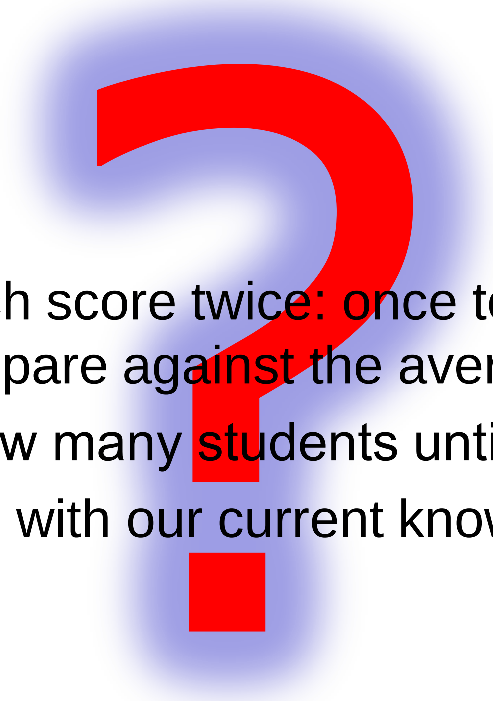
- Given a set of test scores for a class of students, compute the average
  - Number of students may vary from class to class

- Solution:
  - Read in number of students in the class
  - Write a cumulative sum loop, for the number of students, that reads a score and adds it to the total of all scores
  - Divide the total by the number of students

Sample problem

- Given a set of test scores for a class of students, compute the average ***and report which scores were below the average***

- Solution:
  - We now need each score twice: once to compute the average and once more to compare against the average
  - We don’t know how many students until the program is running
  - We cannot do this with our current knowledge of Java



Solution Requirements

1. Need to store a collection of data, rather than just one data item

2. Need to be able to treat the collection as a single entity while still allowing access to any one member

3. Need an easy way to process each and every member of the collection

4. Need to access members of the collection multiple times, sometimes in an unknown order

Java’s Solution

- Arrays

- Collections Framework
  - Lists
    - ArrayList
  - Maps
    - HashMap
  - Sets
  - Queues
  - Etc.