Data and Information

Google Announces Plan To Destroy All Information It Can’t Index

How and why do we organize data? Differences between data and information? What about knowledge?

Daily Plan: What will we do today?

- Pithy aphorisms to remember as we think and code
  - Sometimes from “famous people”

- Discussion of Java arrays/ArrayLists, code examples
  - Indexable, why are there two types? Tradeoffs
  - Reference to current assignment as well as other code

- Demo of APT, plan, execute, attack, dance
  - When is enough planning too much? Coding at 2:00 am?

- Final pithy aphorism

Organizing Data: ideas and issues

- Often there is a time/space tradeoff
  - If we use more space (memory) we can solve a data/information problem in less time: *time efficient*
  - If we use more more time, we can solve a data/information problem with less space: *space efficient*

- Search v Store: repeating the same thing again ...
  - We’re not “smart” enough to avoid the repetition
    - Learn new data structures or algorithms!
  - The problem is small enough or done infrequently enough that being efficient doesn’t matter
  - Markov illustrates this (two assignments from now)

Arrays: random access/one type

- Data stored in memory, one object per "slot"
  - Contiguous in memory, why?
  - Addressable by index
  - Zero is first index

- In Java: homogeneous
  - All int, or String or ...
  - Python lists are different

- Array v. ArrayList
  - primitive v object
  - int v Integer
John von Neumann

“Anyone who attempts to generate random numbers by deterministic means is, of course, living in a state of sin.”

“There’s no sense in being precise when you don’t even know what you’re talking about.”

“There are two kinds of people in the world: Johnny von Neumann and the rest of us.”

Eugene Wigner, Noble Physicist

Objects and values

- **Primitive variables are boxes**
  - think memory location with value
- **Object variables are labels that are put on boxes**

```java
String s = new String("genome");
String t = new String("genome");
if (s == t) {they label the same box}
if (s.equals(t)) {contents of boxes the same}
```

What’s in the boxes? “genome” is in the boxes

Objects, values, classes

- **For primitive types: int, char, double, boolean**
  - Variables have names and are themselves boxes (metaphorically)
  - Two int variables assigned 17 are equal with ==

- **For object types: String, ArrayList, others**
  - Variables have names and are labels for boxes
  - If no box assigned, created, then label applied to null
  - Can assign label to existing box (via another label)
  - Can create new box using built-in new

- **Object types are references/pointers/labels to storage**

Anatomy of a class

```java
public class Foo {
    private int mySize;
    private String myName;
    public Foo(){  
        // what's needed?
    }
    public int getSize(){
        return mySize;
    }
    public double getArea(){
        double x;
        x = Math.sqrt(mySize);
        return x;
    }
}
```

- **What values for vars (variables) and ivars (instance variables)?**
David Parnas

"For much of my life, I have been a software voyeur, peeking furtively at other people's dirty code. Occasionally, I find a real jewel, a well-structured program written in a consistent style, free of kludges, developed so that each component is simple and organized, and designed so that the product is easy to change."

Parnas on re-invention

"We must not forget that the wheel is reinvented so often because it is a very good idea; I've learned to worry more about the soundness of ideas that were invented only once."

David Parnas (entry in Wikipedia)

- **Module Design:** Parnas wrote about the criteria for designing modules, in other words, the criteria for grouping functions together. This was a key predecessor to designing objects, and today's object-oriented design.
- **Social Responsibility:** Parnas also took a key stand against the Strategic Defense Initiative (SDI) in the mid 1980s, arguing that it would be impossible to write an application that was free enough from errors to be safely deployed.
- **Professionalism:** He believes that software engineering is a branch of traditional engineering.

Preview of APTs and Assignments

- **Rules in writing code to solve APTs**
  - Have a plan for how to solve the problem
  - Demonstrate plan by using algorithm on one example
  - Think about how to implement plan
    - Instance variables? Helper methods?
    - Write and rewrite. Don't worry about lengthy methods.
    - Do worry about length methods, rewrite!
  
- **From Hangman to APT to Jotto**
  - Arrays, loops, Sets, reasoning
  - Three braids of grading: engineering, algorithmic, analysis
  - There is no try? Yoda is wrong