Tools: Solve Computational Problems

- **Algorithmic techniques**
  - Brute-force/exhaustive, greedy algorithms, dynamic programming, divide-and-conquer, ...

- **Programming techniques**
  - Recursion, memo-izing, compute-once/lookup, tables, ...

- **Java techniques**
  - java.util.*, Comparator, Priority Queue, Map, Set, ...

- **Analysis of algorithms and code**
  - Mathematical analysis, empirical analysis

---

Quota Exceeded

- You’re running out of disk space
  - Buy more
  - Compress files
  - Delete files

- How do you find your “big” files?
  - What’s big?
  - How do you do this?

---

Recursive structure matches code

```java
public static long THRESHOLD = 1000000L; // one mil

public static void findBig(File dir, String tab)
{ File[] dirContents = dir.listFiles();
  System.out.println(tab + dir.getPath());
  for (File f : dirContents)
    { if (f.isDirectory())
     { findBig(f, tab + “\t”); } 
    else {
      if (f.length() > THRESHOLD)
     { System.out.printf(“%s%n”, f.getName(), f.length()); } 
    }
  }
}
```

- Does findBig call itself?

---

Solving Problems Recursively

- Recursion is an indispensable tool in a programmer’s toolkit
  - Allows many complex problems to be solved simply
  - Elegance and understanding in code often leads to better programs: easier to modify, extend, verify (and sometimes more efficient!!)
  - Sometimes recursion isn’t appropriate, when it’s bad it can be very bad—every tool requires knowledge and experience in how to use it

- The basic idea is to get help solving a problem from coworkers (clones) who work and act like you do
  - Ask clone to solve a simpler but similar problem
  - Use clone’s result to put together your answer

- Need both concepts: call on the clone and use the result
Print words read, but print backwards
- Can use an ArrayList, store all the words and print in reverse order
  - Probably the best approach, recursion works too
    public void printReversed(Scanner s)
    if (s.hasNext())
      String word = s.next(); // store word
      System.out.println(word); // print the word
    }
  - The function printReversed reads a word, prints the word only after the clones finish printing in reverse order
    - Each clone has own version of the code, own word variable
    - Who keeps track of the clones?
    - How many words are created when reading N words?
      - Can we do better?

Exponentiation
- Computing \(x^n\) means multiplying n numbers (or does it?)
  - What's the easiest value of n to compute \(x^n\)?
  - If you want to multiply only once, what can you ask a clone?
    public static double power(double x, int n)
    if (n == 0)
      return 1.0;
    else
      return x * power(x, n-1);
  - Number of multiplications?

Faster exponentiation
- How many recursive calls are made to compute \(2^{1024}\)?
  - How many multiplies on each call? Is this better?
    public static double power(double x, int n)
    if (n == 0)
      return 1.0;
    double semi = power(x, n/2);
    if (n % 2 == 0)
      return semi*semi;
    else
      return x * semi * semi;
  - What about an iterative version of this function?

Back to Recursion
- Recursive functions have two key attributes
  - There is a base case, sometimes called the exit case, which does not make a recursive call
    - See print reversed, exponentiation
  - All other cases make a recursive call, with some parameter or other measure that decreases or moves towards the base case
    - Ensure that sequence of calls eventually reaches the base case
    - “Measure” can be tricky, but usually it’s straightforward
  - Example: finding large files in a directory (on a hard disk)
    - Why is this inherently recursive?
    - How is this different from exponentiation?
Thinking recursively

- Problem: find the largest element in an array
  - Iteratively: loop, remember largest seen so far
  - Recursive: find largest in [1..n), then compare to 0th element

```java
public static double max(double[] a) {
    double maxSoFar = a[0];
    for (int k = 1; k < a.length; k++) {
        maxSoFar = Math.max(maxSoFar, a[k]);
    }
    return maxSoFar;
}
```

- In a recursive version what is base case, what is measure of problem size that decreases (towards base case)?

Recursive Max

```java
public static double recMax(double[] a, int index) {
    if (index == a.length-1) { // last element, done
        return a[index];
    }
    double maxAfter = recMax(a, index+1);
    return Math.max(a[index], maxAfter);
}
```

- What is base case (conceptually)?
- Do we need variable maxAfter?
- We can use recMax to implement arrayMax as follows
  ```java
  return recMax(a, 0);
  ```

Recognizing recursion:

```java
public static void change(String[] a, int first, int last) {
    if (first < last) {
        String temp = a[first]; // swap a[first], a[last]
        a[first] = a[last];
        a[last] = temp;
        change(a, first+1, last-1);
    }
}
```

- What is base case? (no recursive calls)
- What happens before recursive call made?
- How is recursive call closer to the base case?

More recursion recognition

```java
public static int value(int[] a, int index) {
    if (index < a.length) {
        return a[index] + value(a, index+1);
    }
    return 0;
}
```

- What is base case, what value is returned?
- How is progress towards base case realized?
- How is recursive value used to return a value?
- What if a is array of doubles, does anything change?
**Recursive example: BlobCount**

- How do we find images? Components? Paths?
  - Create information from data

**From programming techniques to Java**

- Is recursion a language independent concept?
  - Do all languages support recursion?
  - What are the issues in terms of computer/compiler/runtime support?

- We use a language and its libraries, do we study them?
  - Should we know how to implement ArrayList
    - What are the building blocks, what are our tools
  - Should we know how to implement different sorts
    - Should we know how to call existing sorts

**java.util.Comparator**

- How does sorting work in general and in Java?
  - Characteristics of Java library sort methods
  - What can be sorted?
  - How do you change how sorting works?

- Use APT ClientsList as an example to explore Comparator
  - Creating new Comparator: nested class, anonymous class
  - Access issues: final or “persistent” data (why?)

- What does it mean to implement Comparable?
  - Other Java interfaces: cloneable, serializable, ...

**Eugene (Gene) Myers**

- Lead computer scientist/software engineer at Celera Genomics (now at Berkeley, now at …?)

- “What really astounds me is the architecture of life. The system is extremely complex. It’s like it was designed.” … “There’s a huge intelligence there.”