Admin

- Assign 2 due today, Assign 3 out
  - Joy poll
- Today's topics
  - Procedural recursion
- Reading
  - Reader ch. 5-6 (today, next)

Thinking recursively

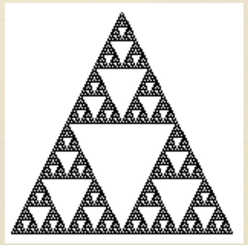
- Recursive decomposition is the hard part
  - Find recursive sub-structure
    - Solve problem using result from smaller subproblem(s)
  - Identify base case
    - Simplest possible case, directly solvable, recursion advances to it
- Common patterns
  - Handle first and/or last, recur on remaining
  - Divide in half, recur on one/both halves
  - Make a choice among options, recur on updated state
- Placement of recursive call(s)
  - Recur-then-process versus process-then-recur

Procedural vs functional

- Functional recursion
  - Function returns result
  - Computers using result from recursive call(s)
- Procedural recursion
  - No return value (function returns void)
  - Task accomplished during recursive calls
- Example: drawing fractal
  - Self-similar structure
  - Repeats itself within
  - Outer fractal makes recursive call to draw inner fractal(s)

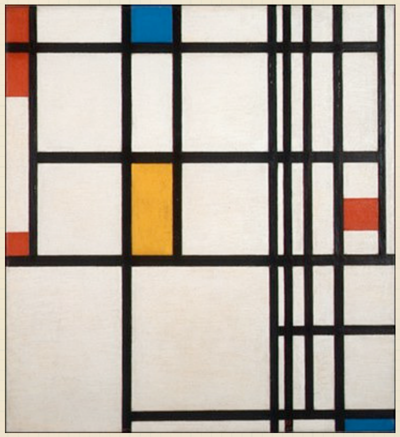
A familiar fractal

```
void DrawFractal (double x, double y, double w, double h)
{
   DrawTriangle(x, y, w, h);
   if (w < .2 || h < .2) return;
   double halfH = h/2;
   double halfW = w/2;
   DrawFractal(x, y, halfW, halfH);                    // left
   DrawFractal(x + halfW/2, y + halfH, halfW, halfH); // top
   DrawFractal(x + halfW, y, halfW, halfH);           // right
}
```



Recursive art

- Piet Mondrian
  - Dutch painter, 1872-1944
  - cubism, neoplasticism



I believe it is possible that, through horizontal and vertical lines constructed with awareness, but not with calculation, led by high intuition, and brought to harmony and rhythm, these basic forms of beauty, supplemented if necessary by other direct lines or curves, can become a work of art, as strong as it is true."

Random pseudo-Mondrian

- Choose one of three options
  - Divide canvas horizontally
  - Divide canvas vertically
  - Do nothing
- Dividing produces two smaller canvases
  - That can also be recursively painted in Mondrian style
- Base case stops at too-small canvas

Mondrian code

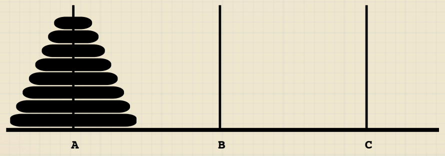
```
void DrawMondrian(double x, double y, double w, double h)
{
   if (w < 1 || h < 1 ) return;   // base case

   FillRectangle(x, y, w, h, RandomColor()); // fill background

   switch (RandomInteger(0, 2)) {
      case 0:     // do nothing
         break;
      case 1:     // bisect vertically
         double midX = RandomReal(0,w);
         DrawBlackLine(x+midX, y, h);
         DrawMondrian(x, y, midX, h);
         DrawMondrian(x+midX, y, w-midX, h);
         break;
      case 2:        // bisect horizontally
         double midY = RandomReal(0, h);
         DrawBlackLine(x, y+midY, w);
         DrawMondrian(x, y, w, midY);
         DrawMondrian(x, y + midY, w, h-midY);
         break;
   }
```

Towers

- Set of graduated disks stacked on a spindle
- Goal is move tower from source to destination



- Rules
  - All disks on a spindle (when not actively being moved)
  - Have one spare spindle
  - Can move only one disk at a time
  - Can only place disk on top of larger disk

Tower recursion

- Move tower of height N from A to B, using C
  - Starting thought: divide the tower
    - What is smaller instance of similar problem that helps?
    - Divide N height tower into one disk and tower of height n-1?
      - Which one to separate? Top or bottom disk?
      - What do you do with other tower?

Tower code

```
void MoveTower(int n, char src, char dst, char tmp)
{
   if (n > 0) {
      MoveTower(n-1, src, tmp, dst);
      MoveSingleDisk(src, dst);
      MoveTower(n-1, tmp, dst, src);
   }
}
```

Permutations

- Want to enumerate all rearrangements:
  - ABCD permutes to DCBA, CABD, etc.
- Solving recursively
  - Choose a letter from input to append to output
  - Recursively permute remaining letters onto output
  - What other options do you need to explore?
  - How to ensure each letter is used exactly once?
  - What is the base case?

Permute strategy

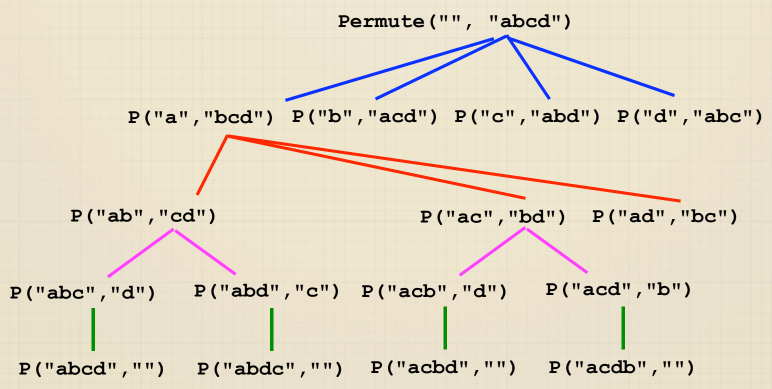
- Result is empty, starting input is "abcd"
- Choose a letter to be first, say "a"
- Result so far is "a", remaining input is "bcd"
- Recursively permute to get all "bcd" combos
- After finishing permutations with "a" in front, need to go again with "b" in front and then "c" and so on

Permute code

```
void RecPermute(string soFar, string rest)
{
    if (rest == "") {
        cout << soFar << endl;
    } else {
        for (int i = 0; i < rest.length(); i++) {
            string next = soFar + rest[i];
            string remaining = rest.substr(0, i)
                             + rest.substr(i+1);
            RecPermute(next, remaining);
        }
    }
}


 // "wrapper" function
void ListPermutations(string s)
{
    RecPermute("", s);
}
```

Tree of recursive calls



Permute("", "abcd")
P("a","bcd")
P("b","acd")
P("c","abd")
P("d","abc")
P("ab","cd")
P("ac","bd")
P("ad","bc")
P("abc","d")
P("abd","c")
P("acb","d")
P("acd","b")
P("abcd","")
P("abdc","")
P("acbd","")
P("acdb","")