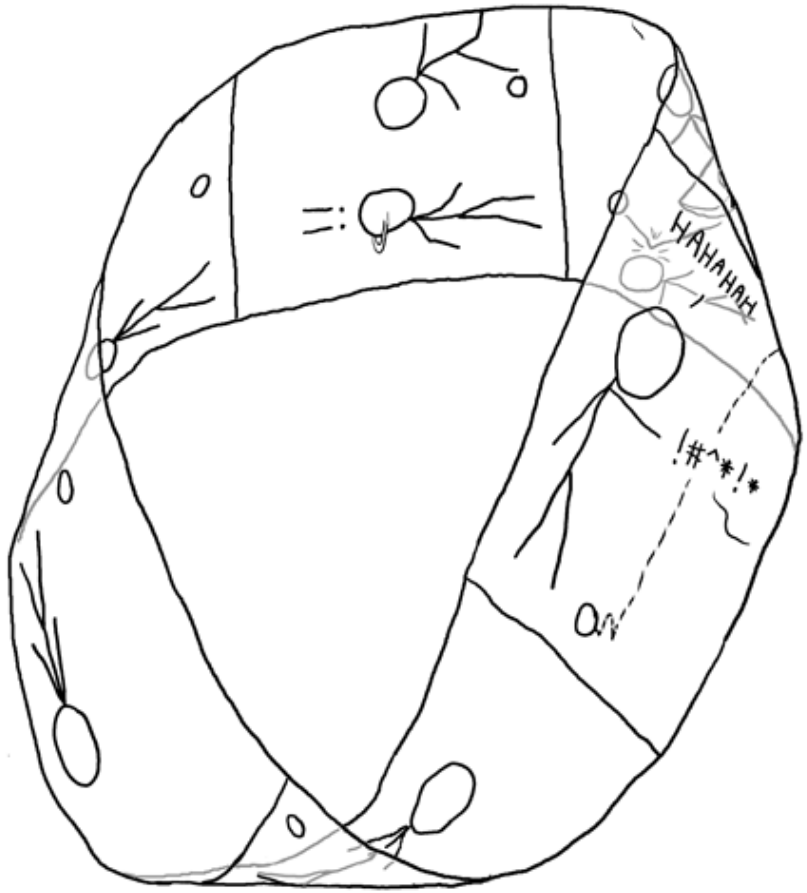
Recursion

Ch9 (functions): p320-327

CMPT 130

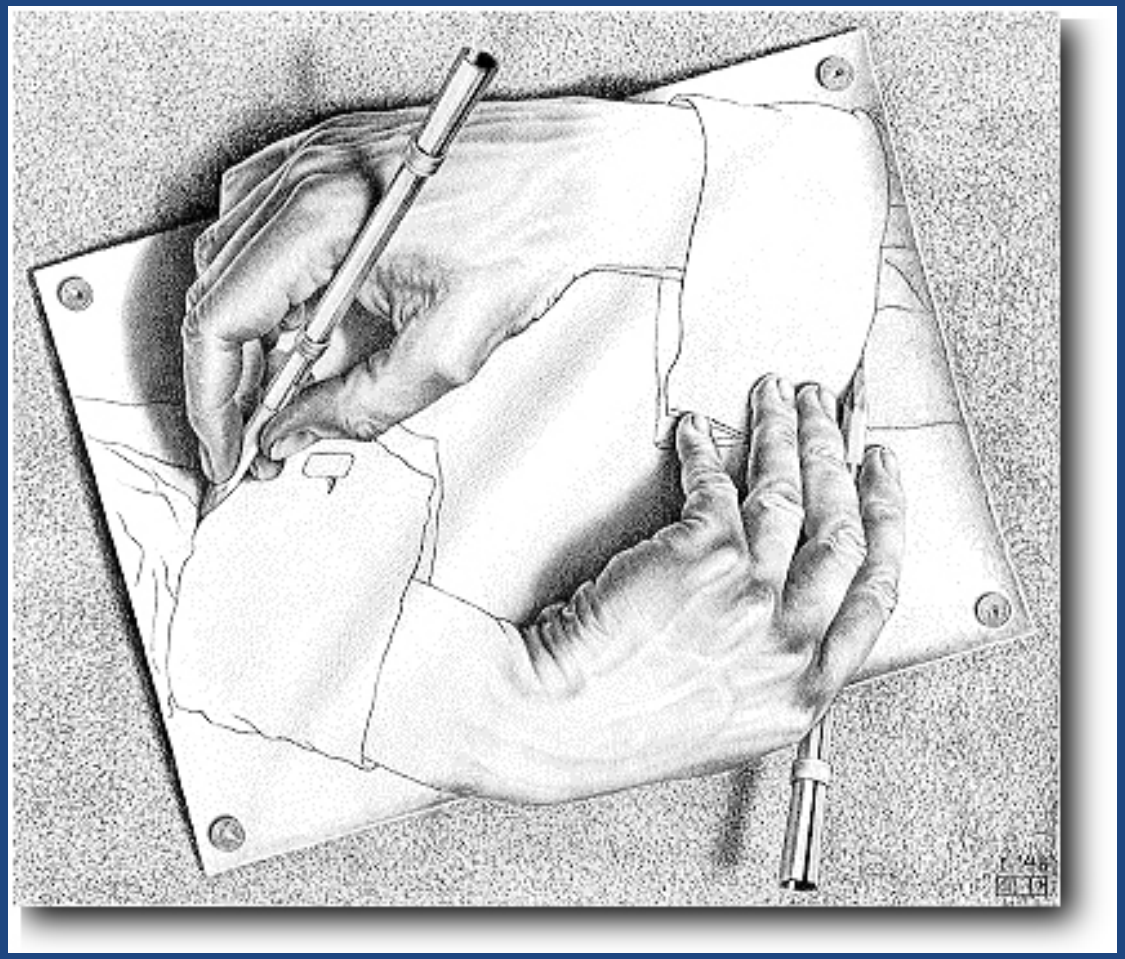


Moebius Battle
www.xkcd.com

Topics

Recursive:

1) Thinking
2) Programming
3) Problems & Design



Recursive Thinking

Recursion Jokes

- Recursion is when..
  - For more information than shown here,..
- GNU (makers of the GCC compiler):
  - “GNU” = GNU is Not Unix
- “Joke”:
  - Knock Knock.
  - Who's there?
  - Knock.
  - Knock Who?
  - Knock Knock.

Example: Factorial

- Math is full of recursion.
- n Factorial (n!):
  - 1! = 1
  - n! = n * (n-1) * (n-2) * ... * 2 * 1
- Example:
  - 3! = 3 * 2 * 1 = 6
  - 5! = 5 * 4 * 3 * 2 * 1 = 120
  - 20! = ... = 2,432,902,008,176,640,000
- n! Recursive definition:
  - Base case:..
  - Recursive Definition:..

n!

```
int factorial (int n) {
    // Base case:
    if (n == 1) {
        return 1;
    }

    // Recursive step:
    return n * factorial(n-1);
}
```

Non recursive step in the algorithm

Call the function on a smaller but..

factorial(n=1)
return 1

factorial(n=2)
return 2 * factorial(1)

factorial(n=3)
return 3 * factorial(2)

```
main {
  factorial(3);
}
```

Recursive Programming

Sum Numbers 1 To n

- Recursive definitions can often be..

```
// Sum the values from 1 through n:
int sum (int n) {
    // Base case:
    if (n == 1) {
        return 1;
    }

    // Recursive step:
    return n + sum(n-1);
}
```

sum(n=1)
return 1

sum(n=2)
return 2 * sum(1)

sum(n=3)
return 3 + sum(2)

```
main {
  sum(3);
}
```

Recursion vs Iteration

- All recursive problems can be solved by iteration.
- Why use recursion?
  - Recursion often more elegant.
  - Recursion can be faster (some cases)
    - Ex: With trees recursion may be much faster.
  - Recursion can be inefficient (extra function calls).
- If performance **really** matters, write it both ways and time it.
  http://www.ahmadsoft.org/articles/recursion/index.html

Stack Overflow

- https://stackoverflow.com/
- What is a stack overflow?
  - Every recursive call is a separate function call
    - And requires its own stack frame
  - Stack memory is finite
    - As is any other memory
  - Repeated recursive calls may exhaust the stack
- Some algorithms are very unlikely to result in stack overflow
  - Recursive binary search – OK
  - Recursive linear search – not so good

Practice

- Write recursive function for the following:
  - Binary Search.
    - What's the base case?
    - What's the recursive case?
  - Fibonacci Number Sequence:
    Sequence: 0, 1, 1, 2, 3, 5, 8, 13, 21, ...
    - Sequence definition:
      fib(0) = 0
      fib(1) = 1
      fib(n) = fib(n-2) + fib(n-1)

Review

- What do the following functions do?

```
bool guess1(int n)
{
    if (n == 0) {
        return true;
    }
    return !guess1(n-1);
}
```

Test Output:

guess1(0) = true
guess1(1) = false
guess1(2) = true
guess1(3) = false
guess1(4) = true
guess1(5) = false

```
int guess2(int data[], int size)
{
    if (size == 1) {
        return data[0];
    }
    return data[size - 1]
        + guess2(data, size -1);
}
```

Test Output:

guess2((int[]){1, 2, 3}, 3) = 6
guess2((int[]){10, 5, 30, 100, 0, 1}, 6) = 146

Summary

- Recursion is a powerful way of thinking about problems.
- Recursive methods call themselves:
  - Base case can be solved trivially.
  - Recursive case reduces the problem, then calls itself.
- Recursive Examples:
  - n!
  - Fibonacci