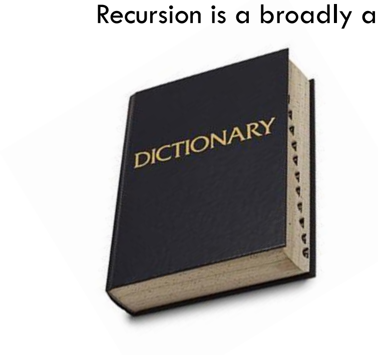
CSCI 2200 FOUNDATIONS OF COMPUTER SCIENCE

David Goldschmidt
goldsd3@rpi.edu
Fall 2022

RECURSION

…to understand recursion, you must first understand recursion…

Recursion is a broadly applicable technique that involves a self-reference…



look-up(w_0): find the definition of word w_0 in the dictionary;
if the definition has unknown words w_1, w_2, …, w_n,
call look-up(w_1), look-up(w_2), …, look-up(w_n)

Will the recursion here ever stop?

The recursive look-up() function works if there are known words to which everything else reduces—i.e., one or more base cases

RECURSION

An example of a recursive function…

$$f(n) = f(n-1) + 3n - 2$$

What is $f(2)$…?

$$f(2) = f(1) + 4 = f(0) + 5 = f(-1) + 3 = \ldots \text{ ????}$$

We must define a stopping point, i.e., a **base case** for recursion

RECURSION – WE NEED A BASE CASE

We must define a stopping point, i.e., a **base case** for recursion

A *well-defined* recursive function…

$$f(n) = \begin{cases} 13 & n = 0 \\ & \text{(or } n \leq 0\text{)} \\ f(n-1) + 3n - 2 & n > 0 \end{cases}$$

What is $f(2)$…?

$$f(2) = f(1) + 4 = f(0) + 5 = 13 + 5 = 18$$

We have a well-defined answer!

RECURSION – EXAMPLE WITH TWO BASE CASES

A well-defined recursive function for the *n*th Fibonacci number…

$$F(n) = \begin{cases} 0 & n = 0 \\ 1 & n = 1 \\ F(n-1) + F(n-2) & n > 1 \end{cases}$$

What is $F(3)$…?

$$F(3) = F(2) + F(1) = F(1) + F(0) + 1 = 1 + 0 + 1 = 2$$

What is $F(4)$…?

Notice the repetition…

RECURSIVE PROGRESS

Is this recursive function well-defined…?

$$f(n) = \begin{cases} 13 & n = 0 \\ & \text{(or } n \leq 0\text{)} \\ f(n+1) + 3n & n > 0 \end{cases}$$

What is $f(2)$…?

$$f(2) = f(3) + 6 = f(4) + 15 = f(5) + 27 = \ldots \quad ????$$

To compute $f(n)$, at each iteration, we must move *strictly closer* to base case $f(0)$

Do Exercise 7.3…

RECURSION AND INDUCTION

Induction and recursion share a similar structure…

Induction

Base case: $P(0)$ is **T**

Induction step: show $P(n) \to P(n+1)$

$\therefore$ $P(n)$ is **T** for all $n \geq 0$

$\boxed{P(0)} \to P(1) \to P(2) \to P(3) \to \ldots$

Recursion

Base case: $f(0) = 0$

Recursive function: $f(n) = f(n-1) + 2n - 1$

$\therefore$ we can compute $f(n)$ for all $n \geq 0$

$\boxed{f(0)} \to f(1) \to f(2) \to f(3) \to \ldots$

What does $P(n) \to P(n+1)$ mean here?

We can compute $f(n+1)$ if $f(n)$ is known…

UNFOLDING THE RECURSION

Consider the following well-defined recursive function…

$$f(n) = \begin{cases} 0 & n = 0 \\ & \text{(or } n \leq 0\text{)} \\ f(n-1) + 2n - 1 & n > 0 \end{cases}$$

Tinker and try to come up with another way to write $f(n)$ that removes the recursion…

UNFOLDING THE RECURSION

$$f(n) = \begin{cases} 0 & n = 0 \\ & \text{(or } n \leq 0\text{)} \\ f(n-1) + 2n - 1 & n > 0 \end{cases}$$

1. Obtain $f(n)$ from $f(n-1)$
2. Below this, obtain $f(n-1)$ from $f(n-2)$
3. Repeat the pattern down to the base case…
4. Equate the sum of all of the LHS terms with the sum of all of the RHS terms…
   (we also might use the product instead of the sum)
5. Every LHS term except for $f(n)$ cancels with a corresponding RHS term—and $f(0) = 0$

We have a conjecture that needs a proof…

Notice we have ($\frac{1}{2} \times n$) terms here!

$$\begin{aligned}
f(n) &= \cancel{f(n-1)} + 2n - 1 \\
\cancel{f(n-1)} &= \cancel{f(n-2)} + 2n - 3 \\
\cancel{f(n-2)} &= \cancel{f(n-3)} + 2n - 5 \\
&\vdots \qquad \vdots \\
\cancel{f(3)} &= \cancel{f(2)} + 5 \\
\cancel{f(2)} &= \cancel{f(1)} + 3 \\
+ \quad \cancel{f(1)} &= f(0) + 1 \\
\hline
f(n) &= 1 + 3 + \ldots + 2n - 1
\end{aligned}$$

Gauss's trick… $f(n) = \frac{1}{2} \times n \times 2n = n^2$

UNFOLDING THE RECURSION

Given $g(1) = 1$ and $g(n) = 2 \times g(n-1)$ for $n > 1$.

Unfold the recursion to derive a simpler form of $g(n)$.

Given $h(0) = 1$ and $h(n) = n \times h(n-1)$ for $n > 0$.

Unfold the recursion to derive a simpler form of $h(n)$.

UNFOLDING THE RECURSION

Given $g(1) = 1$ and $g(n) = 2 \times g(n-1)$ for $n > 1$.

Unfold the recursion to derive a simpler form of $g(n)$.

Given $h(0) = 1$ and $h(n) = n \times h(n-1)$ for $n > 0$.

Unfold the recursion to derive a simpler form of $h(n)$.

$$
\begin{array}{rrcl}
 & g(n) &=& 2 \times \cancel{g(n-1)} \\
 & \cancel{g(n-1)} &=& 2 \times \cancel{g(n-2)} \\
 & \cancel{g(n-2)} &=& 2 \times \cancel{g(n-3)} \\
 & \vdots & & \vdots \\
 & \cancel{g(4)} &=& 2 \times \cancel{g(3)} \\
 & \cancel{g(3)} &=& 2 \times \cancel{g(2)} \\
\times & \cancel{g(2)} &=& 2 \times g(1) \\
\hline
 & g(n) &=& 2 \times 2 \times \ldots \times 2 \\
 & g(n) &=& 2^{n-1}
\end{array}
$$

We have a conjecture that needs a proof...

PROVING OUR CONJECTURE

$$
f(n) = \begin{cases} 0 & n = 0 \\ & \text{(or } n \le 0\text{)} \\ f(n-1) + 2n - 1 & n > 0 \end{cases}
$$

Can we prove our claim $P(n)$ that $f(n) = n^2$ (thereby removing the recursion)?

Proof. We prove by induction that $P(n)$ is **T** for $n \geq 0$.

1. **[Base case]** $P(0)$ claims $f(0) = 0^2 = 0$, which is **T** from the recursive definition.
2. **[Induction step]** We prove $P(n) \rightarrow P(n+1)$ for all $n \geq 0$ via a direct proof.

   Assume $f(n) = n^2$; we must prove that $f(n+1) = (n+1)^2$.

   LHS: $f(n+1) = \underbrace{f(n)}_{\text{induction hypothesis}} + 2(n+1) - 1$ from the recursive definition of $f(n)$

   $= n^2 + 2n + 1$

   $= (n+1)^2$, as was to be shown.
3. By induction, $P(n)$ is **T** for all $n \geq 0$. ■

Now prove the conjectures from the previous slide...

WHAT NEXT…?

Be patient as Exam 1 is graded—we will have grades and review solutions next week

Problem Set 4 is due at recitations on October 12

- Will cover recursion and proofs with recursive objects (Chapter 7)

Homework 3 will be posted next week, due by 11:59PM on October 20

Practice! Tinker! Practice! Tinker! Practice! Tinker! Practice! Tinker! Practice!