CSCI 2200 FOUNDATIONS OF COMPUTER SCIENCE

David Goldschmidt
goldsd3@rpi.edu
Fall 2022

EXAM 1 – SOLUTIONS AND GRADES

Exam 1 average: 33.4/50 (~67%) with a standard deviation of 7.89

Exam 1 multiple choice average: 24.8/36 (~69%) with a standard deviation of 5.78

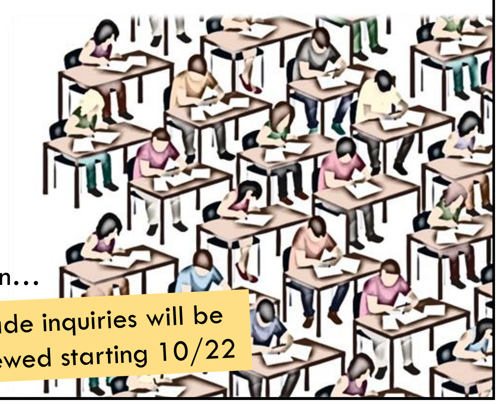
Exam 1 questions and solutions are posted in Submitty

No curve will be applied, but…

…final grade cutoffs will be adjusted at semester's end

Grade inquiries for Exam 1 (also HWs 1 and 2) are open…

…with a deadline of 11:59PM on Friday 10/21



Grade inquiries will be reviewed starting 10/22

UNFOLDING THE RECURSION – TOUGHER EXAMPLE

$$f(n) = \begin{cases} 1 & n = 1 \\ f(n/2) + 1 & n > 1\text{, even} \\ f(n+1) & n > 1\text{, odd} \end{cases}$$

n	1	2	3	4	5	6	7	8	9	10	11	12	13	…
$f(n)$	1 →	2	3 ←	3	4 ←	4	4 ←	4	5 ←	5	5 ←	5	5 ←	…

etc. etc. etc.

Arrows show how $f(n)$ is obtained, e.g., $f(2) \leftarrow f(1)$ and $f(3) \leftarrow f(4) \leftarrow f(2) \leftarrow f(1)$

Is there a path to every n from base case $n = 1$, making $f(n)$ well-defined?

Yes! And what pattern emerges…? How can we rewrite $f(n)$ here…?

UNFOLDING THE RECURSION – TOUGHER EXAMPLE

$$f(n) = \begin{cases} 1 & n = 1 \\ f(n/2) + 1 & n > 1\text{, even} \\ f(n+1) & n > 1\text{, odd} \end{cases}$$

n	1	2	3	4	5	6	7	8	9	10	11	12	13	…
$f(n)$	1	2	3	3	4	4	4	4	5	5	5	5	5	…

The pattern is logarithmic—we propose that $f(n) = 1 + \lceil \log_2 n \rceil$

Tinker with a few values to confirm that it appears to work…

…try $f(1) = 1 + \lceil \log_2 1 \rceil = 1 + 0 = 1$

…and $f(5) = 1 + \lceil \log_2 5 \rceil = 1 + 3 = 4$

We have a conjecture that needs a proof…

PROVING OUR CONJECTURE

$$f(n) = \begin{cases} 1 & n = 1 \\ f(n/2) + 1 & n > 1, \text{even} \\ f(n+1) & n > 1, \text{odd} \end{cases}$$

Can we prove our claim $P(n)$ that $f(n) = 1 + \lceil \log_2 n \rceil$ for all $n \geq 1$?

Proof. We prove by strong induction that $P(n)$ is **T** for $n \geq 1$.

1. **[Base case]** $P(1)$ claims $f(1) = 1 + \lceil \log_2 1 \rceil = 1 + 0 = 1$, which is **T**.
2. **[Induction step]** We prove $P(1) \wedge P(2) \wedge \ldots \wedge P(n) \rightarrow P(n+1)$ for $n \geq 1$.

   Assume $f(k) = 1 + \lceil \log_2 k \rceil$ for $1 \leq k \leq n$.

   We must prove $f(n+1) = 1 + \lceil \log_2 (n+1) \rceil$.

   Case 1. When $n + 1$ is even…

   Case 2. When $n + 1$ is odd…

Complete this (somewhat difficult) proof…

PROVING OUR CONJECTURE

$$f(n) = \begin{cases} 1 & n = 1 \\ f(n/2) + 1 & n > 1, \text{even} \\ f(n+1) & n > 1, \text{odd} \end{cases}$$

2. **[Induction step]** We prove $P(1) \wedge P(2) \wedge \ldots \wedge P(n) \rightarrow P(n+1)$ for $n \geq 1$.

   Assume $f(k) = 1 + \lceil \log_2 k \rceil$ for $1 \leq k \leq n$.

   We must prove $f(n+1) = 1 + \lceil \log_2 (n+1) \rceil$.

   Case 1. When $n + 1$ is even, $f(n+1) = 1 + f(\frac{n+1}{2})$.

   From our strong induction claim… $= 1 + 1 + \lceil \log_2 (\frac{n+1}{2}) \rceil$.

   Using properties of log… $= 1 + 1 + \lceil \log_2 (n+1) - 1 \rceil$.

   And this simplifies down to… $= 1 + \lceil \log_2 (n+1) \rceil$, as was to be shown.

PROVING OUR CONJECTURE

$$f(n) = \begin{cases} 1 & n = 1 \\ f(n/2) + 1 & n > 1\text{, even} \\ f(n+1) & n > 1\text{, odd} \end{cases}$$

Case 2. When $n + 1$ is odd, $n \geq 2$ and we have

$$f(n+1) = f(n+2) \underbrace{= 1 + f(\frac{n+2}{2})}_{\text{how do we know } P(\frac{n+2}{2}) \text{ is true?}}.$$

With $n \geq 2$, observe that $\frac{n+2}{2} \leq \frac{n+n}{2} = n$, so $P(\frac{n+2}{2})$ is true.

Therefore, $f(\frac{n+2}{2}) = 1 + \lceil \log_2(\frac{n+2}{2}) \rceil = 1 + \lceil \log_2(n+2) - 1 \rceil = \lceil \log_2(n+2) \rceil$.

Thus, $f(n+1) = \underbrace{1 + \lceil \log_2(n+2) \rceil = 1 + \lceil \log_2(n+1) \rceil}_{\text{Lemma 7.2: } \lceil \log_2(n+1) \rceil = \lceil \log_2(n+2) \rceil \text{ for } n+1 \geq 3 \text{ odd...}}$, as was to be shown.

PROVING OUR CONJECTURE

$$f(n) = \begin{cases} 1 & n = 1 \\ f(n/2) + 1 & n > 1\text{, even} \\ f(n+1) & n > 1\text{, odd} \end{cases}$$

Case 2. When $n + 1$ is odd, $n \geq 2$ and we have

$$f(n+1) = f(n+2) \underbrace{= 1 + f(\frac{n+2}{2})}_{\text{how do we know } P(\frac{n+2}{2}) \text{ is true?}}.$$

With $n \geq 2$, observe that $\frac{n+2}{2} \leq \frac{n+n}{2} = n$, so $P(\frac{n+2}{2})$ is true.

Therefore, $f(\frac{n+2}{2}) = 1 + \lceil \log_2(\frac{n+2}{2}) \rceil = 1 + \lceil \log_2(n+2) - 1 \rceil = \lceil \log_2(n+2) \rceil$.

Thus, $f(n+1) = \underbrace{1 + \lceil \log_2(n+2) \rceil = 1 + \lceil \log_2(n+1) \rceil}_{\text{Lemma 7.2: } \lceil \log_2(n+1) \rceil = \lceil \log_2(n+2) \rceil \text{ for } n+1 \geq 3 \text{ odd...}}$, as was to be shown.

3. By induction, $P(n)$ is true for all $n > 1$. ■

RECURRENCES

...or $\mathbb{N}_0$

A *recurrence* is a recursive function defined on $\mathbb{N}$

We simplify the notation for $f(n)$ by defining a recurrence as A_n or F_n or T_n or etc.

$$A_1 = 0; \quad A_n = A_{n-1} + 2n - 1 \text{ (for } n > 1\text{)}$$

Defining Fibonacci as a recurrence is...

$$F_1 = 1; \quad F_2 = 1; \quad F_n = F_{n-1} + F_{n-2} \text{ (for } n > 2\text{)}$$

We can use recurrences to describe the runtime of recursive programs...

RECURRENCES AND RUNTIMES – EXAMPLE 2

```
int f( int n )
{
  if ( n == 0 ) return 1;
  elif ( is_even( n ) ) return f( n/2 ) * f( n/2 );
  else return 2 * f( n - 1 );
}
```

(assume *is_even*() always takes two operations)

For $n = 0$, we have $T_0 = 2$, a test and a set

For $n = 1$, we have $T_1 = T_0 + 7$, two tests, *is_even*(), two ops (×, –), a set, and $f(0)$

For $n = 2$, we have $T_2 = 2T_1 + 8$, two tests, *is_even*(), three ops (×, ÷, ÷), a set, two $f(1)$ calls

For $n = 3$, we have $T_3 = T_2 + 7$, two tests, *is_even*(), two ops (×, –), a set, and $f(2)$

For $n = 4$, we have $T_4 = 2T_2 + 8$, two tests, *is_even*(), three ops (×, ÷, ÷), a set, two $f(2)$ calls

Rewrite T_n without recursion...

...see Problem 7.44

RECURSIVE SETS

Write a recursive definition for set A that contains all non-negative multiples of 4, i.e., $A = \{ 0, 4, 8, \dots \}$

Recursion is so powerful, we can also use it to precisely construct sets…

Recursive definition of the natural numbers $\mathbb{N}$.

1. $1 \in \mathbb{N}$. **[basis]**
2. $x \in \mathbb{N} \rightarrow x + 1 \in \mathbb{N}$. **[constructor]**
3. Nothing else is in $\mathbb{N}$. **[minimality]**

$\mathbb{N} = \{ 1, 2, 3, \dots \}$

The *minimality* rule is implied and can therefore be omitted

RECURSIVE SETS

Write a recursive definition for set A that contains all non-negative multiples of 4, i.e., $A = \{ 0, 4, 8, \dots \}$

Recursively define set $A = \{ x \mid x \in \mathbb{N}_0 \text{ is a multiple of } 4 \}$

Recursive definition of set A.

1. $0 \in A$. **[basis]**
2. $x \in A \rightarrow x + 4 \in A$. **[constructor]**

What else can we say about A…?

$x \in A \rightarrow x$ is even ✓

x is even $\rightarrow x \in A$ ✗ (e.g., $x = 6$)

x is odd $\rightarrow x \notin A$ ✓ (*contrapositive!*)

Do all members of recursive set A have a given property…?

…we can use *structural induction* to prove such claims…

STRUCTURAL INDUCTION

Recursive definition of set A.
1. $0 \in A$. **[basis]**
2. $x \in A \rightarrow x + 4 \in A$. **[constructor]**

Can we prove our claim $P : x \in A \rightarrow x$ is even?

Proof. We use structural induction to prove that P is **T** for all $x \in A$.

1. **[Base case]** $P(0)$ claims that 0 is even, which is **T**.
2. **[Induction step]** Suppose $x \in A$ and x is even.

   The constructor rule creates $x + 4 \in A$.

   Adding two even numbers always produces an even number.
3. By structural induction, we conclude that every member of A is even. ■

STRUCTURAL INDUCTION

Strong induction with recursively defined sets is called *structural induction*

Given recursive set S, we have base cases $s_1, s_2, \ldots, s_k \in S$ and constructor rules

Define $P(s)$ to be a property of any element $s \in S$ (e.g., s is odd, s is a palindrome)

Proof. To show that $P(s)$ holds for <u>all</u> elements of S, we must show:

1. **[Base cases]** $P(s_1), P(s_2), \ldots, P(s_k)$ are **T**.
2. **[Induction step]** For each constructor rule…

   …if P is **T** for known *parents*, then P is **T** for generated *children*.

   Is the property *preserved* by all constructor rules?
3. By structural induction, conclude that $P(s)$ is **T** for all $s \in S$.

RECURSIVE SETS – FINITE BINARY STRINGS

Write a recursive definition for set S containing all binary string palindromes, e.g., 010, 11011, 000, …

Let ε be the *empty string* (similar to $\varnothing$ for sets)

Recursive definition of Σ^* (finite binary strings).

1. $\varepsilon \in \Sigma^*$. **[basis]**
2. $x \in \Sigma^* \rightarrow x \bullet 0 \in \Sigma^*$ and $x \bullet 1 \in \Sigma^*$. **[constructor]**

(note that $\bullet$ indicates concatenation)

$\varepsilon \rightarrow 0, 1 \rightarrow 00, 01, 10, 11 \rightarrow 000, 001, 010, 011, 100, 101, 110, 111 \rightarrow \ldots$

$\Sigma^* = \{ \varepsilon, 0, 1, 00, 01, 10, 11, 000, 001, 010, 011, 100, 101, 110, 111, \ldots \}$

RECURSIVE SETS – PALINDROMES

Recursive definition of set S_P (all finite string palindromes).

1. $\varepsilon \in S_P$, $0 \in S_P$, $1 \in S_P$. **[basis]**
2. $x \in S_P \rightarrow 0 \bullet x \bullet 0 \in S_P$; **[constructor (i)]**
   $x \in S_P \rightarrow 1 \bullet x \bullet 1 \in S_P$. **[constructor (ii)]**

Write *derivations* for 10101 and 110011 by showing each step from a base case…

$1 \rightarrow 010 \rightarrow 10101$

$\varepsilon \rightarrow 00 \rightarrow 1001 \rightarrow 110011$

How many palindromes have length 6?

RECURSIVE SETS – PALINDROMES

Recursive definition of set S_P (all finite string palindromes).

1. $\varepsilon \in S_P$, $0 \in S_P$, $1 \in S_P$. **[basis]**
2. $x \in S_P \rightarrow 0 \bullet x \bullet 0 \in S_P$; **[constructor (i)]**
   $x \in S_P \rightarrow 1 \bullet x \bullet 1 \in S_P$. **[constructor (ii)]**

Prove by structural induction that every element in S_P is a palindrome…

How many palindromes have length 6?

(the reversal of string s is denoted as s^R)

A palindrome of length 6 can be written as $x \bullet x^R$ (or just xx^R); the length of x is 3

There are $2^3 = 8$ strings of length 3, so there must be 8 palindromes of length 6

In general, there are $2^{\lceil n/2 \rceil}$ palindromes of length n…

STRUCTURAL INDUCTION

Recursive definition of set S_P.
1. $\varepsilon \in S_P$, $0 \in S_P$, $1 \in S_P$. **[basis]**
2. $x \in S_P \rightarrow 0 \bullet x \bullet 0 \in S_P$; **[constructor (i)]**
   $x \in S_P \rightarrow 1 \bullet x \bullet 1 \in S_P$. **[constructor (ii)]**

But how do we prove that every palindrome is in S_P...?

Proof. We use structural induction to prove that every element of S_P is a palindrome.

1. **[Base cases]** By definition, strings ε, 0, and 1 are palindromes.
2. **[Induction step]** We must prove that each constructor preserves *palindromicity*.

   If x is a palindrome, then $x^R = x$. [*This is essentially our induction hypothesis.*]

   For constructor (i), we must show $(0 \bullet x \bullet 0)^R = 0 \bullet x \bullet 0$.

   Here, $(0 \bullet x \bullet 0)^R = 0^R \bullet x^R \bullet 0^R = 0 \bullet \underbrace{x^R}_{\text{apply our induction hypothesis...}} \bullet 0 = 0 \bullet x \bullet 0$.

   For constructor (ii), repeat the above...
3. By structural induction, we conclude that every member of S_P is a palindrome. ■

See Exercise 8.5 to prove $(0 \bullet x \bullet 0)^R = 0^R \bullet x^R \bullet 0^R$

CONTRADICTION

Recursive definition of set S_P.
1. $\varepsilon \in S_P$, $0 \in S_P$, $1 \in S_P$. **[basis]**
2. $x \in S_P \rightarrow 0 \bullet x \bullet 0 \in S_P$; **[constructor (i)]**
   $x \in S_P \rightarrow 1 \bullet x \bullet 1 \in S_P$. **[constructor (ii)]**

Proof. We use contradiction to prove that every palindrome is in S_P.

Consider palindrome m, the shortest palindrome <u>not</u> in S_P. We have two cases.

Case 1. If m starts with 0, it must end with 0, so $m = 0 \bullet x \bullet 0$.
Then, x must be a palindrome for m to be one, meaning that $x \in S_P$.

But if $x \in S_P$, then by constructor (i), $0 \bullet x \bullet 0 = m \in S_P$, a contradiction!

Case 2. If m starts with 1, it must end with 1, so $m = 1 \bullet x \bullet 1$.
Then, x must be a palindrome for m to be one, meaning that $x \in S_P$.

But if $x \in S_P$, then by constructor (ii), $1 \bullet x \bullet 1 = m \in S_P$, a contradiction!

Given both contradictions, we conclude that there is no shortest palindrome not in S_P, i.e., every palindrome is in S_P. ■

WHAT NEXT…?

Review Exam 1 grades and solutions

- Grade inquiries due by 11:59PM on October 21

Homework 3 is due by 11:59PM on October 20

- Next week's recitations will be Q&A sessions focused on Homework 3…

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