CSCI 2200 — Foundations of Computer Science (FoCS)
Homework 3 (document version 1.2)

Overview

- This homework is due by 11:59PM on Thursday, October 20
- You may work on this homework in a group of no more than four students; unlike recitation problem sets, **your teammates may be in any section**
- You may use at most **three** late days on this assignment
- Please start this homework early and ask questions during office hours; also ask (and answer) questions on the Discussion Forum
- Please be concise in your answers; even if your solution is correct, if it is not well-presented, you may still lose points
- You can type or hand-write (or both) your solutions to the required graded problems below; **all work must be organized in one PDF that lists all teammate names**
- You are strongly encouraged to use LaTeX, in particular for mathematical symbols; see references in Course Materials
- **EARNING LATE DAYS:** for each homework that you complete using LaTeX (including any tables, graphs, etc., i.e., no hand-written anything), you earn one additional late day; you can draw graphs and other diagrams in another application and include them as image files
- To earn a late day, you must submit your LaTeX files (i.e., `*.tex`) along with your one required PDF file—please name the PDF file `hw3.pdf`
- Also note that the earned late day can be used retroactively, even back to the first homework assignment!

Warm-up exercises

The problems below are good practice problems to work on. Do not submit these as part of your homework submission. **These are ungraded problems.**

- **Problem 7.11.**
- **Problem 7.12(a-b).**
  (See Problem 7.28 for hints.)
- **Problem 7.21.**
- **Problem 7.41.**
- **Problem 7.44.**
- **Problem 7.45(a-b,d-f).**
- **Problem 7.46.**
- **Problem 7.47.**
- **Problem 8.12(a-c).**
- **Problem 8.13.**
- **Problem 8.18.**

Graded problems

The problems below are required and will be graded.

- ***Problem 7.9.**
- ***Problem 7.12(c).** (See Problem 7.28 for hints.)
- ***Problem 7.13(a).**
- ***Problem 7.19(d).**
- ***Problem 7.42.**
- ***Problem 7.45(c).**
- ***Problem 7.49.**
- ***Problem 8.12(d).**
- ***Problem 8.14.**

(v1.1) Some of the above problems (graded an ungraded) are transcribed in the pages that follow.

Graded problems are noted with an asterisk (*).

If any typos exist below, please use the textbook description.

- ***Problem 7.9.** $G_0 = 0$, $G_1 = 1$, and $G_n = 7G_{n-1} - 12G_{n-2}$ for $n > 1$. Compute G_5.
  $G_5 = 7G_4 - 12G_3$
  $G_4 = 7G_3 - 12G_2$
  $G_3 = 7G_2 - 12G_1$
  $G_2 = 7G_1 - 12G_0$
  $G_2 = 7 = 7$
  $G_3 = 7 * 7 - 12 = 37$
  $G_4 = 7 * (7 * 7 - 12) - 12 * 7 = 175$
  $G_5 = 7 * (7 * (7 * 7 - 12) - 12 * 7) - 12(7 * 7 - 12) = 781$
  Show $G_n = 4^n - 3^n$ for $n \geq 0$.
  We prove the statement with induction.
  Base case: $G_0 = 4^0 - 3^0 = 0$
  Base case: $G_1 = 4^1 - 3^1 = 1$
  Induction: assume $G_n = 4^n - 3^n$
  $7G_{n-1} - 12G_{n-2} = 4^n - 3^n$
  $7G_n - 12G_{n-1} = 4^{n+1} - 3^{n+1}$
  $7(4^n - 3^n) - 12(4^{n-1} - 3^{n-1}) = 4^{n+1} - 3^{n+1}$
  $(7 * 4^n - 7 * 3^n) - (3 * 4^n - 4 * 3^n) = 4^{n+1} - 3^{n+1}$
  $7 * 4^n - 7 * 3^n - 3 * 4^n + 4 * 3^n = 4^{n+1} - 3^{n+1}$
  $4 * 4^n - 3 * 3^n = 4^{n+1} - 3^{n+1}$
  $4^{n+1} - 3^{n+1} = 4^{n+1} - 3^{n+1}$
  therefore $G_n = 4^n - 3^n$ for $n \geq 0$. ■

- **Problem 7.11.** In each case tinker. Then, guess a formula that solves the recurrence, and prove it.

  (a) $P_0 = 0$, $P_1 = a$, and $P_n = 2P_{n-1} - P_{n-2}$, for $n > 1$.
  (b) $G_1 = 1$; $G_n = (1 - 1/n) \cdot G_{n-1}$, for $n > 1$.

- **Problem 7.12(a-b).** (See Problem 7.28 for hints.) Tinker to guess a formula for each recurrence and prove it. In each case, $A_1 = 1$ and for $n > 1$:

  (a) $A_n = 10A_{n-1} + 1$.
  (b) $A_n = nA_{n-1}/(n - 1) + n$.

- ***Problem 7.12(c).** (See Problem 7.28 for hints.) Tinker to guess a formula for each recurrence and prove it. In each case, $A_1 = 1$ and for $n > 1$:

  (c) $A_n = 10nA_{n-1}/(n - 1) + n = \frac{10nA_{n-1}}{n-1} + n$.
  $A_n = \frac{n(10^n-1)}{9}$
  We prove this with induction.
  Base case: $A_1 = \frac{1(10^1-1)}{9} = 1$
  Incuction: assume $A_n = \frac{n(10^n-1)}{9}$
  $\frac{10nA_{n-1}}{n-1} + n = \frac{n(10^n-1)}{9}$
  $\frac{10(n+1)A_n}{n} + (n + 1) = \frac{(n+1)(10^{n+1}-1)}{9}$
  $\frac{10(n+1)\frac{n(10^n-1)}{9}}{n} + (n + 1) = \frac{(n+1)(10^{n+1}-1)}{9}$

$\frac{10(n+1)n(10^n-1)}{9n} + (n+1) = \frac{(n+1)(10^{n+1}-1)}{9}$
$\frac{10(n+1)(10^n-1)}{9} + (n+1) = \frac{(n+1)(10^{n+1}-1)}{9}$
$\frac{10(n+1)(10^n-1)+9(n+1)}{9} = \frac{(n+1)(10^{n+1}-1)}{9}$
$\frac{(n+1)(10^{n+1}-10)+9(n+1)}{9} = \frac{(n+1)(10^{n+1}-1)}{9}$
$(n+1)\frac{(10^{n+1}-10)+9}{9} = \frac{(n+1)(10^{n+1}-1)}{9}$
$(n+1)\frac{10^{n+1}-1}{9} = \frac{(n+1)(10^{n+1}-1)}{9}$
$\frac{(n+1)(10^{n+1}-1)}{9} = \frac{(n+1)(10^{n+1}-1)}{9}$
therefore $A_n = \frac{n(10^n-1)}{9}$ for ■

- ***Problem 7.13(a).** Analyze these very fast-growing recursions. [Hint: Take logarithms.]

  (a) $M_1 = 2$ and $M_n = aM_{n-1}^2$ for $n > 1$. Guess and prove a formula for M_n. Tinker, tinker.
  $2^{2^{n-1}} * a^{2^n-1}$

- ***Problem 7.19(d).** Recall the Fibonacci numbers: $F_1, F_2 = 1$; and, $F_n = F_{n-1} + F_{n-2}$ for $n > 2$.

  (d) Prove that every third Fibonacci number, F_{3n}, is even.
  We prove this with induction.
  let $j, k \in \mathbb{N}_0$ so $2k$ is even and $2k+1$ is odd.
  Base: $F_3 = F_2 + F_1 = 1 + 1 = 2 = 2k$ is even.
  Induction: assume every third F_n is even so $F_{3n} = 2k$
  $F_{3n} = F_{3n-1} + F_{3n-2}$
  $F_{3(n+1)} = F_{3(n+1)-1} + F_{3(n+1)-2}$
  $F_{3k} = F_{3n+2} + F_{3n+1}$
  $2k = F_{3n+2} + F_{3n+1}$
  $2k = F_{3n+1} + F_{3n} + F_{3n+1}$
  $2k = 2F_{3n+1} + 2j$
  $2k = 2(F_{3n+1} + j)$
  therefore F_{3n} is even ■

- **Problem 7.21.** Show that every $n \geq 1$ is a sum of distinct Fibonacci numbers, e.g., $11 = F_4+F_6$; $20 = F_3 = F_5+F_7$. (There can be many ways to do it, e.g., $6 = F_1+F_5 = F_2+F_3+F_4$.) [Hints: Greedy algorithm; strong induction.]

- **Problem 7.41.** Refer to the pseudocode on the right.

```
out=S([arr],i,j)
 if(j<i) out=0;
 else
  out=arr[j]+S([arr],i,j-1);
```

  (a) What is the function being implemented?
  (b) Prove that the output is correct for every valid input.
  (c) Give a recurrence for the runtime T_n, where $n = j - i$.

(d) Guess and prove a formula for T_n.

- ***Problem 7.42.** Give pseudocode for a recursive function that computes 3^{2^n} on input n.
  (a) Prove that your function correctly computes 3^{2^n} for every $n \geq 0$.
  (b) Obtain a recurrence for the runtime T_n. Guess and prove a formula for T_n.

- **Problem 7.44.** We give two implementations of `Big(n)` from page 90 (`iseven(n)` tests if n is even).

(a)
```
out=Big(n)
 if(n==0) out=1;
 elseif(iseven(n))
    out=Big(n/2)*Big(n/2);
 else out=2*Big(n-1)
```

(b)
```
out=Big(n)
 if(n==0) out=1;
 elseif(iseven(n))
    tmp=Big(n/2); out=tmp*tmp;
 else out=2*Big(n-1)
```

(i) For each, prove that the output is 2^n and give a recurrence for the runtime T_n. (`iseven(n)` is two operations.)
(ii) For each, compute runtimes T_n for $n = 1, \ldots, 10$. Compare runtimes with Exercise 7.10 on page 90.

- ***Problem 7.45(c).** Give recursive definitions for the set $\mathcal{S}$ in each of the following cases.
  (c) $\mathcal{S} = \{$ all strings with the same number of 0's and 1's $\}$ (e.g., 0011, 0101, 100101).

- **Problem 7.45(a-b,d-f).** Give recursive definitions for the set $\mathcal{S}$ in each of the following cases.
  (a) $\mathcal{S} = \{0, 3, 6, 9, 12, \ldots\}$, the multiples of 3.
  (b) $\mathcal{S} = \{1, 2, 3, 4, 6, 7, 8, 9, 11, \ldots\}$, the numbers which are not multiples of 5.
  (d) The set of odd multiples of 3.
  (e) The set of binary strings with an even number of 0's.
  (f) The set of binary strings of even length.

- **Problem 7.46.** What is the set $\mathcal{A}$ defined recursively as shown? (By default, nothing else is in $\mathcal{A}$—minimality.)
  (1) $1 \in \mathcal{A}$
  (2) $x, y \in \mathcal{A} \to x + y \in \mathcal{A}$
  $x, y \in \mathcal{A} \to x - y \in \mathcal{A}$

- **Problem 7.47.** What is the set $\mathcal{A}$ defined recursively as shown? (By default, nothing else is in $\mathcal{A}$—minimality.)
  (1) $3 \in \mathcal{A}$

(2) $x, y \in \mathcal{A} \to x + y \in \mathcal{A}$
$x, y \in \mathcal{A} \to x - y \in \mathcal{A}$

- ***Problem 7.49.** There are 5 rooted binary trees (RBTs) with 3 nodes. How many have 4 nodes?
  12

- **Problem 8.12(a-c).** A set $\mathcal{P}$ of parenthesis strings has a recursive definition (right).

  (1) $\varepsilon \in \mathcal{P}$
  (2) $x \in \mathcal{P} \to [x] \in \mathcal{P}$
  $x, y \in \mathcal{P} \to xy \in \mathcal{P}$

  (a) Determine if each string is in $\mathcal{P}$ and give a derivation if it is in $\mathcal{P}$.
  (i) [[[]]][(ii) [][[]][[]] (iii) [[][][]
  (b) Give two derivations of [][][[]] whose steps are not a simple reordering of each other.
  (c) Prove by structural induction that every string in $\mathcal{P}$ has even length.

- ***Problem 8.12(d).** A set $\mathcal{P}$ of parenthesis strings has a recursive definition (right).

  (1) $\varepsilon \in \mathcal{P}$
  (2) $x \in \mathcal{P} \to [x] \in \mathcal{P}$
  $x, y \in \mathcal{P} \to xy \in \mathcal{P}$

  (d) Prove by structural induction that every string in $\mathcal{P}$ is balanced.

- **Problem 8.13.** Recursively define the binary strings that contain more 0's than 1's. Prove:

  (a) Every string in your set has more 0's than 1's.
  (b) Every string which has more 0's than 1's is in your set.

- ***Problem 8.14.** A set $\mathcal{A}$ is defined recursively as shown.

  (1) $3 \in \mathcal{A}$
  (2) $x, y \in \mathcal{A} \to x + y \in \mathcal{A}$
  $x, y \in \mathcal{A} \to x - y \in \mathcal{A}$

  (a) Prove that every element of $\mathcal{A}$ is a multiple of 3.
  (b) Prove that every multiple of 3 is in $\mathcal{A}$.

- **Problem 8.18.** Recursively define rooted binary trees (RBTs) and rooted full binary trees (RFBTs).

  (a) Give examples, with derivations, of RBTs and RFBTs with 5, 6, and 7 vertices.
  (b) Prove by structural induction that every RFBT has an odd number of vertices.