

# CS/ECE 374 ✧ Spring 2021

## 🌀 Homework 0 🌀

Due Wednesday, Feb 3, 2021 at 10am

---

- **Each student must submit individual solutions for this homework.** For all future homeworks, groups of up to three students can submit joint solutions.
  - **Submit your solutions electronically on the course Gradescope site as PDF files.** Submit a separate PDF file for each numbered problem. If you plan to typeset your solutions, please use the  $\text{\LaTeX}$  solution template on the course web site. If you must submit scanned handwritten solutions, please use a black pen on blank white paper and a high-quality scanner app (or an actual scanner, not just a phone camera).
- 

### 👉 **Some important course policies** 👈

- **You may use any source at your disposal**—paper, electronic, or human—but you *must* cite *every* source that you use, and you *must* write everything yourself in your own words. See the academic integrity policies on the course web site for more details.
  - Unlike some previous semesters we will **not** have the “I Don’t Know (IDK)” policy this semester for home works or exams.
  - **Avoid the Three Deadly Sins!** Any homework or exam solution that breaks any of the following rules will be given an **automatic zero**, unless the solution is otherwise perfect. Yes, we really mean it. We’re not trying to be scary or petty (Honest!), but we do want to break a few common bad habits that seriously impede mastery of the course material.
    - Always give complete solutions, not just examples.
    - Always declare all your variables, in English. In particular, always describe the specific problem your algorithm is supposed to solve.
    - Never use weak induction.
- 

### **See the course web site for more information.**

If you have any questions about these policies, please don’t hesitate to ask in class, in office hours, or on Piazza.

---

0. Read the course policies and the instructions for this home work before you start.

1. Consider the following recurrence.

$$T(n) = T(\lfloor n/2 \rfloor) + 2T(\lfloor n/8 \rfloor) + n \quad n \geq 8, \text{ and } T(n) = 1 \quad 1 \leq n < 8.$$

One can prove that  $T(n) = O(n)$ . Do you see that  $T(n) = \Omega(n)$ ? The goal of this problem is to show a more general statement and to refresh your induction skills.

Let  $c_1, c_2, c_3$  be rational numbers such that  $0 < c_1 \leq c_2 \leq c_3$  and  $c_1 + c_2 + c_3 < 1$ . Let  $\alpha > 0$ . Consider the recurrence

$$T(n) = T(\lfloor c_1 n \rfloor) + T(\lfloor c_2 n \rfloor) + T(\lfloor c_3 n \rfloor) + \alpha n, \quad n > 1/c_1, T(n) = 1 \quad n \leq 1/c_1.$$

- Prove by induction that  $T(n) = O(n)$ . More precisely show that  $T(n) \leq \alpha n + b$  for  $n \geq 1$  where  $\alpha, b \geq 0$  are some fixed but suitably chosen constants (you get to choose and fix them based on  $c_1, c_2, c_3, \alpha$ ). You may first want to try the concrete recurrence at the start of the problem. How does  $\alpha$  depend on  $c_1, c_2, c_3$ ?
- Consider the recursion tree for the recurrence. What is an asymptotic upper bound on the depth of the recursion tree? Express this as a function of  $n$  and  $c_1, c_2, c_3$ . You do not need to prove correctness of your bound.
- We now consider a somewhat more general setting. Let  $0 < c_1 \leq c_2 \leq \dots \leq c_k < 1$  be  $k$  rationals such that  $\sum_{i=1}^k c_i < 1$ . And  $\alpha > 0$ . Suppose we have a recurrence of the form

$$T(n) = \sum_{i=1}^k T(\lfloor c_i n \rfloor) + \alpha n, \quad n \geq 1/c_1, T(n) = 1 \quad n \leq 1/c_1.$$

You can show that  $T(n) = O(n)$  via induction as in the simpler case when  $k = 3$ . State the bound for  $\alpha$  in this more general setting and also the depth of the recursion as a function of  $n, c_1, c_2, \dots, c_k$ . You do not need to prove correctness of your bound.

2. Consider the set of strings  $L_1 \subseteq \{0, 1\}^*$  defined recursively as follows:

- The string  $\varepsilon$  is in  $L_1$ .
- For any string  $x$  in  $L_1$ , the string  $0x1$  is also in  $L_1$ .
- For any string  $x$  in  $L_1$ , the string  $1x0$  is also in  $L_1$ .
- These are the only strings in  $L_1$ .

Let  $\#(a, w)$  denote the number of times symbol  $a$  appears in string  $w$ ; for example,

$$\#(0, 101110101101011) = 5 \quad \text{and} \quad \#(1, 101110101101011) = 10.$$

(a) Let  $L_2$  be the set of binary strings that have equal number of  $0$ 's and  $1$ 's. That is  $L_2 = \{x \in \{0, 1\}^* \mid \#(0, x) = \#(1, x)\}$ .

Prove via induction that  $L_1 \subseteq L_2$ .

(b) Give a counterexample for the statement  $L_2 \subseteq L_1$ .

(c) Now consider the set of strings  $L_3 \subseteq \{0, 1\}^*$  defined recursively as follows:

- The string  $\varepsilon$  is in  $L_3$ .
- For any string  $x$  in  $L_3$ , the string  $0x1$  is also in  $L_3$ .
- For any string  $x$  in  $L_3$ , the string  $1x0$  is also in  $L_3$ .
- For any strings  $x$  and  $y$  in  $L_3$ , the string  $xy$  is also in  $L_3$ .
- These are the only strings in  $L_3$ .

Prove by induction that  $L_2 \subseteq L_3$ . You don't have to prove it for submission but you can generalize the first part to prove that  $L_3 \subseteq L_2$  and hence  $L_2 = L_3$ .

You may assume without proof that  $\#(a, uv) = \#(a, u) + \#(a, v)$  for any symbol  $a$  and any strings  $u$  and  $v$ , or any other result proved in class, in lab, or in the lecture notes. Otherwise, your proofs must be formal and self-contained.

3. **For your reading, do not submit for grading:** Suppose  $S$  is a set of 103 integers. Prove that there is a subset  $S' \subseteq S$  of at least 15 numbers such that the difference of any two numbers in  $S'$  is a multiple of 7. *Hint:* See solved problem.

4. **For your reading, do not submit for grading:** Let  $\Sigma$  be a *finite* alphabet and let  $\mathcal{L}$  be the set of all *finite* languages over  $\Sigma$ . Prove that  $\mathcal{L}$  is countable. Note that Cantor's diagonalization argument (review CS 173 material if you have forgotten about countability) shows that if  $|\Sigma| \geq 2$  the set of all languages over  $\Sigma$  is *not* countable.

Each homework assignment will include at least one solved problem, similar to the problems assigned in that homework, together with the grading rubric we would apply *if* this problem appeared on a homework or exam. These model solutions illustrate our recommendations for structure, presentation, and level of detail in your homework solutions. Of course, the actual *content* of your solutions won't match the model solutions, because your problems are different!

## Solved Problems

1. Suppose  $S$  is a set of  $n + 1$  integers. Prove that there exist distinct numbers  $x, y \in S$  such that  $x - y$  is a multiple of  $n$ .

**Solution:** We will use the pigeon hole principle. Let the  $n + 1$  numbers in  $S$  be  $a_1, a_2, \dots, a_{n+1}$  and consider  $b_1, b_2, \dots, b_{n+1}$  where  $b_i = a_i \bmod n$ . Note that each  $b_i$  belongs to the set  $\{0, 1, \dots, n-1\}$ . By the pigeon hole principle we must have two numbers  $b_i$  and  $b_j$ ,  $i \neq j$  such that  $b_i = b_j$ . This implies that  $a_i \bmod n = a_j \bmod n$  and hence  $a_i - a_j$  is divisible by  $n$ .

**Rubric:** 2 points for recognizing that the pigeon hole principle can be used. 2 points for the idea of using  $\bmod n$ . 6 points for a full correct proof. Any other correct proof would also fetch 10 points.

■

2. Recall that the **reversal**  $w^R$  of a string  $w$  is defined recursively as follows:

$$w^R := \begin{cases} \varepsilon & \text{if } w = \varepsilon \\ x^R \cdot a & \text{if } w = a \cdot x \end{cases}$$

A **palindrome** is any string that is equal to its reversal, like **AMANAPLANACANALPANAMA**, **RACECAR**, **POOP**, **I**, and the empty string.

- Give a recursive definition of a palindrome over the alphabet  $\Sigma$ .
- Prove  $w = w^R$  for every palindrome  $w$  (according to your recursive definition).
- Prove that every string  $w$  such that  $w = w^R$  is a palindrome (according to your recursive definition).

In parts (b) and (c), you may assume without proof that  $(x \cdot y)^R = y^R \cdot x^R$  and  $(x^R)^R = x$  for all strings  $x$  and  $y$ .

**Solution:**

- A string  $w \in \Sigma^*$  is a palindrome if and only if either
  - $w = \varepsilon$ , or
  - $w = a$  for some symbol  $a \in \Sigma$ , or
  - $w = axa$  for some symbol  $a \in \Sigma$  and some *palindrome*  $x \in \Sigma^*$ .

**Rubric:** 2 points =  $\frac{1}{2}$  for each base case + 1 for the recursive case. No credit for the rest of the problem unless this is correct.

- Let  $w$  be an arbitrary palindrome.

Assume that  $x = x^R$  for every palindrome  $x$  such that  $|x| < |w|$ .

There are three cases to consider (mirroring the three cases in the definition):

- If  $w = \varepsilon$ , then  $w^R = \varepsilon$  by definition, so  $w = w^R$ .
- If  $w = a$  for some symbol  $a \in \Sigma$ , then  $w^R = a$  by definition, so  $w = w^R$ .

- Suppose  $w = axa$  for some symbol  $a \in \Sigma$  and some palindrome  $x \in P$ . Then

$$\begin{aligned}
 w^R &= (a \cdot x \cdot a)^R \\
 &= (x \cdot a)^R \cdot a && \text{by definition of reversal} \\
 &= a^R \cdot x^R \cdot a && \text{You said we could assume this.} \\
 &= a \cdot x^R \cdot a && \text{by definition of reversal} \\
 &= a \cdot x \cdot a && \text{by the inductive hypothesis} \\
 &= w && \text{by assumption}
 \end{aligned}$$

In all three cases, we conclude that  $w = w^R$ .

**Rubric:** 4 points: standard induction rubric (scaled)

- (c) Let  $w$  be an arbitrary string such that  $w = w^R$ .

Assume that every string  $x$  such that  $|x| < |w|$  and  $x = x^R$  is a palindrome.

There are three cases to consider (mirroring the definition of “palindrome”):

- If  $w = \varepsilon$ , then  $w$  is a palindrome by definition.
- If  $w = a$  for some symbol  $a \in \Sigma$ , then  $w$  is a palindrome by definition.
- Otherwise, we have  $w = ax$  for some symbol  $a$  and some *non-empty* string  $x$ .  
The definition of reversal implies that  $w^R = (ax)^R = x^R a$ .  
Because  $x$  is non-empty, its reversal  $x^R$  is also non-empty.  
Thus,  $x^R = by$  for some symbol  $b$  and some string  $y$ .  
It follows that  $w^R = bya$ , and therefore  $w = (w^R)^R = (bya)^R = ay^R b$ .

[At this point, we need to prove that  $a = b$  and that  $y$  is a palindrome.]

Our assumption that  $w = w^R$  implies that  $bya = ay^R b$ .

The recursive definition of string equality immediately implies  $a = b$ .

Because  $a = b$ , we have  $w = ay^R a$  and  $w^R = aya$ .

The recursive definition of string equality implies  $y^R a = ya$ .

It immediately follows that  $(y^R a)^R = (ya)^R$ .

Known properties of reversal imply  $(y^R a)^R = a(y^R)^R = ay$  and  $(ya)^R = ay^R$ .

It follows that  $ay^R = ay$ , and therefore  $y = y^R$ .

The inductive hypothesis now implies that  $y$  is a palindrome.

We conclude that  $w$  is a palindrome by definition.

In all three cases, we conclude that  $w$  is a palindrome.

**Rubric:** 4 points: standard induction rubric (scaled).

- No penalty for jumping from  $aya = ay^R a$  directly to  $y = y^R$ .



**Rubric (induction):** For problems worth 10 points:

- + 1 for explicitly considering an *arbitrary* object
- + 2 for a valid **strong** induction hypothesis
  - **Deadly Sin!** Automatic zero for stating a weak induction hypothesis, unless the rest of the proof is *perfect*.
- + 2 for explicit exhaustive case analysis
  - No credit here if the case analysis omits an infinite number of objects. (For example: all odd-length palindromes.)
  - –1 if the case analysis omits a finite number of objects. (For example: the empty string.)
  - –1 for making the reader infer the case conditions. Spell them out!
  - No penalty if cases overlap (for example:
- + 1 for cases that do not invoke the inductive hypothesis (“base cases”)
  - No credit here if one or more “base cases” are missing.
- + 2 for correctly applying the *stated* inductive hypothesis
  - No credit here for applying a *different* inductive hypothesis, even if that different inductive hypothesis would be valid.
- + 2 for other details in cases that invoke the inductive hypothesis (“inductive cases”)
  - No credit here if one or more “inductive cases” are missing.