CS/ECE 374 ✧ Fall 2019 / Section B
Homework 0
Due Tuesday, September 2, 2019 at 8pm

- **Each student must submit individual solutions for this homework.** For all future homeworks, groups of up to three students can submit joint solutions.
- **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.
- There is no "I Don't Know (IDK)" policy in this section. (If you don't know what the "I Don't Know" policy is, you do not need to worry about it.)

1. **String digit sums.** Consider strings over the alphabet $\Sigma = \{\mathtt{0}, \mathtt{1}, \mathtt{2}, \mathtt{3}, \mathtt{4}, \mathtt{5}, \mathtt{6}, \mathtt{7}, \mathtt{8}, \mathtt{9}\}$. We will recursively define the digsum function as follows:
   - $\mathsf{digsum}(\epsilon) = 0$
   - $\mathsf{digsum}(ax) = a + \mathsf{digsum}(x)$, where $a \in \Sigma$ is interpreted as the numeric value of the digit.

   For example, $\mathsf{digsum}(\mathtt{374}) = 3 + 7 + 4 = 14$

   (a) Prove that $\mathsf{digsum}(x \cdot y) = \mathsf{digsum}(x) + \mathsf{digsum}(y)$. You may use the fact that $\#(a, x \cdot y) = \#(a, x) + \#(a, y)$, where $\#(a, x)$ is the number of occurences of the symbol a in string x, as discussed in the lecture notes.

   (b) Prove that $\mathsf{digsum}(x^R) = \mathsf{digsum}(x)$. You can use any of the results proved in lab 1 in this proof.

2. **Just can't even.** Consider a language L_{odd} defined as follows:
   - $a \in L_{\text{odd}}$ for $a \in \{\mathtt{1}, \mathtt{3}, \mathtt{5}, \mathtt{7}, \mathtt{9}\}$
   - $ax \in L_{\text{odd}}$ for $a \in \{\mathtt{0}, \mathtt{2}, \mathtt{4}, \mathtt{6}, \mathtt{8}\}$ and $x \in L_{\text{odd}}$
   - $axb \in L_{\text{odd}}$ for $a, b \in \{\mathtt{1}, \mathtt{3}, \mathtt{5}, \mathtt{7}, \mathtt{9}\}$ and $x \in L_{\text{odd}}$

   (a) Prove that $\mathtt{374}$ is *not* in L_{odd}

   (b) Prove that for any $x \in L_{\text{odd}}$, $\mathsf{digsum}(x)$ is odd.

   (c) (Not for submission) Prove that any string with $\mathsf{digsum}(x)$ odd is in L.

3. **Good things come in threes.** Give a recursive definition (similar to the definition of L_{odd} above) of a language L_{bad} that does not contain either three $\mathtt{0}$'s or three $\mathtt{1}$'s in a row. E.g., $\mathtt{001101} \in L_{\text{bad}}$ but $\mathtt{10001}$ is not in L_{bad}. Explain why your definition is correct but do not give a formal proof.

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. *Hint:*

Solution: We will use the pigeon hole principle. Let the $n+1$ numbers in S be $a_1, a_2, \ldots, a_{n+1}$ and consider $b_1, b_2, \ldots, b_{n+1}$ where $b_i = a_i \mod n$. Note that each b_i belongs to the set $\{0, 1, \ldots, n-1\}$. 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 $a_i \mod n = a_j \mod n$ 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 $\mod n$. 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:

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

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

(a) Give a recursive definition of a palindrome over the alphabet Σ.

(b) Prove $w = w^R$ for every palindrome w (according to your recursive definition).

(c) 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 $(x \cdot y)^R = y^R \bullet x^R$ and $(x^R)^R = x$ for all strings x and y.

Solution:

(a) 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 = ½ for each base case + 1 for the recursive case. No credit for the rest of the problem unless this is correct.

(b) 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

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

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 induction hypothesis
+ 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 an 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 (forexample: even length at least 2, odd length at least 3, and length at most 5.)
+ 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.