Homework 9

> *Last updated:* Tue, 4 Nov 2025 15:18:00 -0500

Out: Mon Nov 03, 12:00pm EDT (noon) **Due**: Mon Nov 10, 12:00pm EDT (noon)

> Note: Assignments are not officially "released" until—and are subject to change without notice up to—the indicated "Out" date and time. If an assignment is posted early, students may look ahead but are responsible for ensuring that they are always working with the most recent version of the homework.

This assignment begins to explore Turing Machines and the languages they recognize.

Homework Problems

1. A Context-Free Language, Again, or Not? (15 points)
2. A Turing-Recognizable Language Example (14 points)
3. Regular Languages Review Question (14 points)
4. Turing Recognizable Languages Closed Under MAKECS? (15 points)
5. README (2 point)

Total: 60 points

Submitting

Submit your solution to this assignment in Gradescope `hw9`. Please assign each page to the correct problem and make sure your solutions are legible.

A submission must also include a `README` containing the required information.

1 A Context-Free Language, Again, or Not?

Here is another language that approximates whitespace checking like what Python has.

$$WSC_3 = \{\texttt{if:}\langle eol\rangle \texttt{_}^*111\langle eol\rangle \texttt{ei:}\langle eol\rangle \texttt{_}^*222\langle eol\rangle \texttt{else:}\langle eol\rangle \texttt{_}^*333 \mid \text{spaces must match}\}$$

where $\langle eol\rangle\}$ represents a newline, `_` is a space, and $\Sigma = \{\texttt{if:}, \texttt{ei:}, \texttt{else:}, \texttt{_}, \langle eol\rangle, 1, 2, 3\}$ (in real-world grammars, terminal (i.e., alphabet) "symbols" can be, and often are, whole words)

Your task is to prove that WSC_3 is **not** a context-free language (CFL).

Specifically,

1. Come up with examples, both in the language and not in the language, to make sure you understand the question (come up with enough examples until you understand the language).

2. Show that the WSC_3 is **not** a context-free language (CFL).

   Use the Pumping Lemma for CFLs and proof by contradiction. Make sure your proof has all the correct components. It should also be presented as a Statements and Justifications Table that starts with assuming the negation of the Statement to Prove.

   Then, the proof should proceed according to a standard proof, where each new Statement logically follows from the previous ones, EXCEPT the proof should eventually have two statements that contradict each other. This contradiction allows you to conclude that the original statement to prove is true.

   The key to the contradiction is finding a counterexample that is in the language, but does not satisfy the conditions of the pumping lemma. Proving this should be a separate analysis that is referenced from the main proof table. This part of the solution should analyze all possible splittings of the counterexample into *five* pieces $uvxyz$, as described by the Pumping Lemma for CFLs, and for each split check whether the second and fourth piece (the u and y in $uvxyz$ split) are "pumpable" together such that the resulting string is still in the language.

Interesting sidenote: This problem is feels similar to the A Context-Free Language? problem from Homework 7. But there, the language was a CFL. Here, by making the language slightly more complicated, we now have a non-CFL language that cannot be recognized by PDAs or decribed by CFGs.

2 A Turing-Recognizable Language Example

The A Context-Free Language, Again, or Not? Problem above asked you to prove that WSC_3 is not a CFL. Now prove that this language **is** a Turing-recognizable language.

Your proof must of course be in the form of a Statements and Justifications table.

Any Turing Machine you create should be a high-level description that follows the required format.

In addition, the criteria to show that a Turing machine recognizes some language is to give **three** example strings that are both accepted by the machine and in the language, and **three** examples that are both not accepted by the machine and not in the language, in an Examples Table.

3 Regular Languages Review Question

NOTE: We are now more than halfway through the semester. As everyone hopefully knows by now, this course has no exams, but we may occasionally give review questions like this one to give students extra feedback on where they might stand with regard to their understanding of

basic course topics. Students who find solving this problem difficult or confusing may wish to seek extra help.

Prove that the regular languages are closed under the following function on languages:

$$\text{COP}(L) = \{\text{strings } str \mid str \text{ isn't in the input}\}$$

As usual, a solution should contain the following parts:

1. An explicitly stated IF-THEN Statement to Prove (please ask if there is any confusion here, if this part is not correct, then the rest of the problem will also not be correct)
2. A proof of the exact statement above in the usual Statements / Justifications Table. Any solutions that prove a different theorem or answer a different question will not receive credit.
3. If necessary, any machine or other transformation functions should be separate from, but referenced from, the main proof table. In particular, this proof **must use a regular expression** in one of a steps, and **must define a DFA-transformation function called** $\text{TRANSFORM}_{\text{DFA}}$ and use it in the proof
4. If the proof has a statement of the form "SOME-MACHINE recognizes SOME-LANGUAGE" (which it should), this must be justified with an Examples Table, separate from the main proof table, that demonstrates the is machine correct with respect to the language. The table should include both examples in the language and not in the language, should cover all paths through the machine, and should think about corner cases. Also, if the proof does not involve one specific machine or language, then the Examples Table needs Justifications.

Further, since this is a review problem, **no external sources** other than the class notes and textbook may be used to solve this problem. This means that use of any undefined notation or terminology will result in no credit for this problem.

---

4 Turing Recognizable Languages Closed Under MAKECS?

The MAKECS function has previously appeared in the Are Regular Languages Closed Under MAKECS? Problem from Homework 3 and the Second Chance! Regular Langs Closed Under MAKECS? Problem from Homework 6 so you should be familiar with it and know how to do proofs involving it.

For this problem, prove that the Turing-recognizable languages are closed under MAKECS.

- The proof should begin by stating the IF-THEN Statement that is being proved. Any modification to the required Statement to Prove will result in an invalid proof.
- Then, the proof itself should be in a properly formatted Statements / Justifications Table (as described in Chapter 1.2 of the HMU textbook), where the Statement to Prove from above should be the last Statement that is proved.
- Each line of the proof must be properly justified and logically follow from previous steps. Any extraneous statements or comments will make the proof invalid.

- Be careful to only use acceptable Justifications. Any extraneous sentences or comments or external theorems not covered will make the proof invalid.

- In particular, the only acceptable Justification for a statement like "TM M recognizes language L" is a properly formatted Examples Table with correctly labeled columns and proper explanations / Justifications for each example if necessary.