University of Nevada Las Vegas Computer Science 456/656 Fall 2025
Assignment 2: Due Saturday September 13, 2025, 11:59:59 PM (midnight)

As usual, I hope that students will identify any problems. Thanks to the student who identified one of the errors in this homework!

Name:____

You are permitted to work in groups, get help from others, read books, and use the internet. You will receive a message from the graduate assistant, Sabrina Wallace at `sabrina.wallace@unlv.edu` telling you how to turn in assignments.

1. Identify which machine accepts the language defined by each regular expression.

   (i) $a^* + b^*$ M_7

   (ii) λ M_2

   (iii) a^* M_3

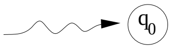
   (iv) $\emptyset$ M_1

   (v) $a(aa+b)^*$ M_8

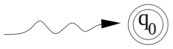
   (vi) a^*b^* M_6

   (vii) $(a+b)^*$ M_4

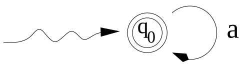
   (viii) $(ab)^*$ M_5



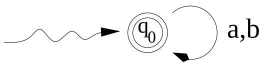
M_1



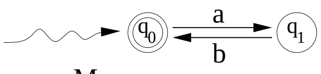
M_2



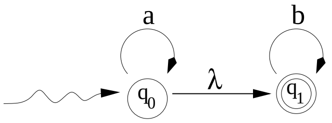
M_3



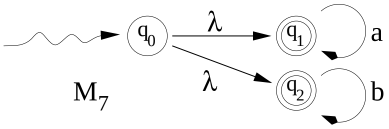
M_4



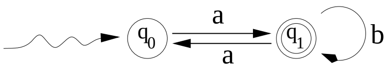
M_5



M_6



M_7



M_8

2. True or False.

   (i) **T** If L is any language, $L + L = L$

   (ii) **T** If L is any language, $L \cap L = L$

   (iii) **T** If L is any language, $\{\lambda\} \in L^*$.

3. Let $L_1 = \{a, ab\}$ and $L_2 = \{a, ba\}$. How many strings are there in the language L_1L_2? **3** How many strings are there in the language L_2L_1? **4**

4. True/False. If the answer is not known to science at this time, enter "O" for Open.

(i) **T** co-$\mathcal{P} = \mathcal{P}$.

(ii) **O** co-$\mathcal{NP} = \mathcal{NP}$.

(iii) **T** co-$\mathcal{P}$–SPACE = $\mathcal{P}$–SPACE.

(iv) **T** Block placement problems are $\mathcal{NP}$.

(v) **T** Sliding block problems are $\mathcal{P}$–SPACE.

(vi) **O** $\mathcal{P}$–SPACE = $\mathcal{NP}$

(vii) **O** Regular expression equivalence is $\mathcal{P}$.

(viii) **T** Regular expression equivalence is decidable.

(ix) **F** Context-free grammar equivalence is decidable.

(x) **T** Every regular language is context-free.

(xi) **F** The language C++ is context-free.

(xii) **F** The intersection of any two context-free languages is context-free.

(xiii) **F** The complement of any context-free language is context-free.

(xiv) **T** Every language is countable.

(xv) **F** For any real number x, there is a program that prints the decimal expansion of x.

(xvi) **T** There are only countably many decidable binary languages.

(xvii) **T** Given a regular grammar G with n variables, there exists an NFA with n states that accepts $L(G)$.

(xviii) **O** Given an integer n written in binary notation, it is possible to find the prime factors of n in polynomial time.

(xix) **T** Given an integer n written in binary notation, it is possible to decide whether n is prime in polynomial time. (This was proved in 2002.)

(xx) **F** Any language generated by a grammar is decidable.

That's true for regular, context-free, and context-sensitive grammars. But it's possible for a general grammar to generate an undecidable langague.

(xxi) **T** The complement of any decidable language is decidable.

(xxii) **T** The union of any two decidable languages is decidable.

(xxiii) **T** The complement of any undecidable language is undecidable.

(xxiv) **F** The union of any two undecidable languages is undecidable.

(xxv) **F** Every context-free language is accepted by some DPDA.

(xxvi) **T** Any language consisting of all decimal numerals of an arithmetic sequence (for example: $L = \{\langle 5+8n\rangle : n \geq 0\} = \{5, 13, 21, 29, 37, 45\ldots\}$) is regular. Note: the members of L are numerals, not numbers.

(xxvii) **T** Let L_1 be a regular binary language. Let L_2 be the language of all strings obtained from members of L_1 by substituting ab for 0 and c for 1. Then L_2 must be regular. For example, if $L_1 = \{0, 10, 10011\}$ then $L_2 = \{ab, cab, cababcc\}$.

(xxviii) **T** DFA equivalence is $\mathcal{P}$–TIME.

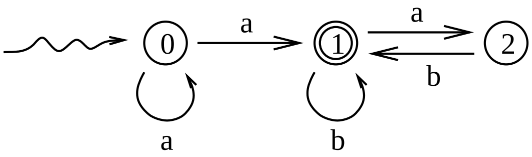
(xxix) **O** NFA equivalence is $\mathcal{P}$–TIME.

(xxx) **O** NFA equivalence is $\mathcal{NP}$–TIME.

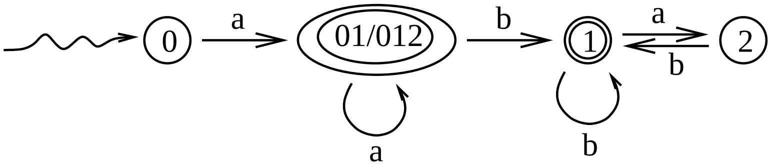
(xxxi) **O** Regular expression equivalence is $\mathcal{NP}$–TIME.

(xxxii) **T** Regular expression equivalence is $\mathcal{P}$–SPACE.

5. Any NFA with n states is equivalent to some DFA with at most 2^n states, counting the dead state. Draw a minimal DFA equivalent to the following three state NFA.



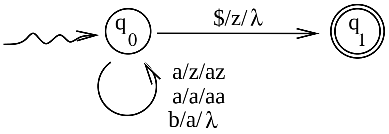
	a	b
0	01	$\emptyset$
1	2	2
2	$\emptyset$	1
01	012	1
012	012	1



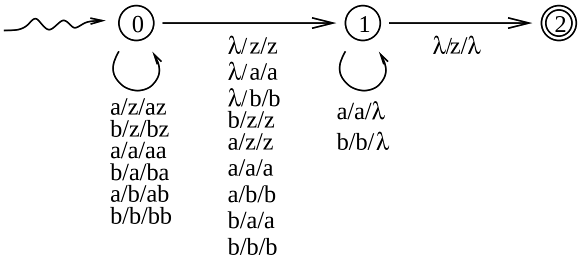
There are 8 subsets of $\{0, 1, 2\}$. The states 02 and 12 are unreachable. The states 01 and 012 are equivalent. The state $\emptyset$ is a dead state, and I did not draw it. The minimal DFA, without the dead state, has 4 states of which 2 are final.

6. Match each PDA with the correct language description in problem 7

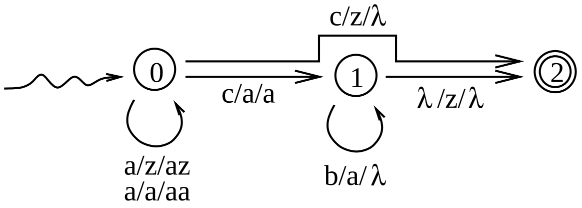
(i) (d)



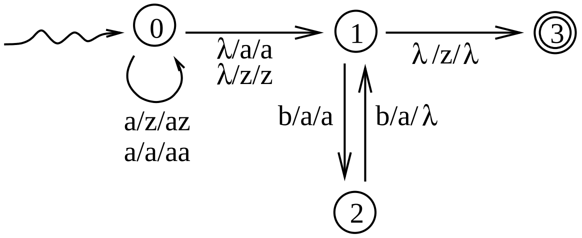
(ii) (f)



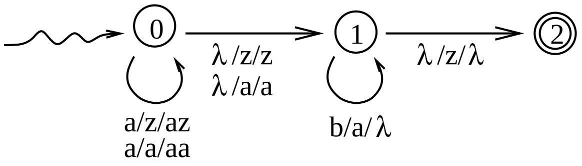
(iii) (e)



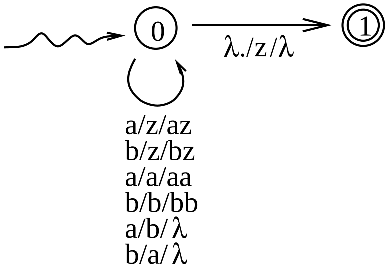
(iv) The language accepted by this machine is $\{a^n b^{2n}\}$, but that description is not listed in problem 7. My error.



(v) (a)



(vi) (b)



This machine was not in the original homework, but I'm putting it in now so that you can see it.

7. Here are the language descriptions for the machines in problem 6.

   (a) $\{a^n b^n \,:\, n \geq 0\}$

   (b) $\{w \in \{a,b\}^* \,:\, \#_a(w) = \#_b(w)\}$, that is, all strings over $\{a,b\}$ with equal numbers of each symbol,

   (c) $w \in \{a,b\}^* \,:\, 2\#_a(w) = \#_b(w)$.

   Oops! This was supposed to be the language of machine (iv). There is a DPDA which accepts it, but the problem of designing that DPDA is too complex for homework2.

(d) Generated by the CF grammar:

1. $S \to aSbS$

2 $S \to \lambda$

This is the Dyck language, writing a for (and b for).

(e) Generated by the CF grammar:

1. $S \to aSb$

2. $S \to c$

$a^n cb^n$

(f) Palindromes over $\{a, b\}$. Note that the matching PDA is not a DPDA. In fact, there are no DPDA which accept languages of palindromes. (I expect you to know this fact.)