University of Nevada, Las Vegas Computer Science 456/656 Spring 2026

Assignment 6: Due Saturday May 2, 2026, 11:59:59 PM

Name:____

You are permitted to work in groups, get help from others, read books, and use the internet. Turn in the assignment as instructed by the Graduate Assistant, Shubhashish Kar, `shubhashish.kar@unlv.nevada.edu`

1. True, False, or Open.

   (i) **T** Every context-sensitive language is decidable.

   (ii) **F** The set of unary numerals for powers of 2 is context-free.

   (iii) **F** The Rice number of a language L is a recursive real number if and only if L is recursively enumerable.

   (iv) **T** $L = \{a^n b^n c^n d^n \, : \, n > 0\}$ is context-sensitive.

   (v) **F** Every context-free grammar can be parsed by some LALR parser.

   (vi) **T** Tropical matrix multiplication, which can be used for minpath problems, is $\mathcal{NC}$.

2. Consider the following annotated CF grammar G, where E is the start symbol, and the action and goto tables of an LALR parser for G.

   1. $E \to E +_2 E_3$
   2. $E \to E *_4 E_5$
   3. $E \to (_6 E_7)_8$
   4. $E \to x_9$

	x	$+$	$*$	(	)	\$	
0	$s1$			$s6$			1
1		$s2$	$s4$	$s6$		HALT	
2	$s9$			$s6$			3
3		$r1$	$s4$		$r1$	$r1$	
4	$s9$			$s6$			5
5		$r2$	$r2$		$s8$	$r2$	
6	$s9$			$s6$			7
7		$s2$	$s4$		$s8$		
8		$r3$	$r3$		$r3$	$r3$	
9		$r4$	$r4$		$r4$	$r4$	

   (a) There are two entries in the action table which together guarantee that multiplication has precedence over addition. There is one entry which guarantees that addition is left associative, and one which guarantees that multiplication is left associative. Identify each of those four entries, by row and column.

   Entry $3, +$ ensures that addition is left-associative, since it calls for reduction, not shift.

   Entry $5, *$ ensures that multiplication is left-associative, since it calls for reduction, not shift.

   Precedence of multiplication over addition is actually two rules. If the code is $x + x * x$ or $x * x + x$, multiplication is executed first. But note that the operators have exchanged position. In the first case, $*$ is later than $+$, in the second case it is earlier. This is in fact two cases, for each of which precedence is resolved by the action table. In the first case, the action is s4 (instead of r1) at entry $(3, *)$. In the second case, case, the action is r2 instead of s2 at entry $(3, *)$

   (b) Walk through a computation of the parser with the input string $x * x + x * ((x + x * x) + x)$.

   I have left the next page blank for this computation.

   .

stack	input	output	action
$\$_0$	$x * x + x * ((x + x * x) + x)\$$		
$\$_0 x_9$	$*x + x * ((x + x * x) + x)\$$		$s9$
$\$_0 E_1$	$*x + x * ((x + x * x) + x)\$$	4	$r4$
$\$_0 E_1 *_4$	$x + x * ((x + x * x) + x)\$$	4	$s4$
$\$_0 E_1 *_4 x_9$	$+x * ((x + x * x) + x)\$$	4	$s9$
$\$_0 E_1 *_4 E_5$	$+x * ((x + x * x) + x)\$$	44	$r4$
$\$_0 E_1$	$+x * ((x + x * x) + x)\$$	442	$r2$
$\$_0 E_1 +_2$	$x * ((x + x * x) + x)\$$	442	$s2$
$\$_0 E_1 +_2 x_9$	$*((x + x * x) + x)\$$	4424	$s9$
$\$_0 E_1 +_2 E_3$	$*((x + x * x) + x)\$$	4424	$r4$
$\$_0 E_1 +_2 E_3 *_4$	$((x + x * x) + x)\$$	4424	$s4$
$\$_0 E_1 +_2 E_3 *_4 (_6$	$(x + x * x) + x)\$$	4424	$s6$
$\$_0 E_1 +_2 E_3 *_4 (_6 (_6$	$x + x * x) + x)\$$	4424	$s6$
$\$_0 E_1 +_2 E_3 *_4 (_6 (_6 x_9$	$+x * x) + x)\$$	4424	$s9$
$\$_0 E_1 +_2 E_3 *_4 (_6 (_6 E_7$	$+x * x) + x)\$$	44244	$r4$
$\$_0 E_1 +_2 E_3 *_4 (_6 (_6 E_7 +_2$	$x * x) + x)\$$	44244	$s2$
$\$_0 E_1 +_2 E_3 *_4 (_6 (_6 E_7 +_2 x_9$	$*x) + x)\$$	44244	$s9$
$\$_0 E_1 +_2 E_3 *_4 (_6 (_6 E_7 +_2 E_3$	$*x) + x)\$$	442444	$r4$
$\$_0 E_1 +_2 E_3 *_4 (_6 (_6 E_7 +_2 E_3 *_4$	$x) + x)\$$	442444	$s4$
$\$_0 E_1 +_2 E_3 *_4 (_6 (_6 E_7 +_2 E_3 *_4 x_9$	$) + x)\$$	442444	$s9$
$\$_0 E_1 +_2 E_3 *_4 (_6 (_6 E_7 +_2 E_3 *_4 E_4$	$) + x)\$$	4424444	$r4$
$\$_0 E_1 +_2 E_3 *_4 (_6 (_6 E_7 +_2 E_3$	$) + x)\$$	44244442	$r2$
$\$_0 E_1 +_2 E_3 *_4 (_6 (_6 E_7$	$) + x)\$$	442444421	$r1$
$\$_0 E_1 +_2 E_3 *_4 (_6 (_6 E_7)_8$	$+x)\$$	442444421	$s8$
$\$_0 E_1 +_2 E_3 *_4 (_6 E_7$	$+x)\$$	4424444213	$r3$
$\$_0 E_1 +_2 E_3 *_4 (_6 E_7 +_2$	$x)\$$	4424444213	$s2$
$\$_0 E_1 +_2 E_3 *_4 (_6 E_7 +_2 x_9$	$)\$$	4424444213	$s9$
$\$_0 E_1 +_2 E_3 *_4 (_6 E_7 +_2 E_3$	$)\$$	44244442134	$r4$
$\$_0 E_1 +_2 E_3 *_4 (_6 E_7$	$)\$$	442444421314	$r1$
$\$_0 E_1 +_2 E_3 *_4 (_6 E_7)_8$	$\$$	442444421314	$s8$
$\$_0 E_1 +_2 E_3 *_4 E_5$	$\$$	4424444213143	$r3$
$\$_0 E_1 +_2 E_3$	$\$$	44244442131432	$r2$
$\$_0 E_1$	$\$$	442444421314321	$r1$

HALT

3. Every non-empty context-free language is generated by a CNF (Chomsky normal form) grammar.

   Let L be the set of all non-empty strings generated by the grammar given in Problem 2. Construct a CNF grammar for L. You will need to introduce a number of variables.

   1. $E \to x$
   2. $L \to ($
   3. $R \to)$
   4. $P \to +$
   5. $T \to *$
   6. $A \to EP$
   7. $A \to ET$
   8. $E \to AE$
   9. $B \to LE$
   10. $E \to BR$

4. Prove the pumping lemma for regular languages. Let L be a regular language. Pick a DFA which accepts L, and let p be the number of states of M. Let $w \in L$ have length $n \geq p$. Let $q_0, q_1, \ldots q_n$ be the sequence of states of M with input w. Then q_0 is the start state of M and q_n is a final state of M. Since M has only p states, there must be integers j and p, for $0 \leq j <\leq p$ such that $q_j = q_p$.

   We need to pick strings x, y, and z which satisfy conditions 1. through 4. of the pumping lemma. We let x be the prefix of w of length j, y the string such that xy is the prefix of w of length k, and z the string such that $xyz = w$. Condition 1. holds since $w = xyz$. Condition 2. holds since $|xy| = k \leq p$, and Condition 3. holds since $|y| = k - j > 0$. To prove Condition 4., let $i \geq 0$. Note that $\delta(q_0, x) = q_j$ and $\delta(q_j, y) = q_k = {}_j$; thus the sequence of steps in that computation is a loop and i-fold concatenating of that loop with itself is also a loop, and takes the computation from ${}_k$ to ${}_j = {}_k$. Finally, $\delta(q_k, z) = q_n$, which is a final state. The computation of M with input xy^iz is from q_0 to q_j, then a loop from q_j back to itself, namely q_k, then to ${}_n$ which is final. Thus $xy^iz \in L$, which satisfies Condition 4. Since L was an arbitrary regular language, we have proved the pumping lemma.

5. Use the pumping lemma for context-free languages to prove that $L = \{a^nb^nc^n : n \geq 0\}$ is not context-free.

   Proof by contradiction. Assume that w is context-free. Let p be the pumping length given by the pumping lemma for context-free languages, and let $w = a^pb^pc^p$, which is a member of L. By the pumping lemma, there exist strings u, v, x, y, z which satisfy the four conditions of the lemma. By Condition 1., $w = uvxyz$, and by Condition 2., vxy is a substring of w of length no greater than p. But vxy cannot contain both the symbol a and the symbol c, because it would then have to contain all the symbols between that a and that c, which would include b^p, making its length at least $p + 2$, contradiction; thus, vxy either has no a nor no c. By condition 4., choosing $i = 0$, $uxz \in L$. Thus, for some m, $uxz = a^mb^mc^m$. By Condition 3., $|v| + |y| = k$ for $k > 0$. Thus, $|uxz| = m \leq 3p - k$. It follows that uxz has $m < n$ a's and $m < n$ c's, contradiction. We conclude that L is not context-free.

6. Give a proof, by induction, that $\sum_{i=1}^{n} i^3 = \dfrac{n^2(n+1)^2}{4}$

   The formula is true for $n = 1$, since both sides of the equation equal 1. The inductive hypothesis is that the formula is true for $n - 1$. We must then prove the formula for n.

$$\begin{aligned}
\sum_{i=1}^{n} i^3 &= \sum_{i=1}^{n-1} i^3 + n^3 \\
&= \frac{(n-1)^2 n^2}{4} + n^3 \text{ by the inductive hypothesis} \\
&= \frac{n^4 - 2n^3 + n^2}{4} + n^3 \\
&= \frac{(n^4 + 2n^3 + n^2}{4} \\
&= \frac{n^2(n+1)^2}{4} \text{ and we are done.}
\end{aligned}$$

7. Give a context-sensitive grammar for $\{a^n b^n c^n d^n \ : \ n > 0\}$ 1. $S \to abc$
   2. $S \to aaAbc$
   3. $A \to AaA$
   4. $Aa \to aA$
   5. $Ab \to bbB$
   6. $Bb \to bB$
   7. $Bc \to cc$
   8. $Cd \to dd$

8. Give a proof, by contradiction, that $\sqrt{2}$ is irrational.

   Assume that $\sqrt{2} = \frac{p}{q}$ where p and q are integers. Without loss of generality, the fraction is reduced to the lowest terms, that is, p and q have no common divisor greater than 1.

$$\begin{aligned}
\sqrt{2} &= \frac{p}{q} \\
2 &= \frac{p^2}{q^2} \\
2q^2 &= p^2
\end{aligned}$$

   Thus p^2 is even, hence p is even. Then $p = 2k$ where k is an integer.

$$\begin{aligned}
2q^2 &= (2k)^2 \\
2q^2 &= 4k^2 \\
q^2 &= 2k^2
\end{aligned}$$

   Thus q^2 is even, hence q is even. contradicting that the original fraction is reduced to the lowest terms. We conclude that $\sqrt{2}$ is not rational.

9. Which of these problems or languages are known to be $\mathcal{NP}$–complete? Mark each language either T if it is known to be $\mathcal{NP}$–complete, F otherwise.

   (i) **T** Boolean satisfiability

(ii) **T** 3-SAT

(iii) **F** 2-SAT

(iv) **T** TSP

(v) **F** DFA Equivalence

(vi) **F** NFA Equivalence

(vii) **F** CFG Equivalence

(viii) **F** Regular Grammar Equivalence

(ix) **F** Program Equivalence

(x) **F** CVP

(xi) **T** Knapsack problem

(xii) **F** Regular Expression Equivalence

(xiii) **F** Graph Coloring with 2 colors

(xiv) **T** Graph Coloring with 3 colors

(xv) **T** Block Sorting

(xvi) **T** Firehouse problem

(xvii) **T** Hamiltonian Circuit

(xviii) **F** Graph Isomorphism

(xix) **F** Minimum Spanning Tree

(xx) **T** Dominating Set

(xxi) **T** Independent Set

(xxii) **F** Generalized Checkers (Arbitrary size board)

(xxiii) **F** Furniture Mover's Problem

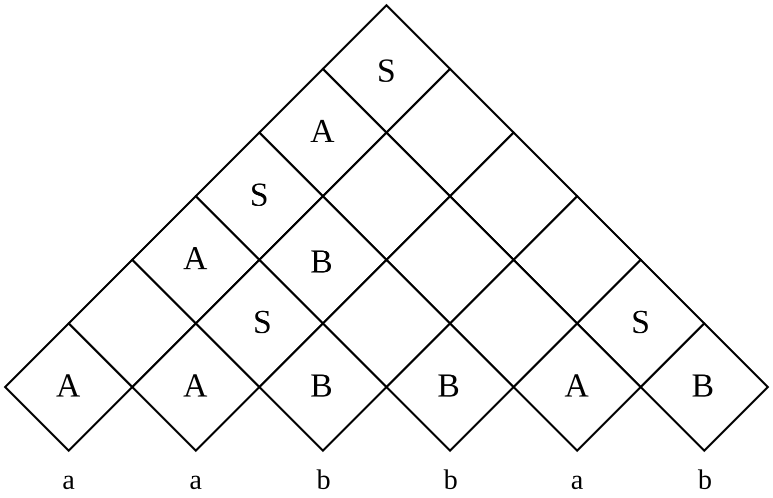
(xxiv) **T** Jigsaw Problem

(xxv) **F** Rush Hour, arbitrary size parking lot

(xxvi) **T** Canadian Traveler's Problem

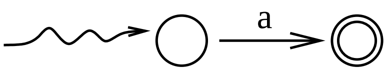
10. Let G be the following CNF grammar, which generates the Dyck language. Use the CYK algorithm to decide whether $aabbab \in L(G)$.

1. $S \to AB$
2. $A \to a$
3. $B \to b$
4. $A \to AS$
5. $B \to BS$

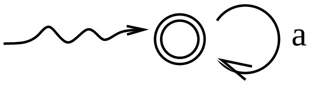


11. For each regular epression, draw a DFA with at most 2 states which accepts the language described by that expression.

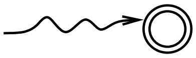
(a) a



(b) a^*

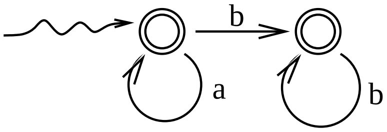


(c) λ

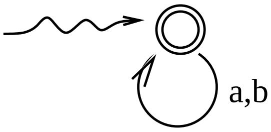


(d) $\emptyset$

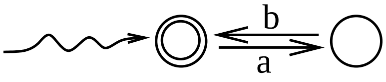
(e) a^*b^*



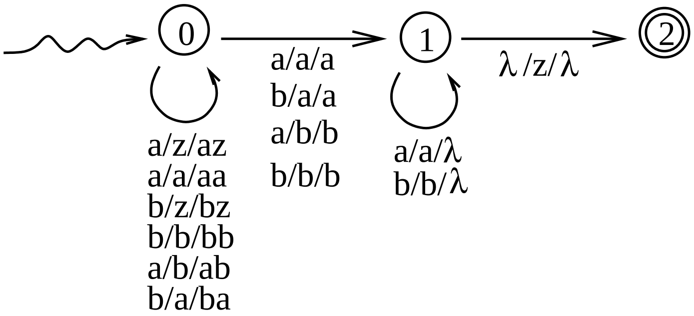
(f) $(a+b)^*$



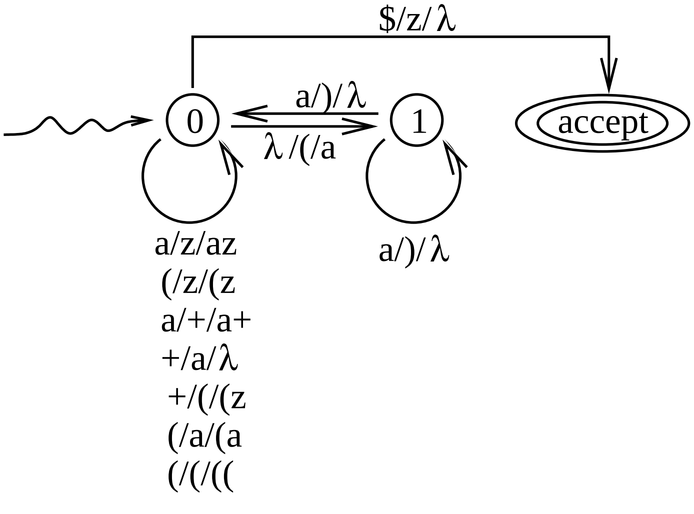
(g) $(ab)^*$



12. Construct a PDA which accepts the language of all palindromes over $\{a, b\}$.

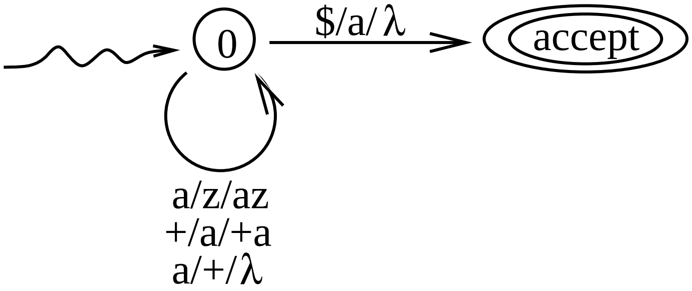


13. Construct a DPDA which accepts the language generated by this grammar. The start symbol is E.



1. $E \to a$
2. $E \to E + E$
3. $E \to (E)$

This problem is harder than I realized. Substitute the language generated by the grammar below. You are still welcome to work the original version.



1. $E \to a$
2. $E \to E + E$

14. Recall that the Rice number of a language L over an alphabet Σ is defined to be $\sum\limits_{w_i \in L} 2^{-i}$, where $w_1, w_2, \ldots$ is the canonical enumeration of Σ^*. Let $\Sigma = \{1\}$, the *unary* alphabet. Let L be the language consisting of all strings of odd length over that alphabet. Compute the Rice number of L. (Hint: it's a simple fraction.)

$\Sigma^* = \left\{1^i \,:\, i \geq 0\right\} = \{\lambda, 1, 11, 111, 1111, 11111, \ldots\}$ In the canonical enumeration, $w_i = 1^i$. Thus $L = \left\{1^i \,:\, i \text{ is odd}\right\} = \{1, 111, 11111, \ldots\}$. The Rice number of L is thus $\sum\limits_{j=0}^{\infty} 2^{-(1+2j)} = \frac{1}{2} + \frac{1}{8} + \frac{1}{32} + \cdots$ a geometric series whose sum is $\frac{2}{3}$.

Here's an interesting fact for you: the Rice number of any regular language over the unary alphabet is rational. In fact, the Rice number of a language over the unary alphabet is rational if and only if the language is regular. I wonder whether that holds for any alphabet!