CSC 456/656 Spring 2022 Final Examination 8:00-10:00 May 11, 2022

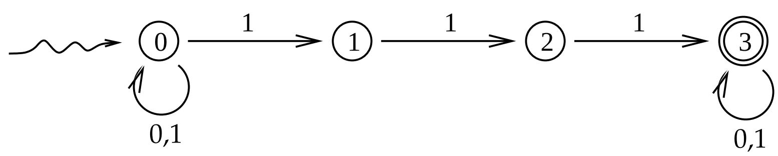
Study Guide, Version II

1. True/False/Open.

2. Alphabets, strings, languages.

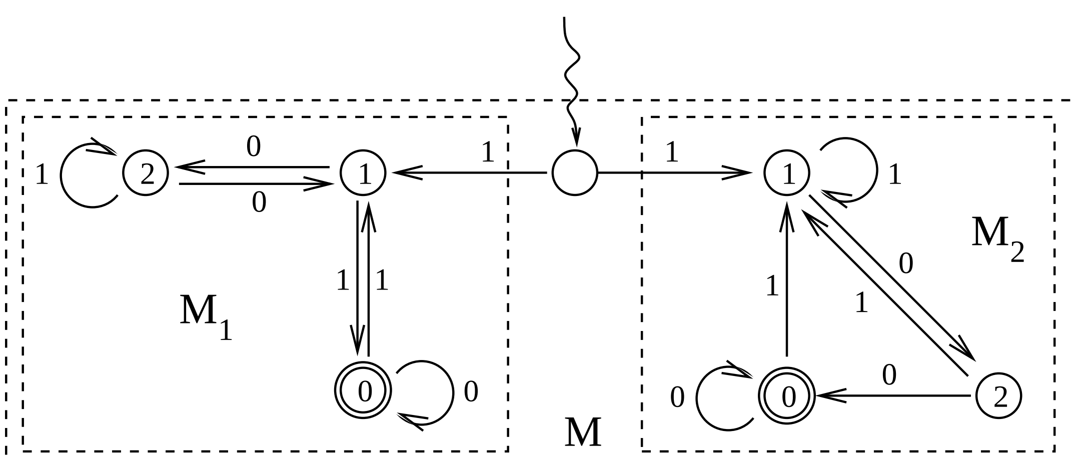
3. Give an NFA for a regular language.

   (i) Find an NFA with at most 4 states which accepts the language of binary strings which contain the substring 111.



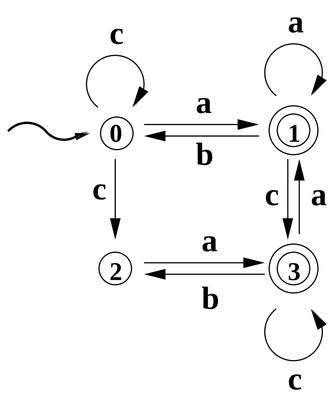
   (ii) Let L be the language of all binary numerals for positive multiples of either 3 or 4, where leading zeros are not permitted. That is, $L = \{11, 100, 110, 1000, 1001, 1100, \ldots\}$. Find an NFA with 8 states which accepts L.

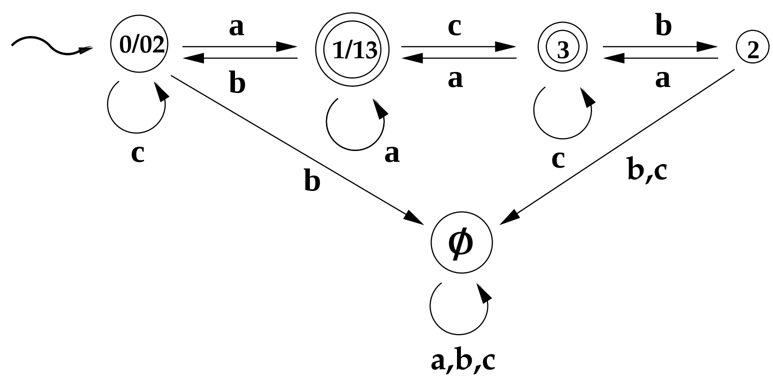
   Let M_1 be the minimal DFA which accepts all positive multiples of 3 and let M_2 be the minimal DFA which accepts all positive multiples of 4. Then M_1 has states labeled 0, 1, and 2, corresponding to residue classes modulo 3, as well as a dead state for when the first symbol is 0, while M_2 has states labeled 0, 1, and 2 as well as the dead state. We define an NFA M has follows. The start state of M has two out-arrows, both labeled 1. One of them goes to state 1 of M_1, the other to state 1 of M_2. We can neglect the dead states, since M is an NFA. The first symbol of any accepted input is 1. M must then guess whether the input string will be a numeral for a multiple of 3, in which case it goes to M_1, or a multiple 4, in which case it goes to M_2. If the input string is not a multiple of either 3 or 4, either choice will lead to non-acceptance. The following figure shows how M is constructed from M_1 and M_2

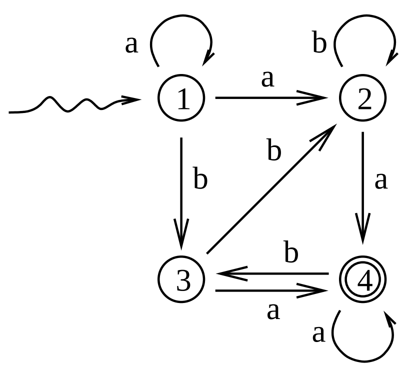


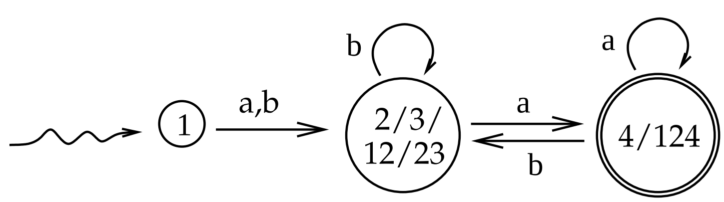
I will not draw the minimal DFA which accepts L, since I won't be asking any question that complicated.

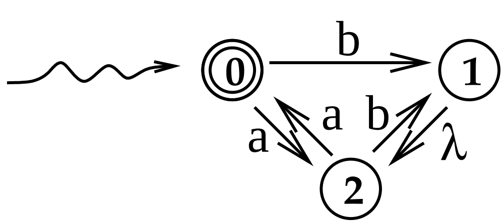
4. Convert these NFAs to DFAs:

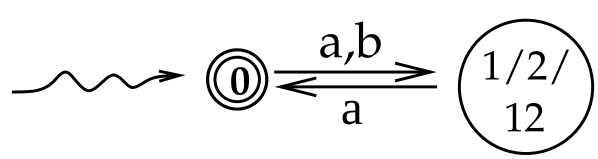




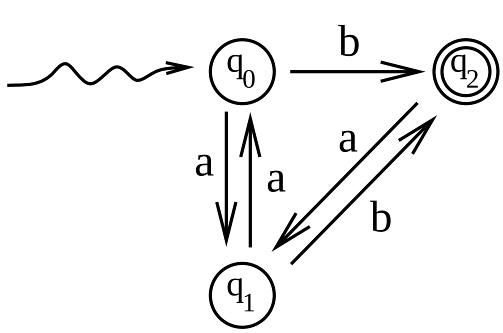


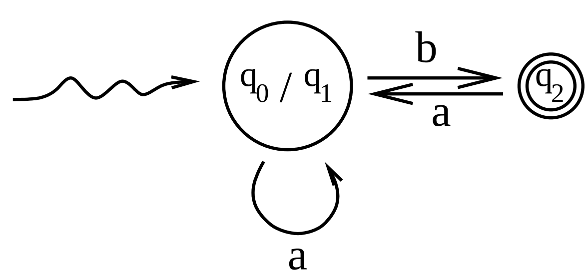


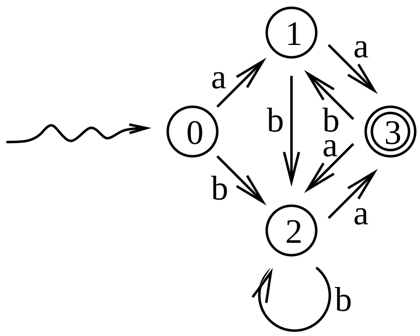


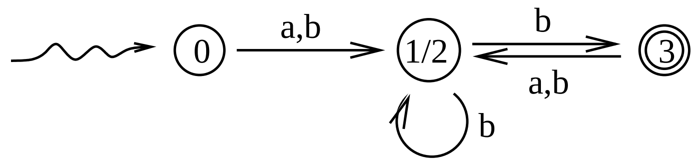


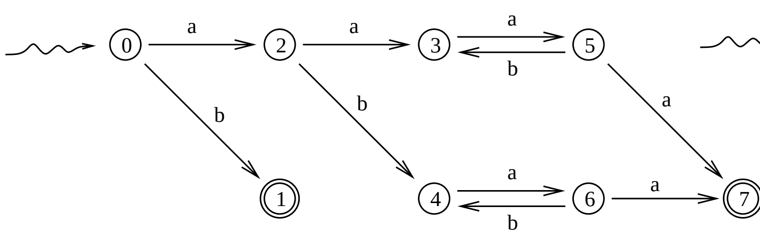
5. Minimize these DFAs:

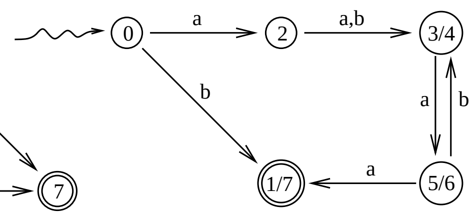






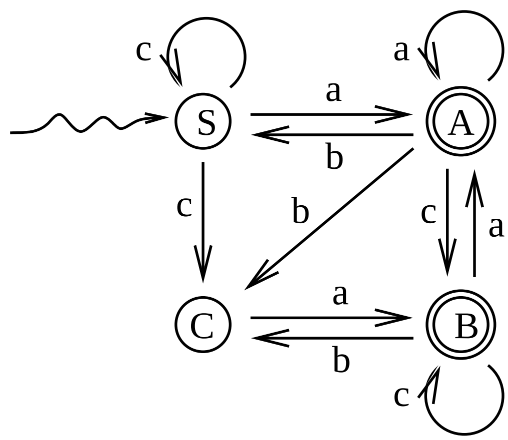




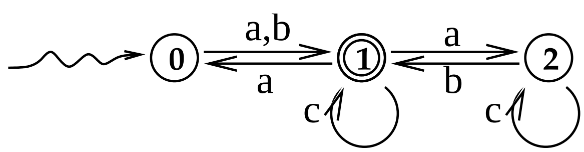


6. Find an NFA which accepts the language generated by this regular grammar.

$S \to aA|cS|cC$
$A \to aA|bS|cB|\lambda$
$B \to aA|cB|bC|\lambda$
$C \to aB$

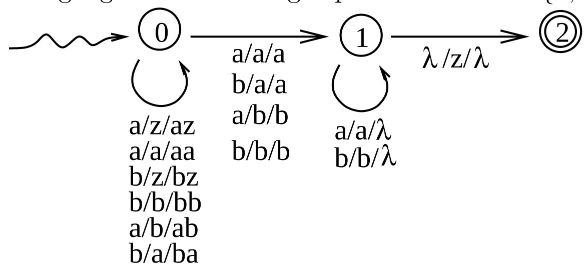


7. Find a regular expression which describes the language accepted by an NFA.

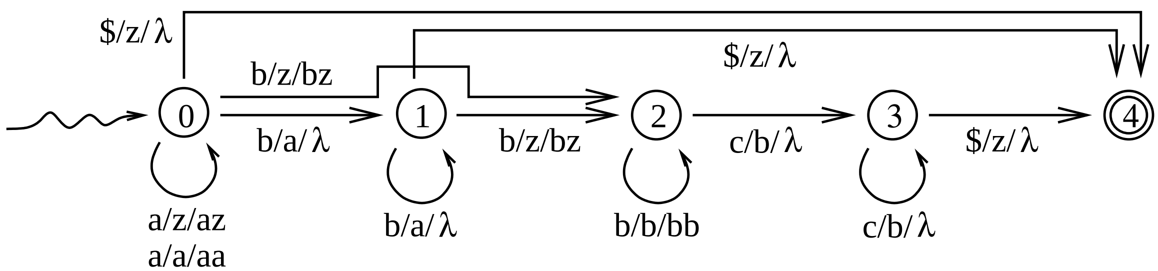


$$(a+b)(a(a+b)+c+ac^*b)^*$$

8. (i) Find a PDA for the language of all odd length palindromes over $\{a, b\}$.



(ii) Find a DPDA for the language $\{a^i b^j c^k \,:\, j = i + k\}$



9. Give a context-free grammar for a context-free language. Let L be the language of all algebraic expressions where The operations are addition, subtraction and multiplication, there are parentheses, and the only variables are x, y, and z. Give an unambiguous CF grammar for L, where addition and subtraction have equal precedence, multiplication has precedence over addition and subtraction, and all three operations are left associative.

The answer is the grammar given in Problem 10 below.

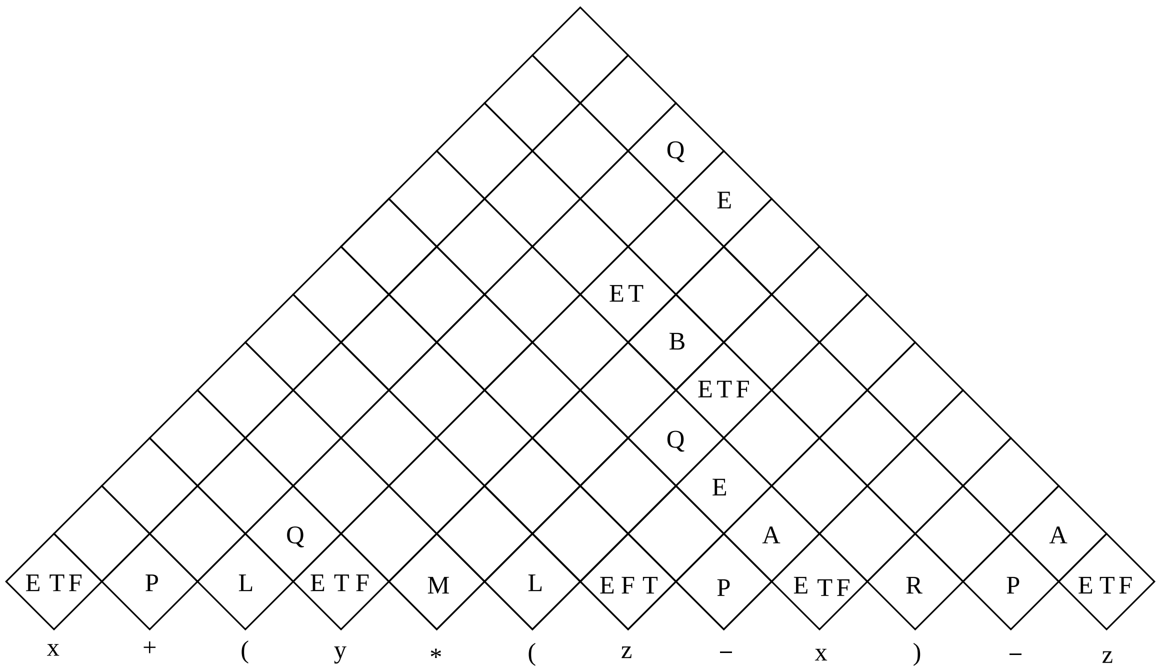
10. Use the CYK algorithm to prove that $x + (y * (z - x) - z$ is not generated by this CF grammar. Hint: You must first find an equivalent CNF grammar.

$E \to E + T \,|\, E - T \,|\, T$
$T \to T * F \,|\, F$
$F \to (E) \,|\, x \,|\, y \,|\, z$

$E \to EA \,|\, TB \,|\, QR \,|\, x \,|\, y \,|\, z$
$A \to TP$
$P \to + \,|\, -$
$T \to TB \,|\, QR \,|\, x \,|\, y \,|\, z$
$B \to MF$
$M \to *$
$F \to QR \,|\, x \,|\, y \,|\, z$
$Q \to LE$
$L \to ($
$R \to)$

The following figure shows that $w \notin L$, since there is no copy of E in the top cell.

(I apologized if I missed something.)



11. Grammar classes: regular, context-free, context-sensitive, unrestricted.

   (i) Find a context-sensitive grammar for $\{a^n b^{2n} c^{3n} \,:\, n \geq 1\}$

   $S \to abbccc \,|\, aAbbccc$
   $Ab \to bA$
   $bAc \to Bbbbcccc$
   $bB \to Bb$
   $aB \to aa \,|\, aaA$

   (ii) Is there an grammar which generates the language of all binary numerals for primes? What grammar class(es) does it belong to?

   Yes, there is an unrestricted grammar for the primes. I do not think there is a context-sensitive grammar, but I am not sure.

12. Let L_1 be the language generated by the CF grammar given in 10, but not the CNF grammar. Give a left-most and a rightmost derivation of the string $x + y * z * (x - y)$.

   Leftmost derivation: $E \Rightarrow E+T \Rightarrow x+T \Rightarrow x+T*F \Rightarrow x+T*F*F \Rightarrow x+F*F*F \Rightarrow x+y*F*F \Rightarrow x+y*z*F \Rightarrow x+y*z*F \Rightarrow x+y*z*(E) \Rightarrow x+y*z*(E-T) \Rightarrow x+y*z*(T-T) \Rightarrow x+y*z*(F-T) \Rightarrow x+y*z*(x-T) \Rightarrow x+y*z*(x-F) \Rightarrow x+y*z*(x-y)$

   Rightmost derivation: $\Rightarrow E+T \Rightarrow E+T*F \Rightarrow E+T*(E) \Rightarrow E+T*(E-T) \Rightarrow E+T*(E-T) \Rightarrow E+T*(E-F) \Rightarrow E+T*(E-y) \Rightarrow E+T*(T-y) \Rightarrow E+T*(F-y) \Rightarrow E+T*(x-y) \Rightarrow E+T*F*(x-y) \Rightarrow E+T*z*(x-y) \Rightarrow E+F*z*(x-y) \Rightarrow E+y*z*(x-y) \Rightarrow E+y*z*(x-y) \Rightarrow T+y*z*(x-y) \Rightarrow F+y*z*(x-y) \Rightarrow x+y*z*(x-y)$

13. Ambiguous and unambiguous CF grammars. Inherently ambiguous CF languages.

Let $L_1 = \{a^i b^i c^j \, : \, i, j \geq 0\}$. Let $L_2 = \{a^i b^j c^j \, : \, i, j \geq 0\}$. Let $L = L_1 + L_2$.

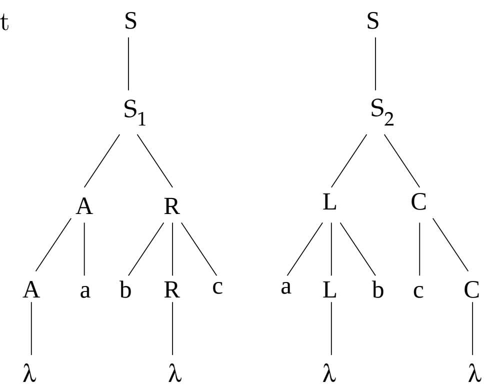
(i) Find a context-free grammar G for L.

$S \to S_1 \,|\, S_2$
$S_1 \to AR$
$A \to Aa \,|\, \lambda$
$R \to bRc \,|\, \lambda$
$S_2 \to LC$
$C \to Cc \,|\, \lambda$
$L \to aLb \,|\, \lambda$

(ii) Is there an unambiguous context-free grammar for L?

No. L is inherently ambiguous.

(iii) Prove that G is ambiguous by giving two different parse trees for some string.



14. Design an LALR parser for the grammar
1. $E \to E -_2 E_3$
2. $E \to E *_4 E_5$
3. $E \to x_6$

	x	$-$	$*$	\$	E
0	$s6$				1
1		$s2$	$s4$	*halt*	
2	$s6$				3
3		$r1$	$s4$	$r1$	
4	$s6$				5
5		$r2$	$r2$	$r2$	
6		$r3$	$r3$	$r3$	

15. Complexity classes: $\mathcal{NC}$, $\mathcal{P}$–complete, $\mathcal{P}$–TIME, $\mathcal{NP}$, co–$\mathcal{NP}$, $\mathcal{NP}$–complete, $\mathcal{P}$–SPACE, recursive, recursively enumerable, co-RE, undecidable.

16. Closure of language classes under various operations.

Which of the above complexity classes are closed under intersection?

$\mathcal{NC}$, $\mathcal{P}$–TIME, $\mathcal{NP}$, co–$\mathcal{NP}$, recursive, recursively enumerable, co-RE.

17. Machine classes and language classes.

For each class of machines, what is the class of languages it accepts?

DFAs and NFAs accept regular languages. PDAs accept context-free languages. TMs accept RE languages.

18. Enumeration, enumeration in canonical order.

(i) A language L is RE if and only if there is a machine that enumerates L.

(ii) A language L is decidable if and only if there is a machine that enumerates L in canonical order.

(iii) What does it mean to say that M enumerates L?

It writes all strings of L.

(iv) What does it mean to say that M enumerates L in canonical order?

It writes all strings of L in canonical order, which means shorter strings before longer, otherwise alphabetical.

19. Both pumping lemmas.

(i) State the pumping lemma for regular languages.

You have that already.

(ii) State the pumping lemma for context-free languages.

You have that already.

20. Reduction definition of $\mathcal{NP}$.

Fill in the blanks: L is $\mathcal{NP}$ if and only if there is a polynomial time reduction of L_2 to L, where L_2 is $\mathcal{NP}$–complete.

21. Certificate/verifier definition of $\mathcal{NP}$.

Fill in the blanks:

Given a language L, if there is a machine V_L such that, for every $w \in L$ there is a string c such that

(i) $|c|$ is a polynomial function of $|w|$

(ii) Given inputs w and c V_L halts in time which is polynomial in $|w|$.

then L is $\mathcal{NP}$.

22. Guide strings.

What is a guide string? (Think about Theseus, Ariadne, and the Minotaur.)

A string of Booleans which tells a non-deterministic machine which choice to make at each step.

23. $\mathcal{NP}$–complete problems.

List as many $\mathcal{NP}$–complete problems, or languages, that we've studied this semester as you can.

Obviously, I can't ask that question, since it's open-ended. I might ask you to list some specific number of them.

24. Finding new $\mathcal{NP}$–complete problems using reduction.

You could prove that L is $\mathcal{NP}$–complete as follows:

(i) Show that L is $\mathcal{NP}$.

(ii) Give a polynomial time reduction of 3-SAT to L.

25. Give a polynomial time reduction of 3-SAT to the independent set problem.

You already have that.

26. Give a polynomial time reduction of the subset sum problem to the partition problem.

    You already have that.

27. $\mathcal{NC}$ problems.

    (i) List a number of well-known polynomial time problem that are now known to be $\mathcal{LC}$.

        i. All the usual operations on binary numerals.
        ii. The membership problem for a regular language.
        iii. The membership problem for a context-free language.
        iv. Matrix multiplication.

    (ii) What is the importance nowadays of $\mathcal{NC}$?

        Say something about parallel machines becoming important, we are just about at the limit of the speed of one processor.

28. $\mathcal{P}$–completeness.

    Name a $\mathcal{P}$–complete problem.

    Dynamic programming is $\mathcal{P}$–complete, unless it has polylogarithmic reachback. The circuit value problem is an example.

29. Problems known or believed to be harder than $\mathcal{NP}$.

    What complexity classes contain the following problems?

    (i) Sliding block problems.

        $\mathcal{P}$–SPACE.

    (ii) Games such Chess, Checkers, and Go: the problem is, given a game configuration, is it true that Black can force a win?

        The complexities of these games are listed in Wikipedia. But all you need to know for the final is that they are at least as hard as $\mathcal{P}$–SPACE.

30. Undecidable problems.

    A language L is undecidable if no machine can decide the membership problem for L.

31. Prove that the halting problem is undecidable.

    You have that.

32. Turing Machines.

    (i) What is a Turing machine?

    (ii) Give a TM that shifts its input one space to the right.

        I have given that.

33. Church-Turing thesis.

    Why is it important?

    You have that.

34. Countable and uncountable sets.

What does it mean for a set S to be countable, also known as enumerable, or denumerable?

Either S is finite, or there exists a 1-1 correspondence between S and the natural numbers.

Which of the following sets are countable?

(a) Integers: Yes (b) Rationals: Yes (c) Real numbers: No (d) The set of all binary strings: Yes (e) The set of all binary languages: No (f) The set of recursive binary languages: Yes (g) The set of RE binary languages: Yes (h) The set of co-RE binary languages: Yes (i) The set of undecidable binary languages: No (i) The set of recursive functions from binary strings to binary strings: Yes (j) The set of all functions from binary strings to binary strings: No

35. Uncomputable functions, also called non-recursive functions.

Describe one non-recursive function. This is hard. I won't ask it.

36. Recursive real numbers. https://www.ams.org/journals/proc/1954-005-05/ S0002-9939-1954-0063328-5/S0002-9939-1954-0063328-5.pdf

What does it mean to say that a real number is recursive? It means that there is a machine which can compute the function D, where $D(n)$ is the n^{th} decimal digit of n. (Or binary, octal, or whatever.)

Is the set of recursive real numbers countable? Yes.

37. Truth vs Reason.

Not all truth can be discovered by reason: prove that there is some mathematical statement that is true but which has no mathematical proof.

There is a machine M and a string w such that, with input w M never halts, and yet there is no proof of that fact. We now such an example exists, since otherwise the halting problem would be decidable.