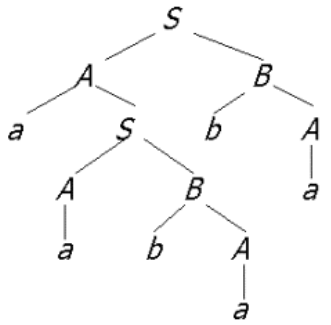
Submission number: 82170
Submission certificate: FG377946
Submission time: 2014-05-01 04:04:41 PST (GMT - 8:00)

Number of questions: 25
Positive points per question: 4.0
Negative points per question: 0.0
Your score: 68

1. The folowing is a parse tree in some unknown grammar G:



Which of the following productions is **definitely not** a production of G?

a) S → CB
b) None of the other choices.
c) S → aC
d) A → a

Answer submitted: **b)**

You have answered the question correctly.

2. There is a Turing transducer *T* that transforms problem P1 into probem P2. *T* has one read-only input tape, on which an input of length *n* is placed. *T* has a read-write scratch tape on which it uses O(S(n)) cells. *T* has a write-only output tape, with a head that moves only right, on which it writes an output of length O(U(n)). With input of length *n*, *T* runs for O(T(n)) time before halting. You may assume that each of the upper bounds on space and time used are as tight as possible.

A given combination of S(n), U(n), and T(n) may:

1. Imply that *T* is a polynomial-time reduction of P1 to P2.
2. Imply that *T* is NOT a polynomial-time reduction of P1 to P2.
3. Be impossible; i.e., there is no Turing machine that has that combination of tight bounds on the space used, output size, and running time.

What are all the constraints on S(n), U(n), and T(n) if *T* is a polynomial-time reducer? What are the constraints on feasibility, even if the reduction is not polynomial-time? After working out these constraints, identify the true statement from the list below.

a) $S(n) = n^2$; $U(n) = n^3$; $T(n) = n^4$ is not physically possible.

b) $S(n) = n$; $U(n) = n^2$; $T(n) = n!$ is possible, but not a polynomial-time reduction.

c) $S(n) = n$; $U(n) = n^2$; $T(n) = n!$ is a polynomial-time reduction

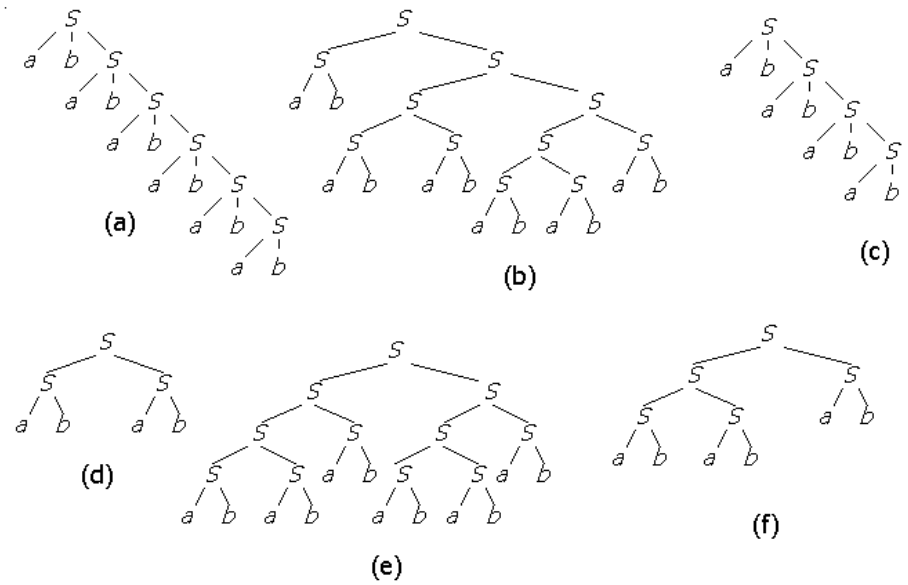
d) $S(n) = n^2$; $U(n) = 1$; $T(n) = n^{10}$ is a polynomial-time reduction.

Answer submitted: **b)**

Your answer is incorrect.

There are two reasons a combination might be impossible. Perhaps the running time is so large that the Turing machine cannot run that long without entering a loop because it repeats an ID (combination of state, head positions on the input and scratch tapes, and scratch-tape contents). Or the running time is so small that the Turing machine cannot possibly use that many cells of the scratch or output tapes. The conditions for a reduction to be polynomial-time are in Section 10.1.5 (p. 433).

3. Consider the grammar G: S → SS, S → ab. Which of the following strings is a word of L(G) AND is the yield of one of the parse trees for grammar G in the figure below?



a) abababab

b) SababSabS

c) ababab

d) ab

Answer submitted: **c)**

You have answered the question correctly.

4. Which of the following grammars derives a subset L_s of the language: L= {x | (i) x contains a and c in proportion 4:3, (ii) x does not begin with c and (iii) there are no two consecutive c's} such that L_s is missing at most a finite number of strings from L.

a) S → ε, S → acacaScSaS

b) S → acacacacacaaca , S → ε, S → SaScSaScSaScSaS

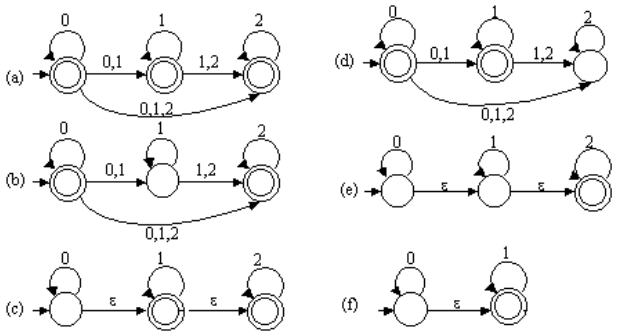
c) S → acacaca , S → SaScSaScSaScSaS, S → SaSaSaScSaScSaS

d) S → ε , S → SaScSaScSa

Answer submitted: **b)**

You have answered the question correctly.

5. Identify which automata define the same language and provide the correct counterexample if they don't. Choose the correct statement from the list below.



a) (a) and (b) do not define the same language and the following counterexample shows it. String 0111 is accepted by one and not by the other.

b) (a) and (e) do not define the same language and the following counterexample shows it. String 0012 is accepted by one and not by the other.

c) (e) and (d) do not define the same language and the following counterexample shows it. String 01 is accepted by one and not by the other.

d) (b) and (c) define the same language.

Answer submitted: **d)**

You have answered the question correctly.

6. The Turing machine M has:

- States q and p; q is the start state.
- Tape symbols 0, 1, and B; 0 and 1 are input symbols, and B is the blank.
- The following next-move function:

State	Tape Symbol	Move
q	0	(q,0,R)
q	1	(p,0,R)
q	B	(q,B,R)
p	0	(q,0,L)
p	1	none (halt)
p	B	(q,0,L)

Simulate M on the input 1010110, and identify one of the ID's (instantaneous descriptions) of M from the list below.

a) 00000q10

b) 00q00110

c) 101p0110

d) 101q0110

Answer submitted: **b)**

You have answered the question correctly.

7. Suppose one transition rule of some PDA P is $\delta(q,0,X) = \{(p,YZ), (r,XY)\}$. If we convert PDA P to an equivalent context-free grammar G in the manner described in Section 6.3.2 (p. 247), which of the following could be a production of G derived from this transition rule? You may assume *s* and *t* are states of P, as well as *p*, *q*, and *r*.

a) [qXr] → [pYs][sZr]

b) [qXr] → 0[rXs][sYr]

c) [qXr] → [rXs][sYr]

d) [qXr] → 0[rXs][qYr]

Answer submitted: **b)**

You have answered the question correctly.

8. Here is the transition table of a DFA:

	0	**1**
→A	E	D
*B	A	C
C	G	B
D	E	A
*E	H	C
F	C	B
G	F	E
H	B	H

Find the minimum-state DFA equivalent to the above. Then, identify in the list below the pair of equivalent states (states that get merged in the minimization process).

a) D and G

b) D and H

c) F and H

d) E and G

Answer submitted: **a)**

Your answer is incorrect.

On input 1, G goes to accepting state E, while D goes to nonaccepting state A. Thus, these states are distinguishable. See Section 4.4.3 (p. 160) for the state-minimization algorithm.

9. Suppose a problem P_1 reduces to a problem P_2. Which of the following statements can we conclude to be TRUE based on the above?

a) If P_1 is decidable, then it must be that P_2 is decidable.

b) If P_1 is undecidable, then it must be that P_2 is undecidable.

c) If P_1 is non-RE, then it must be that P_2 is RE.

d) If P_2 is undecidable, then it must be that P_1 is decidable.

Answer submitted: **a)**

Your answer is incorrect.

Hint: Let $P_1 = \{\ \}$ and P_2 be the language L_{ne} defined in Section 9.3.2. p. 394. You should examine Section 9.3.1 (p. 392) for a discussion of what is implied by the existance of a reduction from one problem to another.

10. Programming languages are often described using an extended form of context-free grammar, where square brackets are used to denote an optional construct. For example, A → B[C]D says that an *A* can be replaced by a *B* and a *D*, with an optional *C* between them. This notation does not allow us to

describe anything but context-free languages, since an extended production can always be replaced by several conventional productions.

Suppose a grammar has the extended productions:

$A \rightarrow 0B[C10D]1EF0 \mid 0BC1[0D1E]F0$

Convert this pair of extended productions to conventional productions. Identify, from the list below, the conventional productions that are equivalent to the extended productions above.

a) $A \rightarrow 0BA_1F0$
$A_1 \rightarrow C10D \mid 0D1E \mid \varepsilon$

b) $A \rightarrow 0BA_1 1EF0 \mid 0BC1A_2F0$
$A_1 \rightarrow C10D \mid \varepsilon$
$A_2 \rightarrow 0D1E \mid \varepsilon$

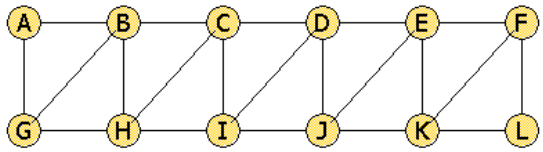
c) $A \rightarrow 0BC10D1EF0 \mid 0BF0$

d) $A \rightarrow 0BA_1F0$
$A_1 \rightarrow C10D \mid 0D1E$

Answer submitted: **b)**

You have answered the question correctly.

11. What is the size of a minimal node cover for the graph below?



Identify one of the minimal node covers below.

a) {B,C,E,F,G,I,K}

b) {A,B,C,E,H,J,K,L}

c) {A,C,E,G,H,I,L}

d) {B,C,D,F,G,I,J,K}

Answer submitted: **c)**

Your answer is incorrect.

This set does not cover all the edges. For example, it does not cover (D,J). The definition of the problem Node-Cover is in Section 10.4.3 (p. 463).

12. The language of regular expression (0+10)* is the set of all strings of 0's and 1's such that every 1 is immediately followed by a 0. Describe the complement of this language (with respect to the alphabet {0,1}) and identify in the list below the regular expression whose language is the complement of L ((0+10)*).

a) (0+1)*11(0+1)*

b) (0+1)*1(ε+11(0+1)*)

c) (0+1)*(1+11)(0+1)*

d) (0+1)*11(0+10)* + (0+1)*1

Answer submitted: **d)**

You have answered the question correctly.

13. Consider the grammars:

G_1: S → AB, A → aAA|ε , B → abBB|ε
G_2:S → CB, C → aCC|aC|a, B → abBB|abB|ab
G_3:S → CB|C|B| ε , C → aCC|aC|a, B → abBB|abB|ab
G_4:S → ASB|ε, A → aA|ε, B → abB|ε
G_5:S → ASB|AB, A → aA|a, B → abB|ab
G_6:S → ASB|aab, A → aA|a, B → abB|ab

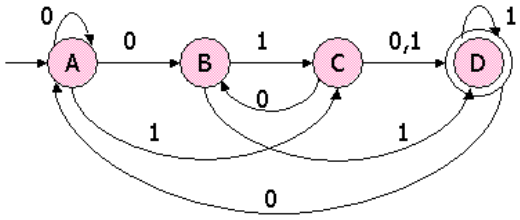
Describe the language of each of these grammars. Then, identify from the list below a pair of grammars that define the same language?

a) G_1 and G_6
b) G_4 and G_2
c) G_1 and G_4
d) G_2 and G_6

Answer submitted: **c)**

You have answered the question correctly.

14. Here is a nondeterministic finite automaton:



Convert this NFA to a DFA, using the "lazy" version of the subset construction described in Section 2.3.5 (p. 60), so only the accessible states are constructed. Which of the following sets of NFA states becomes a state of the DFA constructed in this manner?

a) {A,B,C,D}
b) {A,C,D}
c) {B,C,D}
d) {B,D}

Answer submitted: **d)**

You have answered the question correctly.

15. The polynomial-time reduction from SAT to CSAT, as described in Section 10.3.3 (p. 452), needs to introduce new variables. The reason is that the obvious manipulation of a boolean expression into an equivalent CNF expression could exponentiate the size of the expression, and therefore could not be polynomial time.

Suppose we apply this construction to the expression (u+(vw))+x, with the parse implied by the parentheses. Suppose also that when we introduce new variables, we use y1, y2,...

After constructing the corresponding CNF expression, identify one of its clauses from the list below. Note: logical OR is represented by +, logical AND by juxtaposition, and logical NOT by -.

a) (-y2+x)
b) (-y2+y1+w)
c) (-y1+w)
d) (-y3+x)

Answer submitted: **c)**

Your answer is incorrect.

Possible error: you may have reversed the constructions for AND and OR. Hint: remember that when you take the OR of two CNF expressions (i.e., AND's of clauses), introduce a new variable *y* that is added positively (as *y*) to each clause of the first expression and negatively (as NOT*y*) to each clause of the second expression. You should consult the reduction from SAT to CSAT in Section 10.3.3 (p. 452).

16. Consider the following identities for regular expressions; some are false and some are true. You are asked to decide which and in case it is false to provide the correct counterexample.

(a) R(S+T)=RS+RT
(b) (R*)*=R*
(c) (R*S*)*=(R+S)*
(d) (R+S)*=R*+S*
(e) S(RS+S)*R=RR*S(RR*S)*
(f) (RS+R)*R=R(SR+R)*

a) (c) is false and a counterexample is: R={ab},T={b}, S={b}
b) (a) is false and a counterexample is: R={ab},T={a}, S={b}
c) (d) is false and a counterexample is: R={a},T={a}, S={b}
d) (d) is true

Answer submitted: **c)**

You have answered the question correctly.

17. Here is a context-free grammar:

```
S → AB | CD
A → BG | 0
B → AD | ε
C → CD | 1
D → BB | E
E → AF | B1
F → EG | 0C
G → AG | BD
```

Find all the nullable symbols, and then use the construction from Section 7.1.3 (p. 265) to modify the grammar's productions so there are no ε-productions. The language of the grammar should change only in that ε will no longer be in the language.

a) C → CD | C
b) F → EG | 0C
c) A → BG | 0 | B | G
d) A → BG | G

Answer submitted: **d)**

Your answer is incorrect.

Since G is nullable, A → B must be a production. Also, we do not eliminate productions with bodies that consist of terminals only, as long as the body is not ε. The algorithm for modifying the grammar to eliminate ε-productions is within Section 7.1.3 starting on p. 266.

18. Let *h* be the homomorphism defined by h(a) = 01, h(b) = 10, h(c) = 0, and h(d) = 1. If we take any string *w* in (0+1)*, h^{-1}(w) contains some number of strings, N(w). For example, h^{-1}(1100) = {ddcc, dbc}, i.e., N(1100) = 2. We can calculate the number of strings in h^{-1}(w) by a recursion on the length of *w*. For example, if w = 00x for some string x, then N(w) = N(0x), since the first 0 in w can only be produced from *c*, not from *a*.

Complete the reasoning necessary to compute N(w) for any string *w* in (0+1)*. Then, choose the correct value of N(0100100).

a) 21
b) 6
c) 9

d) 64

Answer submitted: **a)**

Your answer is incorrect.

Possible error: you may be assuming that regardless of what *w* is, N(w) is the sum of N() for the two suffixes of *w* that are formed when you delete either the first symbol of *w* or the first two symbols of *w*. That is true in certain situations, e.g., if *w* begins with 01, but not always. You should check the definitions of homomorphisms (Section 4.2.3, p. 140) and their inverses (Section 4.2.4, p. 142). It may also be useful to try to develop an induction to define *N*. The ideas behind inductive proofs are described in Section 1.4 (p. 19).

19. Let G be the grammar:

```
S → SS | (S) | ε
```

L(G) is the language BP of all strings of balanced parentheses, that is, those strings that could appear in a well-formed arithmetic expression. We want to prove that L(G) = BP, which requires two inductive proofs:

1. If w is in L(G), then w is in BP.
2. If w is in BP, then w is in L(G).

We shall here prove only the second. You will see below a sequence of steps in the proof, each with a reason left out. These reasons belong to one of three classes:

A)
Use of the inductive hypothesis.

B)
Reasoning about properties of grammars, e.g., that every derivation has at least one step.

C)
Reasoning about properties of strings, e.g., that every string is longer than any of its proper substrings.

The proof is an induction on the length of w. You should decide on the reason for each step in the proof below, and then identify from the available choices a correct pair consisting of a step and a kind of reason (A, B, or C).

Basis: Length = 0.

(1)
The only string of length 0 in BP is ε because _____

(2)
ε is in L(G) because _____
Induction: |w| = n > 0.

(3)
w is of the form (x)y, where (x) is the shortest proper prefix of w that is in BP, and y is the remainder of w because _____

(4)
x is in BP because _____

(5)
y is in BP because _____

(6)
|x| < n because _____

(7)
|y| < n because _____

(8)
x is in L(G) because _____

(9)
y is in L(G) because _____

(10)
(x) is in L(G) because _____

(11)
w is in L(G) because _____

a) (3) for reason B

b) (8) for reason B

c) (1) for reason B

d) (9) for reason A

Answer submitted: **d)**

You have answered the question correctly.

20. Which of the following problems about a Turing Machine *M* does Rice's Theorem imply is undecidable?

a) Is the language of *M* empty?

b) Is there some input that causes *M* to enter more than 100 states?

c) Is there some input that causes *M* to halt after no more than 500 moves?

d) Does *M* make more than 1000 moves when started with a blank tape?

Answer submitted: **c)**

Your answer is incorrect.

Although this question is undecidable, Rice's Theorem does not imply it is undecidable. Rice's Theorem does not address properties of Turing machines, but rather of the languages they accept. In this case there are languages for which some TM's that accept them have this property, and some TM's that accept them do not have the property. See the discussion of Rice's Theorem in Section 9.3.3 (p. 397).

21. Let L be the language of all strings of a's and b's such that no prefix (proper or not) has more b's than a's. Let G be the grammar with productions

$S \rightarrow aS \mid aSbS \mid \varepsilon$

To prove that L = L(G), we need to show two things:

1. If S =>* w, then w is in L.
2. If w is in L, then S =>* w.

We shall consider only the proof of (1) here. The proof is an induction on n, the number of steps in the derivation S =>* w. Here is an outline of the proof, with reasons omitted. You need to supply the reasons.

Basis:

1) If n=1, then w is ε because _____.

2) w is in L because _____.

Induction:

3) Either (a) S => aS $\Rightarrow^{n-1}$ w or (b) S => aSbS $\Rightarrow^{n-1}$ w because _____.

4a) In case (a), w = ax, and S $\Rightarrow^{n-1}$ x because _____.

5a) In case (a), x is in L because _____.

6a) In case (a), w is in L because _____.

4b) In case (b), w can be written w = aybz, where S $\Rightarrow^{p}$ y and S $\Rightarrow^{q}$ z for some p and q less than n because _____.

5b) In case (b), y is in L because _____.

6b) In case (b), z is in L because _____.

7b) In case (b), w is in L because _____.

For which of the steps above the appropriate reason is contained in the following argument:
"All n-step derivations of w produce either ε (for n=1) or use one of the productions with at least one nonterminal in the body (for n > 1). In case the production S → aS is used, then w=ax with x being produced by a (n-1)-step derivation. In case the production S → aSbS is used then w=aybz with y and z being produced by derivations with number of steps less than n."

a) 5b

b) 3

c) 5a

d) 6b

Answer submitted: **b)**

You have answered the question correctly.

22. Which among the following languages is not regular (cannot be defined by a regular expression or finite automaton)?

a) $L=\{x \mid x=a^m b^n, n, m \text{ positive integers}\}$

b) $L=\{x \mid x=a^m(bc^k)^n, n, m, k \text{ positive integers}\}$

c) $L=\{x \mid x=a^m b^n c^k, n, m, k \text{ positive integers}\}$

d) $L=\{x \mid x=(ab^4c)^n, n \text{ a positive integer}\}$

Answer submitted: **b)**

You have answered the question correctly.

23. G_1 is a context-free grammar with start symbol S_1, and no other nonterminals whose name begins with "S." Similarly, G_2 is a context-free grammar with start symbol S_2, and no other nonterminals whose name begins with "S." S_1 and S_2 appear on the right side of no productions. Also, no nonterminal appears in both G_1 and G_2.

We wish to combine the symbols and productions of G_1 and G_2 to form a new grammar G, whose language is the union of the languages of G_1 and G_2. The start symbol of G will be S. All productions and symbols of G_1 and G_2 will be symbols and productions of G. Which of the following sets of productions, added to those of G, is guaranteed to make L(G) be $L(G_1)$ [union] $L(G_2)$?

a) $S \rightarrow S_1, S_1 \rightarrow S_2$

b) $S \rightarrow S_1S_3, S_3 \rightarrow S_2$

c) $S \rightarrow S_1S_2 \mid S_2S_1$

d) $S \rightarrow S_1, S_1 \rightarrow S_2, S_2 \rightarrow \varepsilon$

Answer submitted: **a)**

You have answered the question correctly.

24. Here is the transition function of a simple, deterministic automaton with start state A and accepting state B:

	0	**1**
A	A	B
B	B	A

We want to show that this automaton accepts exactly those strings with an odd number of 1's, or more formally:

$$\delta(A,w) = B \text{ if and only if } w \text{ has an odd number of 1's.}$$

Here, δ is the extended transition function of the automaton; that is, $\delta(A,w)$ is the state that the automaton is in after processing input string w. The proof of the statement above is an induction on the length of w. Below, we give the proof with reasons missing. You must give a reason for each step, and then demonstrate your understanding of the proof by classifying your reasons into the following three categories:

A)
Use of the inductive hypothesis.

B)
Reasoning about properties of deterministic finite automata, e.g., that if string s = yz, then $\delta(q,s) = \delta(\delta(q,y),z)$.

C)

Reasoning about properties of binary strings (strings of 0's and 1's), e.g., that every string is longer than any of its proper substrings.

Basis (|w| = 0):

(1) w = ε because _____

(2) δ(A,ε) = A because _____

(3) ε has an even number of 0's because _____

Induction (|w| = n > 0)

(4) There are two cases: (a) when w = x1 and (b) when w = x0 because _____
Case (a):

(5) In case (a), w has an odd number of 1's if and only if x has an even number of 1's because _____

(6) In case (a), δ(A,x) = A if and only if w has an odd number of 1's because _____

(7) In case (a), δ(A,w) = B if and only if w has an odd number of 1's because _____
Case (b):

(8) In case (b), w has an odd number of 1's if and only if x has an odd number of 1's because _____

(9) In case (b), δ(A,x) = B if and only if w has an odd number of 1's because _____

(10) In case (b), δ(A,w) = B if and only if w has an odd number of 1's because _____

a) (2) for reason C.
b) (6) for reason A.
c) (5) for reason A.
d) (1) for reason B.

Answer submitted: **b)**

You have answered the question correctly.

25. Here are the transitions of a deterministic pushdown automaton. The start state is q_0, and *f* is the accepting state.

State-Symbol	a	b	ε
q_0-Z_0	(q_1,AZ_0)	(q_2,BZ_0)	(f,ε)
q_1-A	(q_1,AA)	(q_1,ε)	-
q_1-Z_0	-	-	(q_0,Z_0)
q_2-B	(q_3,ε)	(q_2,BB)	-
q_2-Z_0	-	-	(q_0,Z_0)
q_3-B	-	-	(q_2,ε)
q_3-Z_0	-	-	(q_1,AZ_0)

Describe informally what this PDA does. Then, identify below, the one input string that takes the PDA into state q_3 (with any stack).

a) ababba
b) babbbaa
c) abbba
d) aabbbbb

Answer submitted: **c)**

You have answered the question correctly.