Gradiance Online Accelerated Learning

Zayd

- Home Page
- Assignments Due
- Progress Report
- Handouts
- Tutorials
- Homeworks
- Lab Projects
- Log Out

Help

Copyright © 2007-2013 Gradiance Corporation.

Submission number:	87994
Submission certificate:	CE513683
Submission time:	2014-05-18 03:50:04 PST (GMT - 8:00)

Number of questions:	25
Positive points per question:	4.0
Negative points per question:	0.0
Your score:	100

1. Consider the grammars:

G_1:S → AB | a | abC, A → b, C → abC | c

G_2:S → a | b | cC, C → cC | c

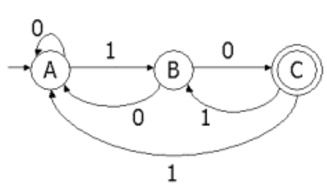
These grammars do not define the same language. To prove, we use a string that is generated by one but not by the other grammar. Which of the following strings can be used for this proof?

a) cc
b) aba
c) abababab cccc
d) cacaba

Answer submitted: **a)**

You have answered the question correctly.

2. The following nondeterministic finite automaton:



accepts which of the following strings?

a) 1011010
b) 00110100
c) 1100101
d) 010111

Answer submitted: **a)**

You have answered the question correctly.

3. Apply the CYK algorithm to the input ababaa and the grammar:

```
S → AB | BC
A → BA | a
```

```
B → CC | b
C → AB | a
```

Compute the table of entries X_{ij} = the set of nonterminals that derive positions *i* through *j*, inclusive, of the string ababaa. Then, identify a true assertion about one of the X_{ij}'s in the list below.

a) $X_{34} = \{C\}$
b) $X_{15} = \{S,C\}$
c) $X_{26} = \{S,A\}$
d) $X_{36} = \{S,A,C\}$

Answer submitted: **c)**

You have answered the question correctly.

4. Here is a context-free grammar G:

```
S → AB
A → 0A1 | 2
B → 1B | 3A
```

Which of the following strings is in L(G)?

a) 000211111321
b) 21113021
c) 021131021
d) 0021113002111

Answer submitted: **b)**

You have answered the question correctly.

5. Consider the following languages and grammars. G_1: $S \rightarrow aA|aS$, $A \rightarrow ab$
G_2: $S \rightarrow abS|aA$, $A \rightarrow a$
G_3: $S \rightarrow Sa|AB$, $A \rightarrow aA|a$, $B \rightarrow b$
G_4: $S \rightarrow aS|b$
L_1: $\{a^ib|\ i=1,2,...\}$
L_2: $\{(ab)^iaa|\ i=0,1,...\}$
L_3: $\{a^ib|\ i=2,3,...\}$
L_4: $\{a^iba^j|\ i=1,2,..., j=0,1,...\}$
L_5: $\{a^ib|\ i=0,1,...\}$

Match each grammar with the language it defines. Then, identify a correct match from the list below.

a) G_1 defines L_4.
b) G_4 defines L_5.
c) G_3 defines L_3.
d) G_1 defines L_2.

Answer submitted: **b)**

You have answered the question correctly.

6. Find, in the list below, a regular expression whose language is the reversal of the language of this regular expression: 10(2+3). Recall that the reversal of a language is formed by reversing all its strings, and the reversal of a string $a_1a_2...a_n$ is $a_n...a_2a_1$.

a) (2+3)01
b) 1(2+3)0
c) 01(2+3)
d) 0(2+3)1

Answer submitted: **a)**

You have answered the question correctly.

7. Identify from the list below the regular expression that generates all and only the strings over alphabet {0,1} that end in 1.

a) (00+01+10+11)*1

b) (0*1)*

c) $(0+1)^*1^+$

d) $(0^+1^+)^*1$

Answer submitted: **c)**

You have answered the question correctly.

8. Which of the following pairs of grammars define the same language?

a) G_1: S → SaBaS|aca, B → ScS|ε
G_2: S → SaAaS|ε, A → cS

b) G_1: S → AB, A → aAA|ε , B → baB|ε
G_2: S → CB|C|B, C → aCC|aC|a, B → baBB|baB|ba

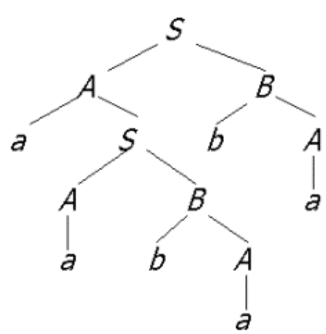
c) G_1: S → AB, A → aAA|ε , B → baBB|ε
G_2: S → CB|C|B|ε , C → aCC|aC|a, B → baBB|baB|ba

d) G_1: S → SaScSaS|aca|ε
G_2: S → SaBaS|aca, B → cS|ε

Answer submitted: **c)**

You have answered the question correctly.

9. The parse tree below represents a rightmost derivation according to the grammar `S → AB, A → aS|a, B → bA`.



Which of the following is a right-sentential form in this derivation?

a) Aba

b) aaBB

c) AbaS

d) aSB

Answer submitted: **a)**

You have answered the question correctly.

10. Here are eight simple grammars, each of which generates an infinite language of strings. These strings tend to look like alternating *a*'s and *b*'s, although there are some exceptions, and not all grammars generate all such strings.

1. S → abS | ab
2. S → SS | ab
3. S → aB; B → bS | a
4. S → aB; B → bS | b
5. S → aB; B → bS | ab
6. S → aB | b; B → bS
7. S → aB | a; B → bS
8. S → aB | ab; B → bS

The initial symbol is S in all cases. Determine the language of each of these grammars. Then, find, in the list below, the pair of grammars that define the same language.

a) G1: S → abS, S → ab
G2: S → aB, B → bS, B → b

b) G1: S → aB, B → bS, B → b
G2: S → aB, B → bS, S → b

c) G1: S → aB, B → bS, B → ab
G2: S → SS, S → ab

d) G1: S → aB, B → bS, B → ab
G2: S → aB, B → bS, S → ab

Answer submitted: **a)**

You have answered the question correctly.

11. To prove A AND (NOT B) → C OR (NOT D) by contradiction, which of the statements below would we prove? Note: each of the choices is simplified by pushing NOT's down until they apply only to atomic statements A through D.

a) ((NOT A) AND B AND (NOT C) AND D) → false
b) ((NOT B) AND A AND (NOT C) AND D) → false
c) ((NOT B) AND A AND (NOT D) AND C) → false
d) ((NOT B) AND A AND C AND (NOT D)) → false

Answer submitted: **b)**

You have answered the question correctly.

12. 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
```

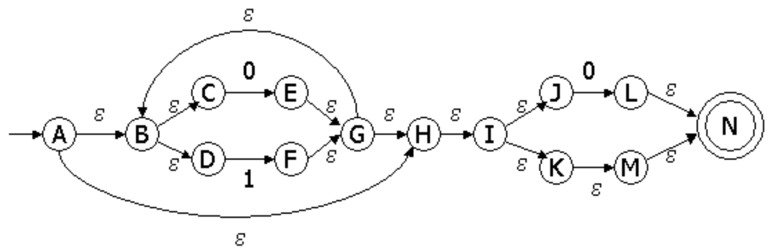
Find all the nullable symbols (those that derive ε in one or more steps). Then, identify the true statement from the list below.

a) E is not nullable.
b) D is not nullable.
c) G is not nullable.
d) F is nullable.

Answer submitted: **a)**

You have answered the question correctly.

13. Here is an epsilon-NFA:



Suppose we construct an equivalent DFA by the construction of Section 2.5.5 (p. 77). That is, start with the epsilon-closure of the start state A. For each set of states *S* we construct (which becomes one state of the DFA), look at the transitions from this set of states on input symbol 0. See where those transitions lead, and take the union of the epsilon-closures of all the states reached on 0. This set of states becomes a state of the DFA. Do the same for the transitions out of *S* on input 1. When we have found all the sets of epsilon-NFA states that are constructed in this way, we have the DFA and its transitions.

Carry out this construction of a DFA, and identify one of the states of this DFA (as a subset of the epsilon-NFA's states) from the list below.

a) BCDFGHIJK

b) ABCD

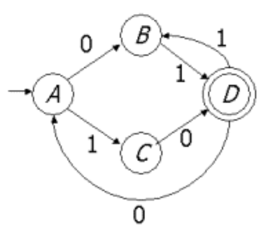
c) BCDGHIJKMN

d) BCDEGHIJKLMN

Answer submitted: **d)**

You have answered the question correctly.

14. The finite automaton below:



accepts no word of length zero, no word of length one, and only two words of length two (01 and 10). There is a fairly simple recurrence equation for the number N(k) of words of length k that this automaton accepts. Discover this recurrence and demonstrate your understanding by identifying the correct value of N(k) for some particular k. Note: the recurrence does not have an easy-to-use closed form, so you will have to compute the first few values by hand. You do not have to compute N(k) for any k greater than 14.

a) N(13) = 624

b) N(12) = 10

c) N(13) = 16

d) N(12) = 50

Answer submitted: **d)**

You have answered the question correctly.

15. 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	Move
	Symbol	
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)

Your problem is to describe the property of an input string that makes M halt. Identify a string that makes M halt from the list below.

a) 010001

b) 0100

c) 11010

d) 00001

Answer submitted: **c)**

You have answered the question correctly.

16. Design the minimum-state DFA that accepts all and only the strings of 0's and 1's that have 110 as a substring. To verify that you have designed the correct automaton, we will ask you to identify the true statement in a list of choices. These choices will involve:

1. The number of *loops* (transitions from a state to itself).
2. The number of transitions into a state (including loops) on input 1.
3. The number of transitions into a state (including loops) on input 0.

Count the number of transitions into each of your states ("in-transitions") on input 1 and also on input 0. Count the number of loops on input 1 and on input 0. Then, find the true statement in the following list.

a) There are two loops on input 0 and two loops on input 1.

b) There are three states that have one in-transition on input 1.

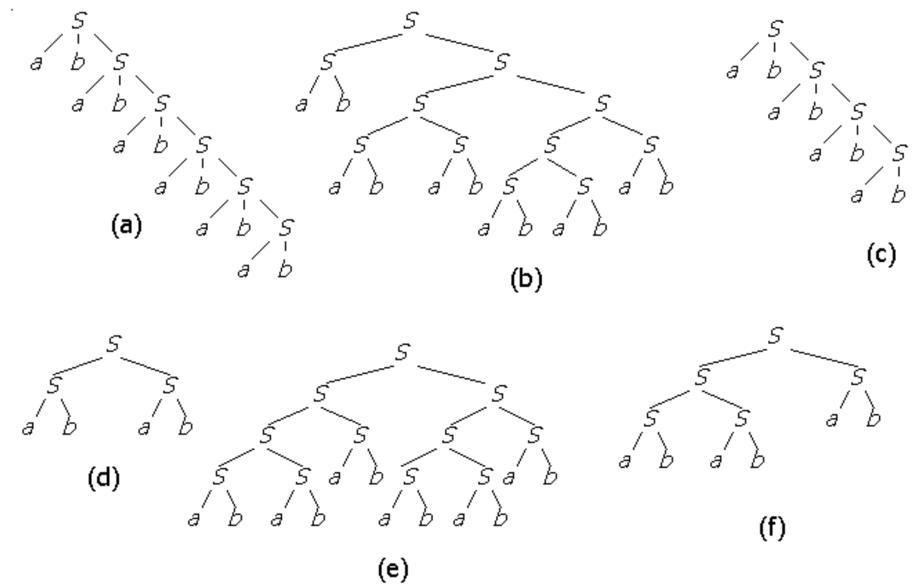
c) There is one loop on input 0 and two loops on input 1.

d) There are two states that have two in-transitions on input 1.

Answer submitted: **a)**

You have answered the question correctly.

17. Consider the grammar: S → SS, S → ab. Identify in the list below the one set of parse trees which includes a tree that is NOT a parse tree of this grammar?



a) {(f)}
b) {(b),(d),(e)}
c) {(a),(b)}
d) {(b),(d),(f)}

Answer submitted: **c)**

You have answered the question correctly.

18. Consider the pushdown automaton with the following transition rules:

1. $\delta(q,0,Z_0) = \{(q,XZ_0)\}$
2. $\delta(q,0,X) = \{(q,XX)\}$
3. $\delta(q,1,X) = \{(q,X)\}$
4. $\delta(q,\varepsilon,X) = \{(p,\varepsilon)\}$
5. $\delta(p,\varepsilon,X) = \{(p,\varepsilon)\}$
6. $\delta(p,1,X) = \{(p,XX)\}$
7. $\delta(p,1,Z_0) = \{(p,\varepsilon)\}$

From the ID $(p,1101,XXZ_0)$, which of the following ID's can NOT be reached?

a) $(p,101,\varepsilon)$
b) $(p,01,\varepsilon)$
c) $(p,01,Z_0)$
d) $(q,01,\varepsilon)$

Answer submitted: **d)**

You have answered the question correctly.

19. For the purpose of this question, we assume that all languages are over input alphabet {0,1}. Also, we assume that a Turing machine can have any fixed number of tapes.

Sometimes restricting what a Turing machine can do does not affect the class of languages that can be recognized --- the restricted Turing machines can still be designed to accept any recursively enumerable language. Other restrictions limit what languages the Turing machine can accept. For example, it might limit the languages to some subset of the recursive languages, which we know is smaller than the recursively enumerable languages. Here are some of the possible restrictions:

1. Limit the number of states the TM may have.

2. Limit the number of tape symbols the TM may have.
3. Limit the number of times any tape cell may change.
4. Limit the amount of tape the TM may use.
5. Limit the number of moves the TM may make.
6. Limit the way the tape heads may move.

Consider the effect of limitations of these types, perhaps in pairs. Then, from the list below, identify the combination of restrictions that allows the restricted form of Turing machine to accept all recursively enumerable languages.

a) Allow the TM to run for only n^{10} moves when the input is of length n.

b) Allow the TM to use only n^2 tape cells when the input is of length n.

c) Allow the TM to use only n^3 tape cells when the input is of length n.

d) Allow only two input symbols, 0 and 1, and one other tape symbol, B.

Answer submitted: **d)**

You have answered the question correctly.

20. Here are seven regular expressions:

1. (0*+10*)*
2. (0+10)*
3. (0*+10)*
4. (0*+1*)*
5. (0+1)*
6. (0+1*0)*
7. (0+1*)*

Determine the language of each of these expressions. Then, find in the list below a pair of equivalent expressions.

a) (0+1)* and (0+1*0)*

b) (0+1)* and (0+1*)*

c) (0+1*0)* and (0*+1*)*

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

Answer submitted: **b)**

You have answered the question correctly.

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 → aS | aSbS | ε
```

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 $=>^{n-1}$ w or (b) S => aSbS $=>^{n-1}$ w because _______.

4a) In case (a), w = ax, and S $=>^{n-1}$ 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 =>^{p} y and S =>^{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:
"The following two statements are true
(i) if string x has no prefix with more b's than a's, then neither does string ax,
(ii) if strings y and z are such that no prefix has more b's than a's, then neither does string aybz."

a) 2
b) 6a
c) 4a
d) 3

Answer submitted: **b)**

You have answered the question correctly.

22. Consider the grammar G with start symbol S:

S → bS | aA | b
A → bA | aB
B → bB | aS | a

Which of the following is a word in L(G)?

a) bababbabaababaa
b) babbbabaaaa
c) ababbbbbbb
d) baababaa

Answer submitted: **b)**

You have answered the question correctly.

23. Let L_1 and L_2 be two languages produced by grammars of a certain type. Let L be the language which is the concatenation of L_1 and L_2. We want to tell for various types of grammars that produce L_1 and L_2 what type is the concatenation L. Choose the triple ($type_1$, $type_2$, $type_3$) so that when the grammar that produces the language L_1 is of $type_1$ and the grammar that produces the language L_2 is of $type_2$, then the grammar that produces the concatenation language L may not be of $type_3$.

Note: A *linear grammar* is a context-free grammar in which no production body has more than one occurrence of one variable. For example, A → 0B1 or A → 001 could be productions of a linear grammar, but A → BB or A → A0B could not. A *linear language* is a language that has at least one linear grammar.

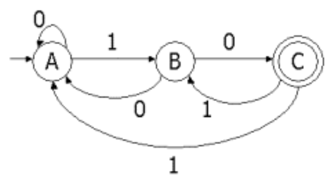
a) (linear,linear,linear)
b) (linear,linear,context-free)
c) (regular,regular,context-free)
d) (context-free,context-free,context-free)

Answer submitted: **a)**

You have answered the question correctly.

24.

When we convert an automaton to a regular expression, we need to build expressions for the labels along paths from one state to another state that do not go through certain other states. Below is a nondeterministic finite automaton with three states. For each of the six orders of the three states, find regular expressions that give the set of labels along all paths from the first state to the second state that never go through the third state.



Then identify one of these expressions from the list of choices below.

a) 0 represents the paths from B to C that do not go through A.

b) (0+1010)*1 represents the paths from A to B that do not go through C.

c) (01)*0 represents the paths from B to C that do not go through A.

d) 0(10)* represents the paths from B to A that do not go through C.

Answer submitted: **c)**

You have answered the question correctly.

25. If *h* is the homomorphism defined by h(a) = 0 and h(b) = ε, which of the following strings is in $h^{-1}(000)$?

a) aabbaa

b) aababb

c) abbbabaab

d) babab

Answer submitted: **b)**

You have answered the question correctly.