CSC 4351 Midterm Exam

1. Consider the following grammar:
S -> E $
E -> id
E -> id (E)
E -> E + id
Suppose we are using a shift-reduce parser for parsing either of the inputs "id (id) $" or "id + id $". What parse conflict (if any) does an LR(0) parser have once the first id is on the stack and either(or+would come next in the input?

S/R conflict: shift or reduce using rule E->id

2. Given the following NFA corresponding to the regular expression a(b|c)*d:

When translating this NFA into a DFA using the subset construction algorithm, which NFA states are in the start state of the DFA?

The correct answer is: 1

3. In Fortran FORMAT statements there can appear a lexical entity known as a Hollerith constant (named after Herman Hollerith, the inventor of the punch card). It consists of some decimal digits giving a count (say n), the letter H, and then n characters. The overall effect is to form a string constant n characters long. Examples include 3HYES, 2HNO, and 5HX Y Z (with one space between X and Y and one space between Y and Z). These strings are equivalent to the C strings "YES", "NO", and "X Y Z", respectively. The string 3HNO is not a valid Hollerith constant, since the number of characters after the H does not match the count.

Read each character one by one from the input. Then keep reading until you hit a digit then keep collecting digits until you hit a non digit. Count the number of digits and this will give you the Hollerith constant. If the next character happens to be H keep collecting characters until you have the specific amount. Once that is finished Holleriths constant is completed.

	Is it possible to write a handwritten scanner in Java that reads individual characters and recognizes Hollerith constants? If no, explain why not. If yes, explain how you would do it.	
4.	Consider the following grammar: S -> a A a b S -> a A c A -> a A -> Which grammar rules are in the LL parse table entry for non-terminalS and tokena? I.e., with lookaheada, how could a parse functionparseS()choose to parse the input?	The correct answers are: S -> a A a b, S -> a A c
5.	Consider the following grammar: S -> a A a b S -> a A c A -> a A -> Which terminal symbols are in FOLLOW(A)? Question 5	The correct answers are: a, c
6.	Translate the regular expression (a|b)c into an NFA using Thompson's construction. How many states and epsilon edges are there in the resulting NFA?	7 states, 4 epsilon edges
7.	In Fortran FORMAT statements there can appear a lexical entity known as a Hollerith constant (named after Herman Hollerith, the inventor of the punch card). It consists of some decimal digits giving a count (say n), the letter H, and	Read the number and 'H', store the count, switch to HOLLERITH state, and then read individual characters while decrementing the count.

then n characters. The overall effect is to form a string constant n characters long. Examples include 3HYES, 2HNO, and 5HX Y Z (with one space between X and Y and one space between Y and Z). These strings are equivalent to the C strings "YES", "NO", and "X Y Z", respectively. The string 3HNO is not a valid Hollerith constant, since the number of characters after the H does not match the count.
Is it possible to implement a lexical analyzer for Hollerith constants in JLex with the help of start states? If no, explain why not. If yes, explain how you would do it.

8. Consider the following grammar:
S -> a A a b
S -> a A c
A -> a
A ->
Is S nullable?

The correct answer is: No

9. Consider the following grammar with line numbers:
1: S -> E $
2: E -> id
3: E -> id (E)
4: E -> E + id
where $ represents end of file. Work out a shift-reduce parse for the input "id + id $" on paper with columns for stack, input, and the parse action until the input is accepted.

S
R2
S
S
R4
A

	In the textbox below, enter the parse actions (i.e., only the third column from your parse trace), in the order in which they are performed, with one parse action per line. For a shift or accept action it's good enough to show a single character s or a, respectively. For a reduce action, show an r and indicate the grammar rule by which the top of the stack is reduced using its line number.	
10.	The regular expression ab* is equivalent to which of the following regular expressions?	The correct answer is: a(b*)
11.	Given the following NFA corresponding to the regular expression a(b|c)*d: Suppose we have a DFA state X = {5;3,4,6,8,9}. If we construct a DFA edge from state X on input c, which NFA states are in the resulting DFA state?	The correct answers are: 3, 4, 6, 7, 8, 9
12.	If for a given grammar, an LALR(1) parser has a conflict then the grammar also cannot be parsed with a recursive-descent parser.	The correct answer is 'False'.
13.	If for a given grammar, an LALR(1) parser has a S/R conflict then the grammar is not LR(1), either.	The correct answer is 'True'.
14.	Consider the following grammar: S -> a A a b S -> a A c A -> a A -> Which grammar rules are in the LL parse table entry for non-terminalS and column $ (EOF)? I.e.,	The correct answer is: This parse table entry is empty.

	when seeing end of file, how would a parse function parseS() choose to parse the input?	
15.	Consider the following grammar: S -> a A a b S -> a A c A -> a A -> Which terminal symbols are inFOLLOW(S)?	The correct answer is: $ (indicating end of file)
16.	Consider the following grammar: S -> a A a b S -> a A c A -> a A -> Which grammar rules are in the LL parse table entry for non-terminal A and token a? I.e., with lookahead a, how could a parse function parseA() choose to parse the input?	The correct answers are: A -> a, A ->
17.	The regular expression ab|c is equivalent to which of the following regular expressions?	The correct answer is: (ab)|c
18.	If for a given grammar, an LR(0) parser has a S/R conflict but an LR(1) parser has no conflict, then the grammar must be SLR as well.	The correct answer is 'False'.
19.	Given the following NFA corresponding to the regular expression a(b|c)*d: What is the minimum number of states that DFA needs that recognizes the same language as this NFA? You can either translate this NFA into a DFA	The correct answer is: 3

	and then combine equivalent states or construct a DFA by hand. In either case, you should end up with the same DFA.	
20.	An NFA generated with Thompson's construction can have multiple final states.	The correct answer is 'False'.
21.	Consider the following grammar with line numbers: 1: S -> E $ 2: E -> id 3: E -> id (E) 4: E -> E + id where $ represents end of file. Work out a shift-reduce parse for the input "id (id) $" on paper with columns for stack, input, and the parse action until the input is accepted. In the textbox below, enter the parse actions (i.e., only the third column from your parse trace), in the order in which they are performed, with one parse action per line. For a shift or accept action it's good enough to show a single character s or a, respectively. For a reduce action, show an r and indicate the grammar rule by which the top of the stack is reduced using its line number.	S S S R2 S R3 A
22.	If for a given grammar, an LR(0) parser has a conflict but the conflict can be resolved by an SLR parser, then the grammar is also LALR(1).	The correct answer is 'True'.
23.	Given the following NFA corresponding to the regular expression a(b|c)*d:	The correct answer is: 10

	Suppose we have a DFA state X = {5;3,4,6,8,9}. If we construct a DFA edge from state X on input d, which NFA states are in the resulting DFA state?	
24.	Consider the following grammar: S -> a A a b S -> a A c A -> a A -> Of the four language classes we have discussed (regular, context-free, context-sensitive, and unlimited), what is the smallest category into which the language generated by this grammar fits? Justify your answer.	Regular Language because all of the terminals and nonterminals are very simple and straightforward and can be recognized by the finite automata Why? Regular: aaab | aab | aac | ac
25.	Consider the following grammar: S -> a A a b S -> a A c A -> a A -> Which terminal symbols are inFIRST(A)?	The correct answer is: a
26.	An LALR(1) parser can always successfully resolve a S/R conflict by giving preference to shift.	The correct answer is 'False'.
27.	Given the following NFA corresponding to the regular expression a(b|c)*d: Suppose we have a DFA state X = {5;3,4,6,8,9}. If we construct a DFA edge from state X on input b, which NFA states are in the resulting DFA state?	The correct answers are: 3, 4, 5, 6, 8, 9

28.	Consider the following grammar: S -> a A a b S -> a A c A -> a A -> Which terminal symbols are inFIRST(S)?	The correct answer is: a
29.	Translate the regular expression ab*c into an NFA using Thompson's construction. How many states and epsilon edges are there in the resulting NFA?	The correct answer is: 6 states, 4 epsilon edges
30.	Consider the following grammar: S -> a A a b S -> a A c A -> a A -> Is this grammar LL(k)? I.e., can a top-down parser with arbitrary lookahead parse the language according to this grammar?	The correct answer is: Yes, because all sentences in the language are of finite length.
31.	Consider the following grammar: S -> a A a b S -> a A c A -> a A -> IsA nullable?	The correct answer is: Yes
32.	Which of the following languages is a regular language?	The correct answers are: { an b | n >= 0 }, { an bn | 0 <= n <= 10 }, The language of all possible floating-point constants.

33.	In Fortran FORMAT statements there can appear a lexical entity known as a Hollerith constant (named after Herman Hollerith, the inventor of the punch card). It consists of some decimal digits giving a count (say n), the letter H, and then n characters. The overall effect is to form a string constant n characters long. Examples include 3HYES, 2HNO, and 5HX Y Z (with one space between X and Y and one space between Y and Z). These strings are equivalent to the C strings "YES", "NO", and "X Y Z", respectively. The string 3HNO is not a valid Hollerith constant, since the number of characters after the H does not match the count. Suppose we limit the length of Hollerith constants to no more than 50 characters. Is it possible to specify the language of Hollerith constants of limited length with an NFA or a DFA?	The correct answer is: Yes, because we could construct an automaton where different states correspond to the possible counts and then count down by having a state transition with each character.
34.	Given ASCII as the underlying character set, the regular expression [a-c] is equivalent to which of the following regular expressions?	The correct answer is: a|b|c
35.	Using start states in JLex makes it easier to process certain lexemes (such as translating escape character sequences inside strings), but it is not necessary to use start states. Any language that can be recognized with JLex with start states could also be recognized with JLex without the use of start states.	The correct answer is 'False'.
36.		The correct answer is: A ->

Consider the following grammar:
S -> a A a b
S -> a A c
A -> a
A ->
Which grammar rules are in the LL parse table entry for non-terminal A and token c? I.e., with look ahead c, how could a parse function parseA()choose to parse the input?

37. In Fortran FORMAT statements there can appear a lexical entity known as a Hollerith constant (named after Herman Hollerith, the inventor of the punch card). It consists of some decimal digits giving a count (say n), the letter H, and then n characters. The overall effect is to form a string constant n characters long. Examples include 3HYES, 2HNO, and 5HX Y Z (with one space between X and Y and one space between Y and Z). These strings are equivalent to the C strings "YES", "NO", and "X Y Z", respectively. The string 3HNO is not a valid Hollerith constant, since the number of characters after the H does not match the count.

Is it possible to specify the language of Hollerith constants (i.e., the set of only valid Hollerith constants) with an NFA or a DFA?

The correct answers are: No, since a finite automaton cannot compare the number of characters after the H to the count., No, this is a context-sensitive language and a finite automaton cannot recognize a context-sensitive language.

38. In Fortran FORMAT statements there can appear a lexical entity known as a Hollerith constant (named after Herman Hollerith, the inventor of

The correct answers are: No, since a regular language cannot express the constraint that the number of the punch card). It consists of some decimal digits giving a count (say n), the letter H, and then n characters. The overall effect is to form a string constant n characters long. Examples include 3HYES, 2HNO, and 5HX Y Z (with one space between X and Y and one space between Y and Z). These strings are equivalent to the C strings "YES", "NO", and "X Y Z", respectively. The string 3HNO is not a valid Hollerith constant, since the number of characters after the H does not match the count.

Is the language of Hollerith constants (i.e., the set of only valid Hollerith constants) a regular language?

characters after H must match the count., No, this is a context-sensitive language.