Midterm 1

Due No due date **Points** 24 **Questions** 19

Available until Feb 16 at 11:20pm **Time Limit** 62 Minutes

Instructions

- You have **60 minutes** to complete this exam.
- The exam will be stopped at **8pm**.
- This exam is closed book/closed notes; **you may not communicate with anyone** other than the instructor and/or TAs and LAs during the exam.
- Any question about the exam can be posted in campuswire; make sure it's **visible only to TAs and instructors**. We will also make broad announcements and clarifications in campuswire. You can join campuswire using https://campuswire.com/p/GB6D343A6 (https://campuswire.com/p/GB6D343A6) and sign-up code of 8502.
- Once you begin the exam, you must complete it within the time limit. If you logout of Canvas or close your browser after you enter the exam, the **countdown will not stop**.
- Any kind of **cheating may result in failing the course.**

This quiz was locked Feb 16 at 11:20pm.

Attempt History

	Attempt	Time	Score
LATEST	Attempt 1	62 minutes	14.5 out of 24

Score for this quiz: **14.5** out of 24

Submitted Feb 16 at 8:02pm

This attempt took 62 minutes.

Question 1

0.5 / 0.5 pts

For an unambiguous grammar, there is exactly one parse tree for any terminal string that belongs
to the language defined by the grammar.

Correct!

● True

○ False

Question 2

0.5 / 0.5 pts

Static binding means that the association between a program entity and a property is decided at
compile time.

Correct!

● True

○ False

Question 3

0.5 / 0.5 pts

Dynamic binding is also called late binding.

Correct!

● True

○ False

Question 4

0.5 / 0.5 pts

Any language that can be defined by a regular expression can also be defined by a context-free
grammar.

Correct!

● True

○ False

Question 5

0.5 / 0.5 pts

BNF grammars do not allow left-recursive rules.

○ True

Correct!

● False

Question 6

0.5 / 0.5 pts

In C, the scope of a static local variable is the entire program.

○ True

Correct!

● False

Question 7

0 / 0.5 pts

An operator's precedence determines whether it associates to the left or right.

ou Answered

● True

orrect Answer

○ False

Question 8

0 / 0.5 pts

For a deterministic FSA, a state may have multiple outgoing edges labeled with the same input character.

ou Answered

● True

orrect Answer

○ False

Question 9

1 / 1 pts

Suppose r stands for the regular expression "(a|b)*c*". Which one of the following strings is NOT a sentence of the language generated by r.

○ aaabbbccc

Correct!

● aaacccbbb

○ bbbaaa

○ aaa

Question 10

1 / 1 pts

Which one of the following is true of E-BNF grammars (extended BNF)?

○ E-BNF grammars allow one to describe the syntax that can not be described by BNF grammars

Correct!

● Any E-BNF grammar can also be expressed in a BNF grammar

○ E-BNF grammars allow one to describe the semantics of a programming language

○ Any E-BNF grammar can also be expressed in a regular expression

Question 11

0 / 1 pts

Which one of the following is true of l-values and r-values?

orrect Answer

○ An l-value refers to a variable's memory location while an r-value to its current value

ou Answered

● l-values are always to the left of r-values

○ l-values are local and r-values are relative.

○ An l-value is a logical value, and an r-value is a real value.

Question 12

0 / 1 pts

Which following scheme of variable scoping is used by most modern programming languages?

ou Answered

● Both static and dynamic scoping

○ Dynamic scoping

orrect Answer

○ Static scoping

Question 13

1 / 1 pts

The following BNF grammar defines the grammar of the scientific notation for non-negative floating point numbers.

<SNFloat> -> <Float> | <Float>E<Exponent>

<Float> -> <NonZeroDigit> | <NonZeroDigit>.<Num>

<Exponent> -> <Num> | +<Num> | -<Num>

<Num> -> <Digit> | <Digit><Num>

<Digit> -> 0 | <NonZeroDigit>

<NonZeroDigit> -> 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

Show the left-most derivation for 5E+2; show every step.

Your Answer:

<SNFloat> -> <Float>E<Exponent>

<Float>E<Exponent>-><NonZeroDigit>E<Exponent>

<NonZeroDigit>E<Exponent>->5E<Exponent>

5E<Exponent>->5E+<Num>

5E+<Num>->5E+<Digit>

5E+<Digit>->5E+<NonZeroDigit>

5E+<NonZeroDigit>->5E+2

Question 14

1 / 1 pts

The following BNF grammar defines the grammar of the scientific notation for non-negative floating point numbers.

<SNFloat> -> <Float> | <Float>E<Exponent>

<Float> -> <NonZeroDigit> | <NonZeroDigit>.<Num>

<Exponent> -> <Num> | +<Num> | -<Num>

<Num> -> <Digit> | <Digit><Num>

<Digit> -> 0 | <NonZeroDigit>

<NonZeroDigit> -> 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

Explain why 100.4E+1 is not part of the language defined by the grammar.

Your Answer:

because the 100 in 100.4E+1 is not accessible by the grammar

for <Float> -> <NonZeroDigit> | <NonZeroDigit>.<Num>, you can only have NonZeroDigit before the decimal point.

Question 15

1 / 1 pts

The following BNF grammar defines the grammar of the scientific notation for non-negative floating point numbers.

<SNFloat> -> <Float> | <Float>E<Exponent>

<Float> -> <NonZeroDigit> | <NonZeroDigit>.<Num>

<Exponent> -> <Num> | +<Num> | -<Num>

<Num> -> <Digit> | <Digit><Num>

<Digit> -> 0 | <NonZeroDigit>

<NonZeroDigit> -> 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

Show the right-most derivation for 3.14E3; show every step.

Your Answer:

<SNFloat> -> <Float>E<Exponent>

<Float>E<Exponent>-><Float>E<Num>

<Float>E<Num>-><Float>E<Digit>

<Float>E<Digit>-><Float>E<NonZeroDigit>

<Float>E<NonZeroDigit>-><Float>E3

<Float>E3-><NonZeroDigit>.<Num>E3

<NonZeroDigit>.<Num>E3-><NonZeroDigit>.<Digit><Num>E3

<NonZeroDigit>.<Digit><Num>E3-><NonZeroDigit>.<Digit><Digit>E3

<NonZeroDigit>.<Digit><Digit>E3-><NonZeroDigit>.<Digit><NonZeroDigit>E3

<NonZeroDigit>.<Digit><NonZeroDigit>E3-><NonZeroDigit>.<Digit>4E3

<NonZeroDigit>.<Digit>4E3-><NonZeroDigit>.<NonZeroDigit>4E3

<NonZeroDigit>.<NonZeroDigit>4E3-><NonZeroDigit>.14E3

<NonZeroDigit>.14E3->3.14E3

Question 16

3 / 3 pts

For the following C program, you will be asked to determine which variables are visible in a number of different situations. In each case, identify each variable by its name and the line number of its declaration. Please clearly label the answer for each subquestion.

```
1 int a,b;

2 void foo(int a) {
3  ...
4 }

5 void bar () {
6  int a;
7  ...
8 }
```

```
9 void main() {
10  int b;
11  ...
12 }
```

(a) (1 point) C uses static scoping. Say which variables are visible in the bodies of each of the functions: main, foo, bar.

(b) (1 point) If C used dynamic scoping and the calling sequence is that main calls bar, say which variables would be visible in bar.

(c) (1 point) If C used dynamic scoping and the calling sequence is that main calls foo, and foo calls bar, say which variables would be visible in bar.

Your Answer:

A)

main - <b,10><a,1>

bar - <a,6><b,1>

foo - <a,2><b,1>

B)

<a,6><b,10>

C)

<a,6><b,10>

Question 17

1 / 4 pts

Answer this question based on the following BNF grammar with the start non-terminal being <foo>.

<foo> -> <bar> | <bar> $ <foo>
<bar> -> <baz> | <bar> # <baz>
<baz> -> x | y | (<foo>)

- Which operator has higher precedence? $
- The associativity of the $ operator is [Select]

- The associativity of the # operator is [Select]
- Is the grammar ambiguous or unambiguous?

  [Select]

Answer 1:

orrect Answer

#

ou Answered

$

Answer 2:

Correct!

right

Answer 3:

orrect Answer

left

ou Answered

right

Answer 4:

ou Answered

ambiguous

orrect Answer

unambiguous

Question 18

1 / 3 pts

Answer the following questions based on the following grammar:

<clause> -> <clause> and <phrase> | <phrase>
<phrase> -> ...

(a) (1 point) Explain in 1 or 2 sentences why we cannot directly use recursive-descent parsing on the grammar.

(b) (1 point) Transform the grammar into an equivalent grammar on which recursive-descent parsing can be applied.

(c) (1 point) Write some pseudo code for the parsing method for nonterminal <clause> in the transformed grammar, assuming there is already a parsing method for nonterminal <phrase>.

Please clearly label your answer for each subquestion.

Your Answer:

a)

because the recursive does not guarantee to stop.

B)

<clause> -> <clause> and <phrase> | <phrase>
<phrase> ->terminate

C)

```
While input != class:

  <clause>;

   if <phrase>:

        break;
```

(-0.5) (a) The answer is partially correct: does not mention the reason. (-0.5) (b) The answer does not remove left recursion. (-1.0) (c) The parsing is incorrect.

Question 19

1.5 / 3 pts

The following E-BNF grammar defines the grammar of the scientific notation for non-negative floating point numbers. Note that "[", "]", "(", ")", "|", "{", and "}" are meta-symbols of E-BNF.

<SNFloat> -> <Float> [E<Exponent>]
<Float> -> <NonZeroDigit> [.<Num>]
<Exponent> -> [(+|-)]<Num>

<Num> -> <Digit>{<Digit>}
<Digit> -> 0 | <NonZeroDigit>
<NonZeroDigit> -> 1 | 2 | ... | 9

The rule for Exponent allows numbers such as 0023; change the rule for <Exponent> so that numbers with leading zeros are illegal in exponents (the number 0 itself should still be legal though). You can use either EBNF or BNF in your answer.

Your Answer:

<Exponent> -> (+|-)+<NonZeroDigit>+<Num>*|(+|-)+<Digit>

- "(+|-)+" not correct. - <NonZeroDigit>+ not correct - shouldn't use the regexp syntax.

Quiz Score: **14.5** out of 24