Lecturer:	*(Date)*	**Approved by:**	(Date)
(Signature and Fullname)		*(Signature and Fullname)*	

HCMC UNIVERSITY OF TECHNOLOGY **FACULTY OF CSE**	**MIDTERM TEST**	Sem. / School year	2	2022-2023
		Date		06-03-2023
	Course name	Principles of Programming Languages		
	Course ID	CO3005		
	Duration	70 minutes	Code	2210

Notes:
- Students are allowed to use one sheet of A4 handwritten reference material and a calculator.
- Students do questions on an answer sheet..
- Each question has either one correct answer or no correct answer.
- If there is no correct answer, students choose option E.
- Students submit the exam paper along with the answer sheet.

Question 1. [L.O.2.1] In the following Python code:

```
def square(f):
    def wrap(x):
        return f(x)**2
    return wrap
(1)
def double(x):
    return x * 2
print(double(3)) # result is 36
```

What should be filled in at (1) so that the result of print(double(3)) is 36?

(A) @square (B) square (C) x = square(6) (D) #square

Question 2. [L.O.1.1] Given a lexical description defined in ANTLR4 as follows:

```
FLOAT_CONSTANT: DIGIT_SEQUENCE EXPONENT? FLOAT_SUFFIX?;

fragment DIGIT_SEQUENCE: DIGIT+ ('.' DIGIT+)?;
fragment EXPONENT: ('e' | 'E') ('+' | '-')? DIGIT+;
fragment FLOAT_SUFFIX: ('f' | 'F' | 'l' | 'L');
fragment DIGIT: [0-9];
```

Which of the following strings is a correct input string for the `FLOAT_CONSTANT` token and has a correct explanation:

(A) `0.0001E-2f`, in which `E-2` is formed by `EXPONENT`
(B) `6.02e23L`, in which `e23L` is formed by `EXPONENT`
(C) `0.123_456` and there is no `FLOAT_SUFFIX` component
(D) `123.456E+7F`, in which `123.456E+7` is formed by `DIGIT_SEQUENCE`

Question 3. [L.O.2.1] According to the convention in the Python language, a `protected` attribute named `ex` should be declared by:

(A) Naming the attribute as `_ex`
(B) Naming the attribute as `__ex`
(C) Declaring `ex: private` in the `init` method
(D) Declaring `ex` with an annotation `@private_attr()`

Question 4. [L.O.1.1] Given the regular expression `a[^abc]*c` and the input strings adc, abbc, ayyyyyyyyyc, abc, aabc, axc. The number of input strings that satisfy the regular expression is

(A) 1 (B) 5 (C) 3 (D) 2

Question 5. [L.O.1.1] A Ruby programming language identifier is a string of alphanumeric characters and underscores. It must start with an underscore or a lowercase letter. Choose a regular expression that matches it.

(A) `[a-z0-9_]+` (B) `[a-z_A-Z0-9]+` (C) `[a-z_] [a-z0-9_]*` (D) `[0-9_] [a-z0-9_]+`

Question 6. [L.O.1.1] Choose a regular expression equivalent to the following regular expression: (a|b)*(abb|b)a

(A) (b*a*)*(ab)?ba (B) [a|b]*[abb|b]a (C) [ab]*[ab]?ba (D) [ab]*(ab)+ba

Question 7. [L.O.2.1] Given a list containing elements that can be nested within a list, for example:

```
nested_lst = [1, 2, [3, 4, [5, 6], 7], 8, [9]]
```

The function `flatten` can take the above list as input and return a flattened list such as `[1, 2, 3, 4, 5, 6, 7, 8, 9]`. The body of `flatten` is:

(A) `return reduce(lambda prev, curr: prev + (flatten(curr) if type(curr) is list else curr), lst, [])`
(B) `return reduce(lambda prev, curr: prev + [curr], lst, [])`
(C) `return reduce(lambda prev, curr: prev + (flatten(curr) if type(curr) is list else [curr]), lst, [])`
(D) `return reduce(lambda prev, curr: prev + curr), lst, [])`

Question 8. [L.O.2.1] Given the following Python code:

```
x,y = 0,4
while x < 3:
    x += 1
else: print(y)
```

After executing the code above,

(A) The value 4 is printed out (B) The value 3 is printed out (C) No value is printed out
(D) The code above causes a syntax error

Apply the following code snippet in Python for questions 9–10:

```
result = (lst[0] * 2) + func(x, y) - (lst[-1] if lst[1] >= -1.2 else lst[2]) % 5 # cal result
```

Question 9. [L.O.3.1] The number of tokens returned when lexing the above string is:

(A) 38 (B) 43 (C) 40 (D) 45

Question 10. [L.O.3.1] The lexeme string of the 25th token is:

(A) -1 (B)] (C) if (D) lst

Question 11. [L.O.1.2] Multiple assignment allows assigning multiple `ID` (on the left-hand side separated by commas) to multiple `exp` (on the right-hand side separated by commas) with the left-hand side and right-hand side separated by an `EQ` sign. Use the `assignment` notation to describe the multiple assignment statement, and write the right-hand side of the `assignment` statement so that the numbers of `ID` and `exp` are equal, with at least one `ID` and one `exp` in the multiple assignment statement:

(A) `(ID (CM ID)* EQ exp (CM exp)*`
(B) `ID CM assignment CM exp | ID EQ exp`
(C) `ID CM assignment CM exp | EQ`
(D) `ID EQ exp CM assignment | ID EQ exp`

Question 12. [L.O.1.2] Please indicate which of the following grammars is ambiguous:

(A) $S \to \epsilon|aSbS$ (B) $S \to \epsilon|aSa|bSb$ (C) $S \to AB|BA, A \to \epsilon|aA, B \to \epsilon|bB$
(D) $S \to \epsilon|a|b|aSa|bSb$

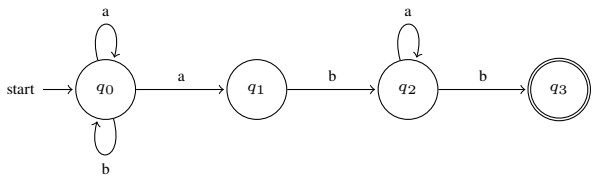
Question 13. [L.O.1.1] Given a lexical description defined in ANTLR4 as follows:

```
UNIVERSE: A* S A A A A A+;
fragment A: D | C | S;
fragment D: [0-9];
fragment C: [a-zA-Z];
fragment S: [@$!%*#?&];
```

What is the characteristic of the input string corresponding to the `UNIVERSE` token?

(A) It has at least 4 characters and must contain at least one special character (@$!%*#?&)
(B) It has at least 6 characters and must contain at least one special character (@$!%*#?&)
(C) It has at most 8 characters and must contain a lowercase letter or a digit
(D) It has at least 6 characters and when containing a lowercase letter it cannot contain an uppercase letter and vice versa

Question 14. [L.O.3.1] Given an automaton as follows:



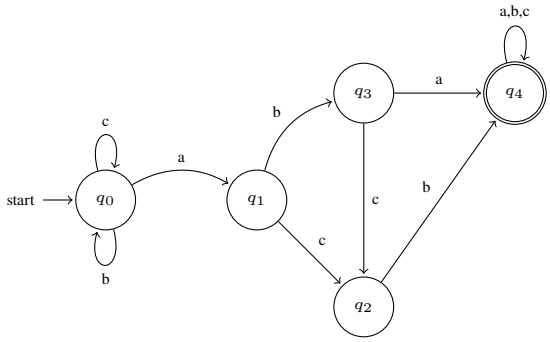
The equivalent regular expression of the above automaton is:

(A) b*a*aba+b (B) (ab)*aba*b (C) a*b*aba*b (D) [ab]*aba*b

Question 15. [L.O.2.1] Which of the following statements about Method Resolution Order (MRO) in Python is correct?

(A) MRO determines the order of imported modules in a Python file.
(B) MRO determines the order of executing a Python script.
(C) MRO determines the order of searching for a method or attribute in a class inheritance hierarchy.
(D) MRO determines the order in which the Python compiler compiles source code.

Question 16. [L.O.3.1] Given an automaton as follows:



Which of the following input strings is accepted by the automaton above:

(A) bccbabcbaab (B) abcaabcbaaa (C) baccbacbcca (D) bcabcaabcab

Question 17. [L.O.1.2] A list of `lcase` consists of consecutive `case` and may be empty. A `case` consists of the keyword `CASE`, followed by an `exp` expression and a `CL` separator, and ending with a list of `stmtlist` statements. The EBNF form of the right-hand side of the `lcase` production rule is:

(A) `CASE (exp CL stmtlist)*?`
(B) `(CASE exp* CL stmtlist*)?`
(C) `(CASE exp CL stmtlist)*`
(D) `CASE exp* CL stmtlist?`

Question 18. [L.O.2.1] A variable in Python will

(A) hold a reference to an object (B) receive a value to store (C) require a type declaration
(D) need to be declared before assigning a value

Question 19. [L.O.2.1] Given the following Python code:

```
class A:
    def who_am_i(self):
        print("A")
class B(A):
    def who_am_i(self):
        print("B")
        super().who_am_i()
class C(A):
    def who_am_i(self):
        print("C")
        super().who_am_i()
class D(B, C):
    def who_am_i(self):
        print("D")
        super().who_am_i()
class E(C, B):
    def who_am_i(self):
        print("E")
        super().who_am_i()
class F(E, D):
    def who_am_i(self):
        print("F")
        super().who_am_i()
f = F()
f.who_am_i()
```

The output of the above code is:

(A) FDBCEA (B) FBDCEA (C) FDBCAE (D) FBCDEA

Question 20. [L.O.1.2] The **`list`** notation describes a list (which may be empty) of `a` elements separated by a `C` symbol. The grammar for **`list`** is written in ANTLR as follows:

```
list: elist | (1);
elist: a C elist | (2);
```

The empty sequence denoted by the underscore symbol (_). (1) and (2) should be.

(A) _ and `a` (B) `C a` and _ (C) _ and _ (D) _ and `C`

Question 21. [L.O.1.2] A context-free grammar may include:

(A) Regular expressions for describing non-terminal symbols.
(B) A set of terminal symbols, a set of production rules.
(C) A single non-terminal symbol and a regular expression describing it
(D) Only one production rule that shows the order of tokens.

Question 22. [L.O.1.2] Consider the following code snippet in ANTLR:

```
decl: ID decl_tail;
decl_tail: CM decl | CL ID CM;
```

Which of the following right-hand sides is appropriate for the `decl` rule to be equivalent to the above code snippet?

(A) `(ID CM)* (CL ID CM)?`
(B) `(ID CM)* CL ID CM`
(C) `ID CM ID (CL ID)* CM`
(D) `ID (CM ID)* CL ID CM`

Question 23. [L.O.1.2] Which of the following statements is true about the language generated by the context-free grammar below?
$S \rightarrow AB, A \rightarrow aA|\epsilon, B \rightarrow bB|\epsilon$

(A) This language contains all strings over the alphabet {a, b}
(B) This language contains all strings of the form $a^m b^n$, where m and n are non-negative integers
(C) This language only contains the empty string
(D) This language contains all strings of the form $a^n b^n$, where n is a non-negative integer

Question 24. [L.O.2.1] A program written in language X has the following content:

- Prompt the user to input a value into variable **a** from the keyboard.
- Perform a loop **a** times with the loop body consisting of:
  - Calculate the value of variable **b**.
  - If the value of variable **b** is greater than 5, then print the value of variable **b** and exit the loop.
  - Execute a statement to print **a** with a grammar error.

During execution, sometimes the program runs successfully, while other times it throws errors. What method was used to implement the aforementioned language X?

(A) Pure interpreter
(B) Compiler
(C) Hybrid implementation
(D) Hybrid implementation with just-in-time compiler

Question 25. [L.O.1.1] Choose a regular expression that accepts at least all strings in the MATCH set, but does not accept any strings in the SKIP set:
MATCH ={Cho, chi, Chung, Che, Chan }
SKIP = {Tro, Ching, Chu, Tre, Tran}

(A) [cCT][hr][aeuio]n?g?
(B) [cC]h[aoiue]n?g?
(C) [Cc]h[oie]|Ch[au]ng?
(D) (C|c)h(o|i|e)|Ch(a|u)n?g?

Question 26. [L.O.1.1] To represent a power form a^b with a and b being positive integers in Latex, we write the base a, followed by the caret symbol, and then the exponent b. If the exponent b has two or more digits, it must be enclosed in curly braces. If b has only one digit, the use of curly braces is optional.
In ANTLR4, assuming the fragments describing the digit character is `DIGIT`, the caret symbol is `HAT`, and the opening and closing curly braces are `LP` and `RP`, respectively. Which of the following descriptions can represent the above power form?

(A) `DIGIT+ HAT (DIGIT | LP DIGIT DIGIT+ RP)`
(B) `DIGIT+ HAT (LP (DIGIT | DIGIT+) RP)`
(C) `DIGIT+ HAT DIGIT | LP DIGIT+ RP`
(D) `DIGIT+ HAT (DIGIT | LP DIGIT+ RP)`

Question 27. [L.O.1.2] The right-hand side production rule of `ui_list` describes a list that contains at least one `UI` element, written in BNF form as:

(A) `UI ui_list |`
(B) `UI ui_list`
(C) `ui_list | UI`
(D) `UI ui_list | UI`

Question 28. [L.O.1.2] Given the context-free grammar G with the set of terminal symbols as ADD, MINUS, MUL, DIV, LB, RB, set of non-terminal symbols as exp, term, fact, start symbol as exp, and the set of production rules as:
exp → term MINUS exp | term
term → fact DIV term | fact
fact → fact ADD factor | fact MUL factor | factor
factor → LB exp RB | INT
where INT is the token for integers, ADD is the token for addition, MINUS is the token for subtraction, MUL is the token for multiplication, DIV is the token for integer division, LB is the token for left parenthesis, and RB is the token for right parenthesis.
To evaluate the value of the input string, we need to determine the precedence and associativity of the operators and apply the rules of the grammar to generate the parse tree. The value of 100 - 4 / 4 + 2 * 3 / 9 - 10?

(A) 108
(B) 109
(C) 90
(D) 89

Question 29. [L.O.1.1] In language X, programmers can write inexact integer constants by replacing unknown digits with the `#` symbol preceded by a digit. An inexact integer constant must start with at least two digits. For example, `12#345` and `123#45#` are valid inexact integer constants, but `123##45` is invalid.
In ANTLR4, assuming the fragments `DIGIT` and `SHARP` describe the digit and `#` characters, respectively, the following description represents inexact integer constants in language X:

Ⓐ `DIGIT (DIGIT SHARP?)+` Ⓑ `DIGIT (DIGIT SHARP?)*` Ⓒ `DIGIT (DIGIT SHARP)+` Ⓓ `DIGIT (DIGIT* SHARP)*`

Question 30. [L.O.2.1] Please provide the output of the following code snippet (in Python):

```
def square(x): return x ** 2
def double(x): return x * 2
numbers = [1, 2, 3, 4, 5]
result = map(square, filter(lambda x: x % 2 == 0, map(double, numbers)))
print(list(result))
```

Ⓐ `[4, 16]` Ⓑ `[4, 16, 36, 64, 100]` Ⓒ `[1, 4, 9, 16, 25]` Ⓓ `[2, 4, 6, 8, 10]`

Question 31. [L.O.1.2] To list a set of integers, one uses a list of elements separated by a comma `CM`. These elements can be either an integer `IL` or an integer range. An integer range consists of two integers separated by a `TP` separator. The EBNF form of the right-hand side of the production rule for the set of integers mentioned above is:

Ⓐ `IL CM IL* | (IL TP IL)*` Ⓑ `(IL | IL TP IL) (CM (IL | IL TP IL))*`
Ⓒ `IL CM (IL TP IL)* | (IL TP IL) CM IL*` Ⓓ `IL CM (IL | IL TP IL)* | CM IL+`

Question 32. [L.O.1.2] Given a set of production rules as follows:
$S \rightarrow aSb|T$
$T \rightarrow cTd|\epsilon$
A leftmost derivation is:

Ⓐ $S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aacTdbb \Rightarrow aaccTddbb \Rightarrow aaccddbb$ Ⓑ $S \Rightarrow aSb \Rightarrow aTb \Rightarrow acTdb \Rightarrow ac\epsilon db$
Ⓒ All other options are correct. Ⓓ $S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aaaSbbb \Rightarrow aaaaSbbbb \Rightarrow aaaaTbbbb \Rightarrow aaaabbbb$

Question 33. [L.O.2.1] Given the declarations in an object-oriented programming language with static type checking:

```
class A { def foo() = print("a") }
class B extends A {  } // B is a subclass of A
class C extends A { override def foo() = print("c") } // C is a subclass of A
class D extends B { override def foo() = print("d") } // D is a subclass of B
```

Knowing that variable `b` is declared as type `B` and is referencing some object. Some statements are made about the result when calling `b.foo()`.

(a) `c` (if `b` is referencing an object of type `C`)
(b) `d` (if `b` is referencing an object of type `D`)
(c) `a` (if `b` is referencing an object of type `A`)
(d) `a` (if `b` is referencing an object of type `B`)

The number of correct statements is:

Ⓐ 1 Ⓑ 2 Ⓒ 3 Ⓓ 0

Question 34. [L.O.3.1] In the stages of implementing a language using a hybrid approach, which of the following stages takes in a parse tree (or abstract syntax tree) and throws errors related to naming and data type systems?

Ⓐ Static checking Ⓑ Parsing Ⓒ Lexical analysis Ⓓ Intermediate code generation

The following passage applies to questions 35–36:
Suppose a Python program has classes A, B, C, D(A,B), E(C,A), and F(D,E,B). Of these classes, only classes A and C have a declaration for a method named `method1`. Of these classes, only class D has a declaration for a method named `meth` and whose body is a call statement self.method1().

Question 35. [L.O.2.1] The MRO of class F is:

Ⓐ [F, D, E, C, A, B, object] Ⓑ [F, D, A, B, E, C, object] Ⓒ [F, D, A, E, C, B, object] Ⓓ [F, E, C, D, A, B, object]

Question 36. [L.O.2.1] Which class's method1 will be called by F().meth() and D().meth()?

Ⓐ F().meth() calls method1 in class C and D().meth() calls method1 in class A
Ⓑ F.meth() and D().meth() both call method1 in C Ⓒ F.meth() and D().meth() both call method1 in A
Ⓓ F().meth() calls method1 in class A and D().meth() calls method1 in class C

Question 37. [L.O.2.1] Given the following Python code:

```
y = 0
for x in range(5):
    if x == 5: break
    y += 1
else: print(y)
```

After executing the code above,

(A) The value 5 is printed out (B) The value 4 is printed out (C) No value is printed out
(D) The code above causes a syntax error

Question 38. [L.O.2.1] In an object-oriented programming language with static type checking such as Java or Scala, given class A is the parent class of class B, with variables a and b having types A and B, respectively. Given the following two assignment statements:
a = new B(); // statement 1
b = new A(); // statement 2

(A) Statement 1 is correct and statement 2 is incorrect (B) Statement 1 is incorrect and statement 2 is correct
(C) Both statements 1 and 2 are correct (D) Both statements 1 and 2 are incorrect

Question 39. [L.O.2.1] When programming in Python, to directly insert a newline character into a string by pressing the Enter key instead of using the \n escape sequence, the string must be placed within:

(A) a pair of triple double quotes (B) a pair of double quotes (C) a pair of single quotes
(D) must use escape, cannot be entered directly

Question 40. [L.O.2.1] Which of the following statements accurately describes a high-order function?

(A) Is a function that can take a function as an argument
(B) Is a function that always takes a full array of arguments and a function to process
(C) Is a function that uses recursion to iterate over a set of values (D) Is a function that always returns another function

Mã đề thi 2210

ĐÁP ÁN

Question 1. A
Question 2. A
Question 3. A
Question 4. C
Question 5. C
Question 6. A
Question 7. C
Question 8. A
Question 9. C
Question 10. C
Question 11. B
Question 12. C
Question 13. B
Question 14. D
Question 15. C
Question 16. A
Question 17. C
Question 18. A
Question 19.
Question 20. A
Question 21. B
Question 22. D
Question 23. B
Question 24. A
Question 25. C
Question 26. D
Question 27. D
Question 28. A
Question 29. A
Question 30. B
Question 31. B
Question 32. D
Question 33. B
Question 34. A
Question 35. A
Question 36. A
Question 37. A
Question 38. A
Question 39. A
Question 40. A

Lecturer:	(Date)	Approved by:	(Date)
(Signature and Fullname)		(Signature and Fullname)	

HCMC UNIVERSITY OF TECHNOLOGY **FACULTY OF CSE**	**MIDTERM TEST**		Sem. / School year	2	2022-2023
			Date		06-03-2023
	Course name	Principles of Programming Languages			
	Course ID	CO3005			
	Duration	70 minutes	Code		2211

Notes:
- Students are allowed to use one sheet of A4 handwritten reference material and a calculator.
- Students do questions on an answer sheet..
- Each question has either one correct answer or no correct answer.
- If there is no correct answer, students choose option E.
- Students submit the exam paper along with the answer sheet.

Question 1. [L.O.2.1] Given a list containing elements that can be nested within a list, for example:
`nested_lst = [1, 2, [3, 4, [5, 6], 7], 8, [9]]`
The function `flatten` can take the above list as input and return a flattened list such as `[1, 2, 3, 4, 5, 6, 7, 8, 9]`. The body of `flatten` is:

(A) `return reduce(lambda prev, curr: prev + curr), lst, [])`
(B) `return reduce(lambda prev, curr: prev + (flatten(curr) if type(curr) is list else curr), lst, [])`
(C) `return reduce(lambda prev, curr: prev + [curr], lst, [])`
(D) `return reduce(lambda prev, curr: prev + (flatten(curr) if type(curr) is list else [curr]), lst, [])`

Question 2. [L.O.1.2] The right-hand side production rule of `ui_list` describes a list that contains at least one `UI` element, written in BNF form as:

(A) `UI ui_list | UI`
(B) `UI ui_list |`
(C) `UI ui_list`
(D) `ui_list | UI`

The following passage applies to questions 3–4:
Suppose a Python program has classes A, B, C, D(A,B), E(C,A), and F(D,E,B). Of these classes, only classes A and C have a declaration for a method named `method1`. Of these classes, only class D has a declaration for a method named `meth` and whose body is a call statement self.method1().

Question 3. [L.O.2.1] The MRO of class F is:

(A) [F, E, C, D, A, B, object]
(B) [F, D, E, C, A, B, object]
(C) [F, D, A, B, E, C, object]
(D) [F, D, A, E, C, B, object]

Question 4. [L.O.2.1] Which class's method1 will be called by F().meth() and D().meth()?

(A) F().meth() calls method1 in class A and D().meth() calls method1 in class C
(B) F().meth() calls method1 in class C and D().meth() calls method1 in class A
(C) F.meth() and D().meth() both call method1 in C
(D) F.meth() and D().meth() both call method1 in A

Question 5. [L.O.2.1] When programming in Python, to directly insert a newline character into a string by pressing the Enter key instead of using the \n escape sequence, the string must be placed within:

(A) must use escape, cannot be entered directly
(B) a pair of triple double quotes
(C) a pair of double quotes
(D) a pair of single quotes

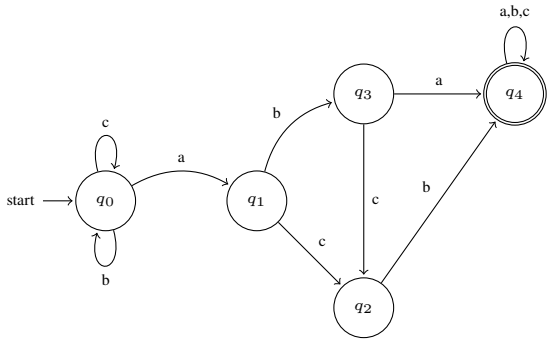
Question 6. [L.O.2.1] In an object-oriented programming language with static type checking such as Java or Scala, given class A is the parent class of class B, with variables a and b having types A and B, respectively. Given the following two assignment statements:
a = new B(); // statement 1
b = new A(); // statement 2

(A) Both statements 1 and 2 are incorrect
(B) Statement 1 is correct and statement 2 is incorrect
(C) Statement 1 is incorrect and statement 2 is correct
(D) Both statements 1 and 2 are correct

Question 7. [L.O.1.1] Choose a regular expression that accepts at least all strings in the MATCH set, but does not accept any strings in the SKIP set:
MATCH ={Cho, chi, Chung, Che, Chan }
SKIP = {Tro, Ching, Chu, Tre, Tran}

Ⓐ (Clc)h(olile)|Ch(alu)n?g? Ⓑ [cCT][hr][aeuio]n?g? Ⓒ [cC]h[aoiue]n?g? Ⓓ [Cc]h[oie]|Ch[au]ng?

Question 8. [L.O.3.1] Given an automaton as follows:



Which of the following input strings is accepted by the automaton above:

Ⓐ bcabcaabcab Ⓑ bccbabcbaab Ⓒ abcaabcbaaa Ⓓ baccbacbcca

Question 9. [L.O.1.1] Given a lexical description defined in ANTLR4 as follows:

```
UNIVERSE: A* S A A A A A+;
fragment A: D | C | S;
fragment D: [0-9];
fragment C: [a-zA-Z];
fragment S: [@$!%*#?&];
```

What is the characteristic of the input string corresponding to the `UNIVERSE` token?

Ⓐ It has at least 6 characters and when containing a lowercase letter it cannot contain an uppercase letter and vice versa
Ⓑ It has at least 4 characters and must contain at least one special character (@$!%*#?&)
Ⓒ It has at least 6 characters and must contain at least one special character (@$!%*#?&)
Ⓓ It has at most 8 characters and must contain a lowercase letter or a digit

Question 10. [L.O.2.1] A program written in language X has the following content:

- Prompt the user to input a value into variable **a** from the keyboard.
- Perform a loop **a** times with the loop body consisting of:
  - Calculate the value of variable **b**.
  - If the value of variable **b** is greater than 5, then print the value of variable **b** and exit the loop.
  - Execute a statement to print **a** with a grammar error.

During execution, sometimes the program runs successfully, while other times it throws errors. What method was used to implement the aforementioned language X?

Ⓐ Hybrid implementation with just-in-time compiler Ⓑ Pure interpreter Ⓒ Compiler
Ⓓ Hybrid implementation

Question 11. [L.O.2.1] Given the following Python code:

```
x,y = 0,4
while x < 3:
    x += 1
else: print(y)
```

After executing the code above,

Ⓐ The code above causes a syntax error Ⓑ The value 4 is printed out Ⓒ The value 3 is printed out
Ⓓ No value is printed out

Question 12. [L.O.1.2] A list of `lcase` consists of consecutive `case` and may be empty. A `case` consists of the keyword `CASE`, followed by an `exp` expression and a `CL` separator, and ending with a list of `stmtlist` statements. The EBNF form of the right-hand side of the `lcase` production rule is:

Ⓐ `CASE exp* CL stmtlist?` Ⓑ `CASE (exp CL stmtlist)*?`
Ⓒ `(CASE exp* CL stmtlist*)?` Ⓓ `(CASE exp CL stmtlist)*`

Question 13. [L.O.1.2] Given the context-free grammar G with the set of terminal symbols as ADD, MINUS, MUL, DIV, LB, RB, set of non-terminal symbols as exp, term, fact, start symbol as exp, and the set of production rules as:
exp → term MINUS exp | term
term → fact DIV term | fact
fact → fact ADD factor | fact MUL factor | factor
factor → LB exp RB | INT
where INT is the token for integers, ADD is the token for addition, MINUS is the token for subtraction, MUL is the token for multiplication, DIV is the token for integer division, LB is the token for left parenthesis, and RB is the token for right parenthesis.
To evaluate the value of the input string, we need to determine the precedence and associativity of the operators and apply the rules of the grammar to generate the parse tree. The value of 100 - 4 / 4 + 2 * 3 / 9 - 10?

Ⓐ 89 Ⓑ 108 Ⓒ 109 Ⓓ 90

Question 14. [L.O.2.1] Given the declarations in an object-oriented programming language with static type checking:

```
class A { def foo() = print("a") }
class B extends A {  } // B is a subclass of A
class C extends A { override def foo() = print("c") } // C is a subclass of A
class D extends B { override def foo() = print("d") } // D is a subclass of B
```

Knowing that variable `b` is declared as type `B` and is referencing some object. Some statements are made about the result when calling `b.foo()`.

(a) `c` (if `b` is referencing an object of type `C`)

(b) `d` (if `b` is referencing an object of type `D`)

(c) `a` (if `b` is referencing an object of type `A`)

(d) `a` (if `b` is referencing an object of type `B`)

The number of correct statements is:

Ⓐ 0 Ⓑ 1 Ⓒ 2 Ⓓ 3

Question 15. [L.O.1.2] Which of the following statements is true about the language generated by the context-free grammar below?
$S \rightarrow AB, A \rightarrow aA|\epsilon, B \rightarrow bB|\epsilon$

Ⓐ This language contains all strings of the form $a^n b^n$, where n is a non-negative integer
Ⓑ This language contains all strings over the alphabet {a, b}
Ⓒ This language contains all strings of the form $a^m b^n$, where m and n are non-negative integers
Ⓓ This language only contains the empty string

Question 16. [L.O.1.2] Consider the following code snippet in ANTLR:

```
decl: ID decl_tail;
decl_tail: CM decl | CL ID CM;
```

Which of the following right-hand sides is appropriate for the `decl` rule to be equivalent to the above code snippet?

Ⓐ `ID (CM ID)* CL ID CM` Ⓑ `(ID CM)* (CL ID CM)?`
Ⓒ `(ID CM)* CL ID CM` Ⓓ `ID CM ID (CL ID)* CM`

Question 17. [L.O.1.1] Choose a regular expression equivalent to the following regular expression: (a|b)*(abb|b)a

Ⓐ [ab]*(ab)+ba Ⓑ (b*a*)*(ab)?ba Ⓒ [a|b]*[abb|b]a Ⓓ [ab]*[ab]?ba

Question 18. [L.O.2.1] Please provide the output of the following code snippet (in Python):

```python
def square(x): return x ** 2
def double(x): return x * 2
numbers = [1, 2, 3, 4, 5]
result = map(square, filter(lambda x: x % 2 == 0, map(double, numbers)))
print(list(result))
```

Ⓐ `[2, 4, 6, 8, 10]` Ⓑ `[4, 16]` Ⓒ `[4, 16, 36, 64, 100]` Ⓓ `[1, 4, 9, 16, 25]`

Question 19. [L.O.1.1] In language X, programmers can write inexact integer constants by replacing unknown digits with the `#` symbol preceded by a digit. An inexact integer constant must start with at least two digits. For example, `12#345` and `123#45#` are valid inexact integer constants, but `123##45` is invalid.
In ANTLR4, assuming the fragments `DIGIT` and `SHARP` describe the digit and `#` characters, respectively, the following description represents inexact integer constants in language X:

Ⓐ `DIGIT (DIGIT* SHARP)*` Ⓑ `DIGIT (DIGIT SHARP?)+` Ⓒ `DIGIT (DIGIT SHARP?)*` Ⓓ `DIGIT (DIGIT SHARP)+`

Question 20. [L.O.1.2] Multiple assignment allows assigning multiple `ID` (on the left-hand side separated by commas) to multiple `exp` (on the right-hand side separated by commas) with the left-hand side and right-hand side separated by an `EQ` sign. Use the `assignment` notation to describe the multiple assignment statement, and write the right-hand side of the `assignment` statement so that the numbers of `ID` and `exp` are equal, with at least one `ID` and one `exp` in the multiple assignment statement:

(A) `ID EQ exp CM assignment | ID EQ exp`
(B) `(ID (CM ID)* EQ exp (CM exp)*`
(C) `ID CM assignment CM exp | ID EQ exp`
(D) `ID CM assignment CM exp | EQ`

Question 21. [L.O.1.1] A Ruby programming language identifier is a string of alphanumeric characters and underscores. It must start with an underscore or a lowercase letter. Choose a regular expression that matches it.

(A) `[0-9_] [a-z0-9_]+`
(B) `[a-z0-9_]+`
(C) `[a-z_A-Z0-9]+`
(D) `[a-z_] [a-z0-9_]*`

Apply the following code snippet in Python for questions 22–23:

```
result = (lst[0] * 2) + func(x, y) - (lst[-1] if lst[1] >= -1.2 else lst[2]) % 5 # cal result
```

Question 22. [L.O.3.1] The number of tokens returned when lexing the above string is:

(A) 45
(B) 38
(C) 43
(D) 40

Question 23. [L.O.3.1] The lexeme string of the 25th token is:

(A) lst
(B) -1
(C)]
(D) if

Question 24. [L.O.2.1] A variable in Python will

(A) need to be declared before assigning a value
(B) hold a reference to an object
(C) receive a value to store
(D) require a type declaration

Question 25. [L.O.1.2] The **`list`** notation describes a list (which may be empty) of `a` elements separated by a `C` symbol. The grammar for **`list`** is written in ANTLR as follows:

```
list: elist | (1);
elist: a C elist | (2);
```

The empty sequence denoted by the underscore symbol (_). (1) and (2) should be.

(A) _ and `C`
(B) _ and `a`
(C) `C a` and _
(D) _ and _

Question 26. [L.O.1.2] Please indicate which of the following grammars is ambiguous:

(A) $S \rightarrow \epsilon|a|b|aSa|bSb$
(B) $S \rightarrow \epsilon|aSbS$
(C) $S \rightarrow \epsilon|aSa|bSb$
(D) $S \rightarrow AB|BA, A \rightarrow \epsilon|aA, B \rightarrow \epsilon|bB$

Question 27. [L.O.2.1] Given the following Python code:

```
y = 0
for x in range(5):
    if x == 5: break
    y += 1
else: print(y)
```

After executing the code above,

(A) The code above causes a syntax error
(B) The value 5 is printed out
(C) The value 4 is printed out
(D) No value is printed out

Question 28. [L.O.1.2] A context-free grammar may include:

(A) Only one production rule that shows the order of tokens.
(B) Regular expressions for describing non-terminal symbols.
(C) A set of terminal symbols, a set of production rules.
(D) A single non-terminal symbol and a regular expression describing it

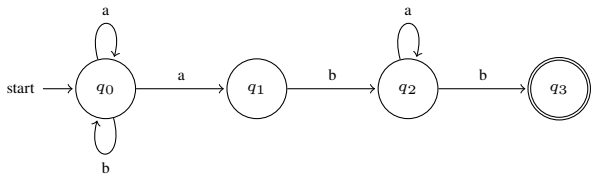
Question 29. [L.O.2.1] Given the following Python code:

```
class A:
    def who_am_i(self):
        print("A")
class B(A):
    def who_am_i(self):
        print("B")
        super().who_am_i()
class C(A):
    def who_am_i(self):
        print("C")
        super().who_am_i()
class D(B, C):
    def who_am_i(self):
        print("D")
        super().who_am_i()
class E(C, B):
    def who_am_i(self):
        print("E")
        super().who_am_i()
class F(E, D):
    def who_am_i(self):
        print("F")
        super().who_am_i()
f = F()
f.who_am_i()
```

The output of the above code is:

(A) FBCDEA (B) FDBCEA (C) FBDCEA (D) FDBCAE

Question 30. [L.O.3.1] Given an automaton as follows:



The equivalent regular expression of the above automaton is:

(A) [ab]*aba*b (B) b*a*aba+b (C) (ab)*aba*b (D) a*b*aba*b

Question 31. [L.O.2.1] Which of the following statements about Method Resolution Order (MRO) in Python is correct?

(A) MRO determines the order in which the Python compiler compiles source code.
(B) MRO determines the order of imported modules in a Python file.
(C) MRO determines the order of executing a Python script.
(D) MRO determines the order of searching for a method or attribute in a class inheritance hierarchy.

Question 32. [L.O.2.1] According to the convention in the Python language, a `protected` attribute named `ex` should be declared by:

(A) Declaring `ex` with an annotation `@private_attr()`
(B) Naming the attribute as `_ex`
(C) Naming the attribute as `__ex`
(D) Declaring `ex: private` in the `init` method

Question 33. [L.O.2.1] Which of the following statements accurately describes a high-order function?

(A) Is a function that always returns another function
(B) Is a function that can take a function as an argument
(C) Is a function that always takes a full array of arguments and a function to process
(D) Is a function that uses recursion to iterate over a set of values

Question 34. [L.O.1.1] Given the regular expression `a[^abc]*c` and the input strings adc, abbc, ayyyyyyyyyc, abc, aabc, axc. The number of input strings that satisfy the regular expression is

(A) 2 (B) 1 (C) 5 (D) 3

Question 35. [L.O.2.1] In the following Python code:

```
def square(f):
    def wrap(x):
        return f(x)**2
    return wrap
(1)
def double(x):
    return x * 2
print(double(3)) # result is 36
```

What should be filled in at (1) so that the result of print(double(3)) is 36?

Ⓐ #square Ⓑ @square Ⓒ square Ⓓ x = square(6)

Question 36. [L.O.3.1] In the stages of implementing a language using a hybrid approach, which of the following stages takes in a parse tree (or abstract syntax tree) and throws errors related to naming and data type systems?

Ⓐ Intermediate code generation Ⓑ Static checking Ⓒ Parsing Ⓓ Lexical analysis

Question 37. [L.O.1.1] To represent a power form a^b with a and b being positive integers in Latex, we write the base a, followed by the caret symbol, and then the exponent b. If the exponent b has two or more digits, it must be enclosed in curly braces. If b has only one digit, the use of curly braces is optional.
In ANTLR4, assuming the fragments describing the digit character is `DIGIT`, the caret symbol is `HAT`, and the opening and closing curly braces are `LP` and `RP`, respectively. Which of the following descriptions can represent the above power form?

Ⓐ `DIGIT+ HAT (DIGIT | LP DIGIT+ RP)`
Ⓑ `DIGIT+ HAT (DIGIT | LP DIGIT DIGIT+ RP)`
Ⓒ `DIGIT+ HAT (LP (DIGIT | DIGIT+) RP)`
Ⓓ `DIGIT+ HAT DIGIT | LP DIGIT+ RP`

Question 38. [L.O.1.2] Given a set of production rules as follows:
$S \rightarrow aSb|T$
$T \rightarrow cTd|\epsilon$
A leftmost derivation is:

Ⓐ $S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aaaSbbb \Rightarrow aaaaSbbbb \Rightarrow aaaaTbbbb \Rightarrow aaaabbbb$
Ⓑ $S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aacTdbb \Rightarrow aaccTddbb \Rightarrow aaccddbb$
Ⓒ $S \Rightarrow aSb \Rightarrow aTb \Rightarrow acTdb \Rightarrow ac\epsilon db$
Ⓓ All other options are correct.

Question 39. [L.O.1.2] To list a set of integers, one uses a list of elements separated by a comma `CM`. These elements can be either an integer `IL` or an integer range. An integer range consists of two integers separated by a `TP` separator. The EBNF form of the right-hand side of the production rule for the set of integers mentioned above is:

Ⓐ `IL CM (IL | IL TP IL)* | CM IL+`
Ⓑ `IL CM IL* | (IL TP IL)*`
Ⓒ `(IL | IL TP IL) (CM (IL | IL TP IL))*`
Ⓓ `IL CM (IL TP IL)* | (IL TP IL) CM IL*`

Question 40. [L.O.1.1] Given a lexical description defined in ANTLR4 as follows:

```
FLOAT_CONSTANT: DIGIT_SEQUENCE EXPONENT? FLOAT_SUFFIX?;

fragment DIGIT_SEQUENCE: DIGIT+ ('.' DIGIT+)?;
fragment EXPONENT: ('e' | 'E') ('+' | '-')? DIGIT+;
fragment FLOAT_SUFFIX: ('f' | 'F' | 'l' | 'L');
fragment DIGIT: [0-9];
```

Which of the following strings is a correct input string for the `FLOAT_CONSTANT` token and has a correct explanation:

Ⓐ `123.456E+7F`, in which `123.456E+7` is formed by `DIGIT_SEQUENCE`
Ⓑ `0.0001E-2f`, in which `E-2` is formed by `EXPONENT`
Ⓒ `6.02e23L`, in which `e23L` is formed by `EXPONENT`
Ⓓ `0.123_456` and there is no `FLOAT_SUFFIX` component

Mã đề thi 2211

ĐÁP ÁN

Question 1. D
Question 2. A
Question 3. B
Question 4. B
Question 5. B
Question 6. B
Question 7. D
Question 8. B
Question 9. C
Question 10. B
Question 11. B
Question 12. D
Question 13. B
Question 14. C
Question 15. C
Question 16. A
Question 17. B
Question 18. C
Question 19. B
Question 20. C
Question 21. D
Question 22. D
Question 23. D
Question 24. B
Question 25. B
Question 26. D
Question 27. B
Question 28. C
Question 29.
Question 30. A
Question 31. D
Question 32. B
Question 33. B
Question 34. D
Question 35. B
Question 36. B
Question 37. A
Question 38. A
Question 39. C
Question 40. B