1. Who is the author of the *primary* textbook for the class?  
   (a) Sibelius; (b) Sebesta; (c) Sethi; (d) Stansifer; (e) Scott

2. How many students received an ‘F’ from the instructor recently for reasons of academic misconduct?  
   (a) 2; (b) 20; (c) 200; (d) 2,000

3. true / false There will be a short quiz at the beginning of (almost) every class meeting.

4. In a class of 25 computer science seniors, one should expect how many to fail to put their name on the quiz card?  
   (a) 0; (b) 5; (c) 10; (d) 25

5. What is the plural of the word *octpus*?  
   (a) octpuses; (b) octopodes; (c) octopi; (d) octpus

Know the meaning of the words *analogy, pedantic, paradigm, linguistics*, i.a.
Quiz: Fri, 11 Jan 2019

1. *Abstraction* means
   (a) signification of words or forms; (b) the medium of expression;
   (c) human-oriented presentation of data; (d) free from convoluted
   interactions; (e) act of determining essential properties.

2. Arabic is to linguistics as:
   (a) expression::visualization; (b) complexity::abstraction;
   (c) Python::programming languages; (d) gender::sex.

3. Software engineers need math because:
   (a) computer operations come from mathematics;
   (b) visualization of data is mathematical; (c) calculus is essential
   to calculating; (d) software consists of abstract constructs.

4. Sapir-Worf hypothesis
   (a) learning new programming languages is hard; (b) the medium
   constrains thought; (c) “don’t repeat yourself” (DRY); (a) all
   computational models are the same.

5. In the field of programming languages one studies: (a) the
   writings of Guido van Rossum; (b) expressing computation;
   (c) visualizing data; (d) the LAMP stack.
1. true / false Frege contributed to the mathematical foundations of the theory of quantification.

2. Which one of the following is not a computational paradigm: (a) logic programming; (b) imperative; (c) categorical; (d) functional.

3. true / false Objects are a distinguishing characteristic of the computational paradigms.

4. The sky does not have a color because: (a) it contradicts physics; (b) the ancient Greeks drank too much (purple) wine; (c) ancient people couldn’t see the sky because of thick (green) foliage; (d) one cannot paint it.

5. Which one of these notational systems do not violate the “arrow of time”? (a) Thai language script; (b) Peano’s notation; (c) Frege’s notation; (d) Incan quipu.
Quiz: Wed, 16 Jan 2019

1. true / false  Declarative programming emphasizes the “how” over the “why.”

2. true / false  HTML is not a general purpose programming language.

3. true / false  FORTRAN can reasonably be considered the first programming language.

4. true / false  Translation to native code can be done by an interactive system.

5. true / false  Translation to native code can be done after execution begins.

6. true / false  ISO stands for the organization officially known as the International Standards Organization.
Quiz: Fri, 18 Jan 2019

1. Ada
2. APL
3. COBOL
4. C
5. FORTRAN
6. Icon
7. Java
8. LISP
9. Pearl
10. Python

A. IBM, J. Backus
B. Augusta Ada Bryon
C. Dahl and Nygaard
D. Ralph Griswold
E. Kenneth Iverson
F. Guido van Rossum
G. John McCarthy
H. Kernighan and Richie
I. Sun, J. Gosling
J. US DoD, G. Hopper
K. US DoD, J. Icbaih
L. Larry Wall
1. A *lexeme* is a
   (a) letter; (b) token; (c) word; (d) phrase

2. In formal languages, a *symbol* is
   (a) a letter used to designate something (b) hallmark or emblem
   (c) a sign to represent something such as an organization (d) one
   indivisible element of a notational system

3. The perspective the programming language field has on syntax
   can best be described as:
   (a) annoyance; (b) basic implementation; (c) construction;
   (d) description

4. true / false  Formal language theory applies to the lexical
   structure of programming languages, but not to the
   phrase structure.

5. true / false  A formal language is a set of symbols from an
   alphabet.

6. true / false  Language can be studied in three parts: pragmatics,
   syntax, and semiotics.
Quiz: Fri, 25 Jan 2019

What formal languages over the alphabet \{a, b, c, d\} do the following regular expression represent? Choose from the formal languages below. (You may choose a letter any number of times.)

1. \(\emptyset^*\)
2. \((a + b)^*\)
3. \((a^*)^*\)
4. \((a + \emptyset)^*\)
5. \(((a \cdot b) + (c \cdot d))^*\)
6. \(((a \cdot b) + (c \cdot d))^* \cdot c\)
7. \(((a + b) + a^*) \cdot c\)
8. \((a^* + b)^*\)
9. \(((a + b)^* + (a + c)^*)\)

A. \{\}\nB. \{\epsilon\}\nC. \{abcd\}\nD. \{ab, cd\}\nE. \{a, b, aa, ab, ba, bb, \ldots\}\nF. \{\epsilon, a, b, aa, ab, ba, bb, \ldots\}\nG. \{ac, bc, aac, abc, bac, bbc, \ldots\}\nH. \{c, abc, cdc, abc\quad c, cdc, cdabc, ababc, \ldots\}\nI. none of the above
1. true / false Back references can be defined in terms of the primitive regular expressions and, so, are just “macros” or “syntactic sugar.”

2. true / false Regular expressions are great because they are more expressive than other common formalisms.

3. true / false Scanner generators and parser generators are examples of a kind of programs which enable programmers to describe what they want and not how to implement it.

4. true / false Syntax diagrams are equivalent to context-free grammars.

5. If \d is the regular expression for a digit ([0-9]), then the regular expression \d+ \d+? matches what part of 321 3451324?
   (a) 3 3; (b) 3 3451324; (c) 321 3; (d) 321 3451324.

6. What part (if any) of Ab+?cAbcd does the Java regular expression A.+? match?
   (a) no part; (b) Ab; (c) Ab+?; (d) Ab+?cAbcd.
1. When is the RSA project due?

2. Show that the following grammar with non-terminals $S$, $A$, and $I$ is ambiguous:

$$
S \rightarrow A \\
A \rightarrow A \times A \mid I \\
I \rightarrow a \mid b \mid c
$$
Quiz: Fri, 1 Feb 2019

1. true / false  The Cherokee script is used in writing FORTRAN programs.

2. true / false  Assertions in programs help debugging.

3. What do regular expressions denote?

4. Complete the last two columns of the following truth table.

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
<td>C</td>
<td>A&amp;B</td>
<td>A&amp;B ⇒ C</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td></td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td></td>
<td></td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td></td>
<td></td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>F</td>
<td>F</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
1. true / false  An inference rule is a method of asserting the truth of one assertion on the basis of the form of other assertions.

2. true / false  Partial correctness means the program satisfies some of the postconditions.

3. true / false  The state of a computer can be modeled as function from identifiers to numbers.

4. true / false  In axiomatic semantic we model the boundedness of integers.

5. true / false  A formula of first-order logic can be used to characterize a set of computer states.

6. true / false  “Sue me if my postcondition is false, sue you if my precondition is false.”

7. true / false  Proving that programs terminate is not possible.
1. What does it mean for a Hoare triple \( \{P\} S \{Q\} \) to be true or valid?

2. Suppose that \( \top \) is a formula in first-order logic (FOL) that always is true. What set of program states does it characterize?

3. What two Hoare triples must be true/valid in order to conclude that the following Hoare triple is?

\[
\{ \top \} \text{ if } x > 0 \text{ then } y := x \text{ else } y := -x \{ 0 \leq y \}
\]
Assuming \( x, y, \) and \( z \) are well-formed names/variables, identify which of the things below are well-formed Hoare triples, valid Hoare triples, or neither.

1. not / wff HT / valid \( \{ x > y \} \ x := 7 \ \{ x > y \} \)
2. not / wff HT / valid \( \{ 7 = 7 \} \ x := 7 \ \{ 7 = x \} \)
3. not / wff HT / valid \( \{ 7 \} \ x := 7 \ \{ x = 7 \} \)
4. not / wff HT / valid \( \{ z = 7 \} \ x := 7 \ \{ z = 7 \} \)
5. not / wff HT / valid \( \{ 7 = 7 \} \ x := 7 \ \{ x = 7 \} \)
6. not / wff HT / valid \( \{ 7 + 1 \} \ x := 7 \ \{ x + 1 \} \)
7. not / wff HT / valid \( \{ \top \} \ x := 7, 8 \ \{ x = 8 \} \)
8. not / wff HT / valid \( \{ 7 = 7 + 1 \} \ x := 7 \ \{ x = x + 1 \} \)
9. not / wff HT / valid \( \{ 8 = 8 + 1 \} \ x := x + 1 \ \{ x = 8 \} \)
10. not / wff HT / valid \( \{ x + 1 = 3 \} \ y := 3; \ x := x + 1 \ \{ x = y \} \)
1. true / false Identifiers in Go are case sensitive.
2. true / false Identifiers in Go can be written in Cyrillic.
3. true / false _ is an identifier in Go.
4. true / false You can refer to a value with _ in Go.
5. What is camel notation aka camel case?
6. What is a keyword?
7. What is a reserved word?
8. What is the difference between a keyword and a reserved word?
1. What does the author mean by the *l-value* and the *r-value word* of an identifier?

2. What is a *variable*?
Quiz: Mon, 18 Feb 2019

1. true / false  Late binding is generally more flexible.
2. true / false  In most programming languages, every identifier has an l-value.
3. true / false  In some programming languages, a function may return an l-value.
4. true / false  An identifier is a variable.
5. true / false  An environment is a kind of a function.
6. What is MIDI?
1. Pointers are evil because (a) aliasing; (b) source of serious errors; (c) confusing; (d) no easy to apply rules of correctness; (e) all of the previous.

2. Pointers are tolerated because there is no efficient alternative (a) trees; (b) stacks; (c) doubly-linked lists; (d) cyclic data; (e) goto.
Quiz: Wed, 22 February 2019

1. true / false  
   Conservative garbage collection may create memory leaks.

2. true / false  
   Dynamically allocated objects tend to live a short amount.

3. true / false  
   The primary problem with reference counting is unpredictability.

4. Which of the following is not a part of generational garbage collection. (a) eden; (b) primogeniture; (c) tenured;
1. Define *referentially transparent*.
2. What is the significance of *referentially transparency*?
3. What is “short-circuit” evaluation?
Give an example illustrating that in many programming languages control flow in unnecessarily deterministic.
1. true / false It is possible for a strongly-typed language to have a heterogeneous, dynamically-accessed, composite data type.

2. Represent 0.3 as a floating-point number with a 10-bit mantissa and an exponent with base two.

3. Represent 0.3 as a floating-point number with a 10-bit mantissa and an exponent with base ten.
1. true / false  A pointer to anything makes static type-checking impossible.

2. true / false  An Ada subtype gives the programmer a compile-time guarantee about the behavior of the program.

3. true / false  Pascal was one of the first languages to take strong typing as a design goal.

4. true / false  It is possible to statically type-check heterogeneous composite data types with dynamic access.

5. true / false  Variant records cannot be statically typed.
Quiz: Fri, 15 Mar 2019 (Ides of March)

1. Today is
   (a) the due date of the Ada project; (b) the Global Youth Climate Strike; (c) Friday, the 13th; (d) quinceañera; (e) pi day.

2. true / false In our context, polymorphism means “many values.”

3. true / false Ada uses name equivalence.

4. true / false A characteristic of universal polymorphism is a finite number of possibilities.

5. true / false Parametric universal polymorphism is the basis of object-oriented programming.

6. true / false Implicit coercion is an example of universal polymorphism.

7. true / false “Branding” allows the programmer to make structural equivalence when the language supports name equivalence.
Quiz: Fri, 19 Mar 2019

1. true / false  Parametric universal polymorphism is the basis of object-oriented programming.

2. true / false  Java, C++, and C# use nominal subtyping.
Quiz: Mon, 18 Mar 2019

1. true / false  The record \{a:int\} is a subtype of \{a:int, b:char\}.

2. true / false  A subtype of a function type must be a function type.

3. true / false  The array type operator in Java is covariant.

4. true / false  The List type operator in Java is covariant.

5. true / false  The List type operator in Java is contravariant.

6. true / false  Arrow (function) types are, by their nature, covariant in the domain and contravariant in the range.

7. true / false  The typing rule for arrays in Java is not type safe.

8. Bounded quantification polymorphism is a combination of _______ and _______.
What is the output of the following program? Explain.

declare
    N: Integer;
procedure First is
    begin
        N := 1;
    end First;
procedure Second is
    N: Integer;
    begin
        First;  -- call first
    end Second;
begin
    N := 2;
    Second;  -- call second
    Integer_Text_IO.Put (Item=>N);
end;