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

2. How many tests (excluding the final) will there be?  
   (a) 0; (b) 1; (c) 2; (d) 3; (e) 4

3. If the mean is 47 and the standard deviation 12; what is the z-score of 53? 

4. What is the meaning of the word “concept”?
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 states:
(a) learning new programming languages is hard; (b) the medium
constrains thought; (c) “don’t repeat yourself” (DRY); (d) triskaidekaphobia;
(e) all computational models are all the same.

5. true / false Complexity can be harmful.

6. 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.

7. true / false As used in chapter 1 of the textbook, the term
orthogonality is applied to vector spaces.

8. true / false For the first programming project you may not
work with a partner.

9. true / false The textbook author believes that if those who
choose languages were well informed, better
languages would eventually squeeze out poorer
ones.

10. What do you think of the textbook author’s division of
programming domains into: scientific applications, business
applications, artificial intelligence, systems programming, and
web software?

11. Who is your partner for the first project?
Quiz 3: Mon, 26 Aug 2019

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

2. 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

3. Alfred Tarski (1902–1983) is known for: (a) theory of computation; (b) semantics; (c) finding Frege’s flaw; (d) theory of quantification

4. Who literally set his own mathematical work in moveable type? (a) Alfred Tarski (1902–1983); (b) Bertrand Russell (1872–1970); (c) Giuseppe Peano (1858–1932); (d) Donald Knuth;

5. Who found the contradiction in Gottlob Frege’s work? (a) Alfred Tarski (1902–1983); (b) Bertrand Russell (1872–1970); (c) Giuseppe Peano (1858–1932); (d) Donald Knuth;

6. XXXIII divided by IV is: (a) V; (b) VI; (c) VII; (d) VIII; (e) IX.
Quiz 4: Wed, 24 Jan 2020

1. true / false  ALGOL can reasonably be considered the most influential programming language.

2. true / false  ISO stands for the organization officially known as the International Standards Organization.

3. true / false  Programs that are interpreted run slower than when compiled.

4. true / false  Unlike a compiler, an interpreter stays around for the execution of the program.

5. true / false  The JDK software javac is a compiler.

6. A JIT compiler is a combination of a compiler and which of the following:
   (a) assembler; (b) linker; (c) interpreter; (d) loader; (e) translator

7. true / false  HTML is a programming language.

8. Our first programming project is in:
   (a) Go; (b) Rust; (c) Ruby; (d) Kotlin; (e) Lua

9. Who is your partner?
Fill in the blanks with the correct programming language from the following list:

Ada, ALGOL, ALGOL-W, APL, BASIC, Beta, C, C++, C#, CLU, COBOL, Eiffel, Forth, FORTRAN, Icon, Haskell, J, Java, LISP, ML, Mesa, Modula-2, Modula-3, Oberon, Pascal, PL/I, PROLOG, Python, Simula, SmallTalk, SNOBOL, Unicon
1. Surprisingly the programming language **APL** is used today in financial applications. It was originally designed in the 1960’s to describe hardware and relies heavily on arrays and vectors. It has a modern successor called **J**. Kenneth Iverson won the Turing Award (1979) for designing **APL** while working for IBM in the 1960s. It has modern successor is named **J**.

2. Like FORTRAN, **PL/I** was developed as an IBM product. It is an early attempt to design a language for all application areas, and includes concurrently executing tasks, exceptions, and pointers.

3. Back in the 1960’s, the language that introduced the notion of class and object was **SmallTalk**. Its modern successor is **C++**.

4. An early string-oriented programming language, **SHL**, was developed in the early 1960s by Ralph Griswold at Bell Labs. Its Pascal-like successor is **Pascal**. And its successor is **Pascal** which incorporates Unicode strings.
Quiz 6: Wed, 29 Jan 2020

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.
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. \{\varepsilon\}\nC. \{ab, cd\}\nD. \{\varepsilon, a, aa, aaa, \ldots\}\nE. \{\varepsilon, aa, aaaa, \ldots\}\nF. \{a, aa, aaa, \ldots\}\nG. \{a, b, aa, ab, ba, bb, \ldots\}\nH. \{\varepsilon, a, b, aa, ab, ba, bb, \ldots\}\nI. \{ac, bc, aac, abc, bac, bbc, \ldots\}\nJ. none of the above
1. true / false  An ambiguous grammar and an unambiguous grammar may both describe the same language.

2. true / false  There are tools in wide-spread use to generate parsers automatically from their descriptions.

3. true / false  There are tools in wide-spread use to generate scanners automatically from their descriptions.

4. true / false  Regular expressions describe more formal languages than BNF definitions.

5. true / false  In most programming languages the "dangling" else belongs to the outermost if statement.
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  The state of a computer can be modeled as function from identifiers to numbers.

3. true / false  A function from identifiers to numbers is a model of first-order logic.

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  All Hoare triples are valid.

7. true / false  The set of states characterized by the formulas $\forall x. x = 3$ is neither empty nor all of the states.
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 \[ \{ 7 = 7 \} \ x = 7 \ { x := 7 } \]
4. not / wff HT / valid \[ \{ 7 \} \ x := 7 \ { x = 7 } \]
5. not / wff HT / valid \[ \{ z = 7 \} \ x := 7 \ { z = 7 } \]
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 \[ \{ x + 1 = 3 \} \ y := 3 ; \ x := x + 1 \ { x = y } \]
10. not / wff HT / valid \[ \{ \exists x \ x > 0 \} \ y := 3 \ { x > 0 } \]