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 50 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) 50

5. What does the common abbreviation IDE stand for (as used in Chapter 1 of the textbook)?
Quiz: Fri, 25 Aug 2017

1. Which of the several paths through the textbook by Scott will we be following?
   (a) plan “F”   (b) plan "R”   (c) plan "P”   (d) plan "C”

2. 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.

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

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

5. Which one of the following is not a computational paradigm:
   (a) logic programming; (b) imperative; (c) categorical; (d) functional.
1. The origin of the word “zero” comes from
   (a) Arabic; (b) Babylonian; (c) Catalan; (d) Dutch
2. The American name of the Greek letter $\tau$ rhymes with
   (a) adieu; (b) bureau; (c) Corfu; (d) [Charles] Dow
3. Which one of the following is not a scripting language:
   (a) Bash; (b) Perl; (c) Whitespace; (d) Python
4. In what language was the operating system Unix written?
   (a) Apl; (b) B; (c) C; (d) D
Quiz: Wed, 30 Aug 2017
Canceled

1. true / false  Declarative programming emphasizes the “how” over the “why.”
2. true / false  HTML is a programming language.
3. true / false  FORTRAN can reasonably be considered the first programming language.
4. true / false  Unlike a compiler, an interpreter stays around for the execution of the program.
5. true / false  ISO stands for the organization officially known as the International Standards Organization.
6. true / false  Machine languages are oriented toward the hardware and “high-level” languages are oriented toward humans.
Quiz: Wed, 1 Sep 2017

1. true / false The syntax known as “Cambridge prefix” is used for all data and all code in LISP.

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

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

4. true / false Java cannot be translated once and then executed over and over.

5. true / false An interactive language system is always an interpreter.

6. true / false Translation to byte-code is becoming more popular.
Quiz: Wed, 6 Sept 2017

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*

1. Back in the 1960’s, the language that introduced the notion of class and object was [ ]. Its modern successor is [ ].

2. Surprisingly the programming language [ ] is used today in financial applications. It was originally designed in the 1960’s to describe hardware and relies heavily on arrays and [ ].

3. Kenneth Iverson won the Turing Award (1979) for designing [ ] while working at IBM around 1960.

4. Like FORTRAN, [ ] 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.

5. The first high-level programming language was [ ].

6. Bertrand Meyer developed the [ ] programming language which is not type-safe because it violates the law of contravariance.

7. Grace Hopper led the development of [ ] , a programming language for business applications.

8. The latest programming language designed by Niklaus Wirth is [ ] , which is both strongly-typed and object-oriented.

9. An early string-oriented programming language, [ ] , was developed in the early 1960s by Ralph Griswold at Bell Labs. Its Pascal-like successor is [ ] .

10. [ ] was designed at Dartmouth College specifically for instructional use, but has since seen widespread use on micro-processors.

11. [ ] is the quintessential logic programming language.

12. [ ] was the first language to use a recursive data structure, the list.

13. In [ ] programs and data have the same form.
Quiz: Wed, 6 Sep 2017

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

A. IBM, J. Backus
B. Augusta Ada
C. Dahl and Nygaard
D. Ralph Griswold
E. Kenneth Iverson
F. Guido van Rossum
G. John McCarthy
H. Kernighan and Richie
I. Bjarne Stroustrup
J. Sun, J. Gosling
K. US DoD, G. Hopper
L. US DoD, J. Icbaih
M. Larry Wall
Quiz: Fri, 15 Sep 2017

1. What is a formal language?

2. When is the Fortran project due?

3. What word best describes the view of syntax in each of the following courses?

   i. FormaL Languages
      A. implementation
      B. representation

   ii. Progr Languages
      C. expressivity
      D. recursion

   iii. Compiler Constr
      E. description

4. true / false \(((a \cdot b) + (c \cdot d))^* \cdot c\) is a regular expression over the alphabet \(\Sigma = \{a, b, c, d, e\}\).
Quiz: Mon, 18 Sept 2017

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\, cdcdc, \, cdabc, ababc, \ldots\}\nI. none of the above
Quiz: Wed, 20 Sept 2017

1. true / false  It is possible to define what good syntax is for a programming language.

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

3. true / false  Back references can be defined in terms of the primitive regular expressions and, so, are just “macros” or “syntactic sugar.”

4. true / false  An ambiguous grammar and an unambiguous grammar may both describe the same language.

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

6. 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.

7. true / false  Lexical analysis determines the phrase structure of a language’s tokens.
Quiz: Fri, 22 Sep 2017

Test on Monday, 25 Sept 2017!
Syntax and Semantics

1. true / false A formula of first-order logic can be used to characterize a set of computer states.
2. true / false “Sue me if my postcondition is false, sue you if my precondition is false.”
3. true / false The Cherokee script is used in writing FORTRAN programs.
4. true / false Partial correctness means the program satisfies some of the postconditions.
5. 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, 22 Sep 2017

*Test on Monday, 25 Sept 2017!*

Syntax and Semantics

Fill in the box with the phrase that best describes the approach of each of the following types of semantics:

<table>
<thead>
<tr>
<th></th>
<th>denotational</th>
<th>is defined in terms of . . .</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>A rules for evaluation</td>
</tr>
<tr>
<td></td>
<td>B Post systems</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>operational</td>
<td>C rules relating states</td>
</tr>
<tr>
<td></td>
<td></td>
<td>D attribute grammars</td>
</tr>
<tr>
<td>3</td>
<td>natural</td>
<td>E mathematical objects</td>
</tr>
<tr>
<td></td>
<td></td>
<td>F “small-step” transitions</td>
</tr>
<tr>
<td>4</td>
<td>structural</td>
<td>G an abstract machine</td>
</tr>
<tr>
<td>5</td>
<td>axiomatic</td>
<td></td>
</tr>
</tbody>
</table>
Quiz: Fri, 29 Sep 2017

1. true / false  Names allows programmers to refer to high-level abstractions.
2. true / false  Late binding in generally more flexible.
3. true / false  The *heap* is a region of storage used for local variables.
4. true / false  If an object is deallocated too soon a program may have a memory leak.
5. true / false  The whole-block scope rule is the same as the declaration-before-use rule.
6. true / false  Using pointers may result in aliasing.
7. true / false  Aliases can be created in C and C++ with the union type.
8. true / false  An environment is a kind of a function.
1. true / false A binding to an object that is no longer live is called *dangling reference*.

2. Which one of the following is *not* a principle storage management mechanism: (a) static; (b) stack; (c) instance; (d) heap

3. true / false Elaboration time is subsumed (it occurs during) run time.

4. Which one of the following does *not* have explicit deallocation: (a) C; (b) C++; (c) Java; (d) Rust
1. true / false  In some languages, for instance Modula-3 and C#, every use of an enumeration constant must be prefixed with a type name.

2. true / false  Sophisticated alias analysis algorithms have allowed C compilers to rival Fortran compilers in the speed of generate code.

3. true / false  In the buddy system the storage management algorithm maintains separate free lists for block of different sizes.

4. true / false  Internal fragmentation is waste within a block, and external fragmentation is non-contiguous free blocks.

5. true / false  In some languages, for instance Pascal, constants are required to have a value than can be determined a compile time.

6. true / false  In C# compile-time and elaboration time constants are distinguished by using the `const` and `readonly` keywords, respectively. Java’s keyword `final` is like C# keyword `const`.

7. true / false  The variable Kotlin does not have to declared before it is used in Fortran.
Quiz: Fri, 6 Oct 2017

1. [ ] mark-sweep
2. [ ] stop and copy
3. [ ] generational
4. [ ] conservative
5. [ ] reference counting

A. Eden, tenured, ...
B. incremental bookkeeping
C. polymorphism
D. if it looks like a pointer, ...
E. theoretical framework
F. follow everything
G. keep half in reserve
Quiz: Wed, 11 Oct 2017

1. What is the difference between a *statement* and an *expression*?
2. How do you parse $a + b \times c \times d \times e / f$ in Fortran? (Put in the missing parentheses.)
3. What error do you get in Ada with the previous expression?
4. true / false Referentially transparency is a property of *language*.
5. true / false Any binary file can be interpreted as a US-ASCII text file.

Questions to think about:

1. What does it mean for a language to be expression-oriented?
2. What distinguishes *operators* from other sorts of functions?
3. Define *orthogonality* in the context of programming language design.
Quiz: Fri, 13 Oct 2017

1. true / false  A subprocedure that takes another subprocedure as an argument is said to be higher-order.

2. true / false  An associative array is dynamically-accessed, homogeneous, composite data structure.

3. true / false  A language with associative arrays can be strongly types.

4. true / false  Defining the right type system is one of the most important research areas in programming language design.

5. true / false  Java and C++ have evolved toward the goal of type completeness.

6. true / false  paraskevidekatriaphobia

Questions to think about:

1. What is the difference between discrete and scalar types?
2. What are aggregates?
3. Can a language be strongly typed and dynamically typed?
1. true / false  If the program misinterprets that bits, it is called a *insecurity*.

2. true / false  Universal reference types (or untyped points) as in PL/I cannot be (statically) type checked.

3. true / false  Variant records cannot be (statically) type checked.

4. true / false  A change of type that does not alter the underlying bits is called a *nonconverting type cast* or *type pun*.

5. true / false  The typing rule in Java for arrays is the same as that for ArrayList.

6. true / false  If a program does not (statically) type check, then the program misinterprets the bits.

Questions to think about:

1. When are two types the same?
Quiz: Wed, 18 Oct 2017

1. true / false A type alias is created when a non-generative type declaration is used.

2. true / false The type declaration with the keyword `type` in Ada is a non-generative type declaration.

3. true / false A derived type (with the keyword `new`, in Ada is a generative type declaration which copies the operations.

4. true / false “Branding” as in Modula-3 allows the programmer to make structural equivalence when the language supports name equivalence.

5. true / false Ada uses name equivalence.

6. true / false C uses name equivalence.

7. true / false Modula-3 uses name equivalence.

8. true / false Java uses name equivalence.

Questions to think about:

1. What is duck typing?
2. What is nominal subtyping?
Quiz: Fri, 20 Oct 2017

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

2. true / false Ada has inheritance as in OO languages using tagged record types.

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

4. true / false Nominal subtyping means subtyping based on the programmer’s declaration of the type.

Questions to think about:

1. What is contravariance?

2. What is bounded quantification?

3. Suppose you work for IBM Rational and you are asked to implement a new feature in the IDE for Ada: a check for buffer overruns. What do you say?
1. true / false In our context, polymorphism means “many values.”

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

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

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

5. true / false The array type operator in C# is covariant.

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?

```
n: integer
procedure first
    n:=1
procedure second
    n: integer
    first () -- call first
n :=2
if readInteger() > 0
    second()
else
    first()
writeInteger(n)
```
Consider two separate, independent executions of the following Ada-like program. Assuming that X is passed by value, what are the values of I and A after the call?

PP: declare
-- declare an array of 5 elements
A: array (1..5) of Integer := (1,2,3,4,5);
I: Integer := 1;
procedure P (X: Integer) is
begin
   X := 0; I := 2; X := 6;
end P;
begin
   P (A[I]); -- call P
   -- value of "I", values of "A"?
end PP;

Questions to think about:
1. Write a simple program that distinguishes pass by reference and pass by value result. Explain.
2. Write a simple program that distinguishes call by reference and call by name. Explain.