Quiz 1: Wed, 21 Aug 2019

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 45 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 *octopus*?  
(a) octpuses; (b) octopodes; (c) octopi; (d) octpus

Know the meaning of the words *analogy*, *pedantic*, *paradigm*, *linguistics*, i.a.
Quiz 2: Fri, 23 August 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 states:
   (a) learning new programming languages is hard; (b) the medium
   constrains thought; (c) “don’t repeat yourself” (DRY); (d) all
   computational models are all 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. true / false Recursion is a distinguishing characteristic of the computational paradigms.

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

4. 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

5. Alfred Tarski (1902–1983) is known for: (a) theory of computation; (b) semantics; (c) finding Frege’s flaw; (d) theory of quantification
Quiz 4: Wed, 28 August 2019

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

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 for an abstract machine is becoming more popular.

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

8. true / false  The instructions of a high-level language are executed directly by the hardware.
Quiz 5: Fri, 30 August 2019

1. □ Ada
2. □ APL
3. □ COBOL
4. □ C
5. □ FORTRAN
6. □ Java
7. □ Simula
8. □ SNOBOL

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
Quiz 6: Fri, 6 Sept 2019

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. \{abcd\}\nD. \{ab, cd\}\nE. \{a, b, aa, ab, ba, bb, \ldots\}\nF. \{\varepsilon, a, b, aa, ab, ba, bb, \ldots\}\nG. \{ac, bc, aac, abc, bac, bbc, \ldots\}\nH. \{c, abc, cdc, abcdc, cdcdc, 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. true / false  Only finite formal languages can be described by BNF.

6. true / false  A BNF definition defines a formal language by providing the means to construct all and only those strings in it.
Quiz 9: Friday, 13 Sep 2019

Note: Fall Career Expo September 26 & 27 in the Clemente Center. Note: Registration survey: https://forms.gle/d4Xnhx8nhzxpmmdN9.

1. When is the Go project due?
2. true / false Triskaidekaphobia?
3. What does it mean for a grammar to be ambiguous?
4. Show that the following grammar with non-terminals $S$, $A$, and $l$ is ambiguous:

\[
S ::= A \\
A ::= A \times A \mid l \\
l ::= a \mid b \mid c
\]
Quiz 10: Mon, 15 Sep 2019

Test on Friday, 27 Sept 2019!
Syntax and Semantics

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

1. [ ] denotational
2. [ ] operational
3. [ ] natural
4. [ ] structural
5. [ ] axiomatic

is defined in terms of . . .

A rules for evaluation
B Post systems
C rules relating states
D attribute grammars
E mathematical objects
F “small-step” transitions
G an abstract machine
H a quinceañera
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  \[ \{ 7 = 7 \} x := 7 \{ x = 7 \} \]
7. not / wff HT / valid  \[ \{ 7 + 1 \} x := 7 \{ x + 1 \} \]
8. not / wff HT / valid  \[ \{ \top \} x := 7,8 \{ x = 8 \} \]
9. not / wff HT / valid  \[ \{ 7 = 7 + 1 \} x := 7 \{ x = x + 1 \} \]
10. not / wff HT / valid \[ \{ 8 = 8 + 1 \} x := x + 1 \{ x = 8 \} \]
11. not / wff HT / valid  \[ \{ x + 1 = 3 \} y := 3; \ x := x + 1 \{ x = y \} \]
12. not / wff HT / valid  \[ \{ \exists x \ x > 0 \} y := 3 \{ x > 0 \} \]
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.
1. true / false  Extent is measured in inches and scope in time.
2. true / false  Stack-allocated objects require complex storage management.
3. Which of the following is not a kind of assignment: (a) let; (b) storage; (c) pointer; (d) with.
4. Which one of the following is not a principle storage management mechanism: (a) static; (b) stack; (c) queue; (d) heap.
5. Which one of the following programming languages does not have deallocation of memory by the programmer: (a) C; (b) C++; (c) Java; (d) Pascal.