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

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. Sun, J. Gosling
J. US DoD, G. Hopper
K. US DoD, J. Icibaith
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. \(((a \cdot b) + (c \cdot d))^*\cdot c)\)
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, 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 $I$ is ambiguous:

   \[
   S ::= A \\
   A ::= A \times A | I \\
   I ::= a | b | c
   \]
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
Quiz 11: Wed, 18 Sep 2019

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.
Read

Why Rust?
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.
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.
1. What is an *expression*?
2. Define *referentially transparent*.
3. What is the significance of *referentially transparency*?
4. What is “short-circuit” evaluation?
Quiz 16: Fri, 4 Oct 2019

1. A significant challenge in programming language design is:
   (a) avoiding over specification; (b) base 16 representation of floating-point numbers; (c) computing the correct answers; (d) deleting garbage.

2. A list comprehension is a programming language
   (a) statement (b) expression (c) construct for understanding (d) declaration

3. true / false In Dijkstra’s guarded command language the if fi construction can abort the program.

4. true / false The syntax of switch statment, as in the C programming language, has been a source of many errors.

5. true / false Where applicable, the switch statement can result in more efficient execution than the use of if statements.
1. true / false  Detecting errors in a program by the compiler is valuable.

2. true / false  It is possible for a compiler to reject all programs that go into an infinite loop on some input data.

3. true / false  A type insecurity arises when the data is misinterpreted.

4. true / false  The word “tractable” means practical.

5. true / false  The early detection of errors reduces programming flexibility.

6. true / false  A program that cannot be statically type–checked has a type insecurity.
1. How does union in the C programming language cause a type insecurity?
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  It is possible to statically type-check heterogeneous composite data types with dynamic access.

4. What comes to mind when thinking about the implementation of variant records: (a) Ada; (b) sum; (c) average; (d) max.

5. What kind of language is Rust? (a) algebraic; (b) systems; (c) scripting; (d) functional.
Quiz: Wed, 11 Oct 2017

1. What is the difference between a statement and an expression?

2. How do you parse `a+b*c**d**e/f` in Fortran? (Put in the missing parentheses.)

3. What error do you get in Ada with the previous expression?

4. true / false Referential 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.
1. true / false  Ada uses name equivalence.
2. true / false  Modula-3 uses name equivalence.
3. true / false  Java uses name equivalence.
4. true / false  Rust was developed by people at Google.
5. true / false  One of two major design goals of Rust was to avoid the security exploits caused by the design of C and C++.
6. true / false  Rust permits code that is not type safe.
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  A Rust enumerated type allows each variant to carry a distinct set of data values along with it.

2. true / false  A Rust program will not crash because of dereferencing a null pointer.

3. true / false  A Rust program will never use a heap-allocated value after it has been freed.

4. true / false  Arrays are naturally contravariant.

5. true / false  Function types are covariant in the domain types.
1. true / false  A subtype of a function type must be a function type.

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

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

4. true / false  Rust’s borrow checking is carried out at compile time.

5. true / false  Rust’s borrow checking is inevitably a conservative approximation.

6. Bounded quantification polymorphism is a combination of [ ] and [ ].
What is the output of the following program Algol-like program assuming dynamic scoping and assuming static scoping?

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;