

Quiz Stuff

- Use a full sheet of 8½ x 11" paper. (Half sheet? Half credit!)
- Put your last name and first initial in the far upper left hand corner of the paper, where a staple would hit it. (It helps when sorting quizzes!)

Mitchell, W.

- No need to write out questions.
- Numbering responses may help you avoid overlooking a question; it's ok to go ahead and pre-number your sheet.
- Two minutes; five questions, plus one extra credit question.
- Can everybody see this line?

Quiz 1, January 10, 2018
2 minutes; ½ point/answer; 2½ points total

1. What is the name of any one programming language created before 1970?
2. How many programming languages are there?
(pick one: dozens, hundreds, thousands)
3. Who founded the UA CS department and in what year?
4. Name an area of research for which the UA CS department was recognized worldwide in the 1970s and 1980s.
5. Ideally, what percentage of your classmates will get an "A" in this course?

EC ½ point: What are the two primary aspects of a programming language specification?

Quiz 2, January 22, 2018

2 minutes; $1 + \frac{1}{2} + \frac{1}{2} + \frac{1}{2} + \frac{1}{2} + 1$ points; 4 (corrected; was 3) points total

1. What are the three aspects of an expression that we often consider?
2. What operation in Haskell has higher precedence than any operator?
3. How would the following type be read aloud? `f :: Int -> Bool`
4. Name one characteristic of the functional paradigm.

5. What is an example of (1) a Haskell type (2) a Haskell type class?
6. Tell me something you learned from the flipped lecture on paradigms.

EC ½ point: What does REPL stand for?

Answers

1. *What are the three aspects of an expression that we often consider?*

Value, type, and side-effect (reversed vest: t-se-v)

2. *What operation in Haskell has higher precedence than any operator?*

function call

3. *How would the following type be read aloud? $f :: Int \rightarrow Bool$*

"f has type int to bool"

4. *Name one characteristic of the functional paradigm.*

Functions that are like mathematical functions; no side-effects; functions are values; recursion in place of looping.

5. *What is an example of (1) a Haskell type (2) a Haskell type class?*

Types: Int, Char, Bool

Classes: Num, Fractional, Bounded, Eq, Ord

6. *Tell me something you learned from the flipped lecture on paradigms.*

A paradigm provides a conceptual framework for understanding and solving problems; a paradigm has a world view, vocabulary, and a set of techniques for solving problems.

Quiz 4, February 5, 2018

4 minutes; $\frac{1}{2} + 1 + \frac{1}{2} + \frac{1}{2} + \frac{1}{2} + 1$ points; 4 points total

1. Name a Haskell type and individually write out all values of that type.
2. Write a function \mathbf{f} that takes a three-element list and returns a 2-tuple with the last then first element of the list. **Restriction:** **Your solution may not use any functions!** Example:

```
> f [10, 20, 30]
(30, 10)
```

3. What is the type of the function \mathbf{f} from the previous question?

4. What is the head and the tail of the list [3 , 4 , 5]?
5. What is a very fast operation with a "cons" list? What's a slow one?
6. What's the basic idea of patterns in Haskell?

EC $\frac{1}{2}$ point: What's a case where we must use a guard instead of a pattern?

Answers

1. *Name a Haskell type and individually write out all values of that type.*

Two short answers: (1) `()` : `()`, (2) `Bool`: `True`, `False`

2. *Write a function that takes a three-element list and returns a 2-tuple with the last then first element of the list.*

`f [x, _, z] = (z, x)`

3. *What is the type of the function `f` from the previous question?*

`f :: [b] -> (b, b)`

4. *What is the head and the tail of the list `[3, 4, 5]`?*

`3` and `[4, 5]`

5. *What is a very fast operation with a "cons" list? What's a slow one?*

Fast: "Add" an element at its beginning. Slow: Get its length.

6. *What's the basic idea of patterns in Haskell?*

Bind names to elements of a data structure.

EC ½ point: *What's a case where we must use a guard instead of a pattern?*

A pattern can't match a range of values.

Quiz 5, February 12, 2018

2'30" (2 mins and 30 secs); 1 + 1 + 1 + ½ points; 3½ points total

1. What are the characteristics of a higher order function?
2. In broad terms, what does the `map` function do?
3. In broad terms, what does the `filter` function do?
4. In Haskell, what is a "section". (OK to just show an example of one.)

EC ½ point: Write an expression whose value is bottom.

Answers

1. What are the characteristics of a higher order function?
A function that takes or returns a function.
2. In broad terms, what does the **map** function do?

Applies a function to each element of a list, producing a list of the results

3. In broad terms, what does the `filter` function do?

Selects elements of a list that satisfy a predicate.

4. In Haskell, what is a "section". (Ok to just show an example of one.)

Short answer: `(+1)`

Long answer: A partial application created by enclosing a binary operator and either operand in parentheses.

5. EC: Write an expression whose value is bottom.

`length [1..]`

Quiz 6, February 19, 2018
2' (2 mins); 1 + 1 + 1 points; 3 points total

1. What is the result of the following expression?

```
foldl1 (*) [2, 1, 2]
```

2. The first argument of the function `foldl` is a *folding function*. How many arguments does that folding function take?

3. What's the BIG difference between `foldl1` and `foldl`?

EC ½ point: What is the type of `foldr1`?

Answers

1. What is the result of the following expression?

```
> foldl1 (*) [2,1,2]  
4
```

2. The first argument of the function `foldl` is a *folding function*. How many arguments does that folding function take?

Two—an accumulated value and an element of the list that is `foldl`'s third argument.

That accumulated value is called "state" by some because it represents the state of the computation, but that seems a little imprecise to me because it typically doesn't contain any representation of how much of the list has been processed and thus isn't the full state of the computation.

I've sometimes named that accumulated value parameter `thusFar` because it represents what we've computed thus far in the folding.

3. What's the BIG difference between `foldl1` and `foldl`? `foldl1` (and `foldr1`) can be thought of as reductions—they produce a value of the same type as in the list they're folding. `foldl` (and `foldr`) can fold a list of values into a different type. Examples:

- A list of `Bools` can be folded into a `String`.
- A list of `Ints` can be folded into a `Bool`.
- A list of chickens could be folded into the transportation cost of the chicken feed they're expected to consume in a year.

EC: What is the type of `foldr1`?

:t foldr1 shows

$$\text{foldr1} :: \text{Foldable } t \Rightarrow (a \rightarrow a \rightarrow a)$$

but as slide 135 suggests, we think of that as

$$\text{foldr1} :: (a \rightarrow a \rightarrow a) \rightarrow [a] \rightarrow a$$

Quiz 7, February 26, 2018
3 minutes; ½ point each; 2.5 points total

1. What is the most significant difference between Java and Ruby's string data types?
2. What's the difference between an *applicative* and an *imperative* method?
3. Ruby has a rich set of operations for accessing characters in a string.
Given `s = "testing"`, show two distinctly different

expressions

of the form `S [. . .]` and the value produced by each.

4. Write a Ruby expression whose value is `nil`.
5. Tell me something about Ruby other than evidenced in the questions above and below. (+½ point for a good one!)

EC 1 ½ point: In the terminology of Ruby, what is the "receiver"?

EC 2 ½ point: Within ten days, what day of the year is Ruby's birthday?

Answers

1. What is the most significant difference between Java and Ruby's string data types?

Ruby strings are mutable.

2. What's the difference between an *applicative* and an *imperative* method?

Imperative methods modify the value of the receiver.

3. Ruby has a rich set of operations for accessing characters in a string.

Given `s="testing"`, show two distinctly different expressions

of the form `s[...]` and the value produced by each.

`s[0] => "t", s[0,4] => "test"`

4. Write a Ruby expression whose value is `nil`.

`nil` is good enough. `""[0]` works, too.

5. Tell me something about Ruby other than evidenced in the questions above and below. (+½ point for a good one!)

EC 1 ½ point: In the terminology of Ruby, what is the "receiver"?

The object on which a method is invoked, like `s` in `s.size`.

EC 2 ½ point: Within ten days, what day of the year is Ruby's birthday?

February 24

Quiz 8, March 14, 2018
2'30"; ½ point each; 2.5 points total

1. Name a statically typed language and a dynamically typed language.
2. How can we quickly tell whether a Ruby identifier is a global variable?
3. What's the fundamental characteristic of duck typing?
4. In Ruby's world, what is an iterator?
5. What keyword does an iterator use to invoke a block?

EC ½ point: What's a "sigil"? (OK to just show an example.)

Answers

1. Name a statically typed language and a dynamically typed language.

Java and Ruby

2. How can we quickly tell whether a Ruby identifier is a global variable?

Preceded by a dollar sign: `$x`

3. What's the fundamental characteristic of duck typing?

Writing methods in a style that's unconcerned about the types of parameters.

4. In Ruby's world, what is an iterator?

A method that can invoke a block.

5. What keyword does an iterator use to invoke a block?

`yield`

EC: What's a "sigil"?

A character preceding an identifier that indicates something about the nature of the named object. Example: Ruby global variables are prefixed with a dollar sign.

Quiz 9, March 26, 2018

3 minutes; 1 + 1 + ½ + ½ points; 3 points total

1. What's the length of the shortest and longest strings described by the following RE?

`/(a | bb | [a-z] [1-3]) /`

2. What is the value of each of the following Ruby expressions?

`/@/ =~ "ab@cd"`

`/.@./ =~ "ab@cd"`

3. Write a RE that will match four character strings whose first and last characters are **x** and whose middle two characters are digits.

4. In computer science theory, a language is a set of

_____.

EC ½ point: How many levels does the Chomsky Hierarchy of languages have?

EC ½ point: Tell me something about regular expressions not evidenced above.

Answers

1. What's the length of the shortest and longest strings described by the following RE?

`/(a|bb|[a-z][1-3])/`

Shortest is 1; longest is 2.

2. What is the value of each of the following Ruby expressions?

`/@/ =~ "ab@cd" => 2`

`/.@./ =~ "ab@cd" => 0`

3. Write a RE that will match four character strings whose first and last characters are **x** and whose middle two characters are digits.

`/x\d\d x/`

4. In computer science theory, a language is a set of strings.

EC ½ point: How many levels does the Chomsky Hierarchy of languages have?

4

EC ½ point: Tell me something about regular expressions not evidenced above.

A regular expression specifies a regular language.

Quiz 10, April 2, 2018; 3 4 minutes; 3 points total

1. Write a Ruby class named `Pair`, as follows:
 - A `Pair` is constructed with two values, both assumed to have a `size` method.
 - The `eqsize` method returns true iff the `size` of both values is the same.
 - The `to_s` method returns a string with the two sizes. Ex: `"3 / 4"`.

Examples:

```
>> p = Pair.new("abc", [1,2,3])
```

```
>> p.eqlsize => true
```

```
>> p.to_s => "3 / 3"
```

```
>> Pair.new([], {1=>2}).to_s => "0"
```

EC ½ point: Tell me something about writing classes in Ruby not evidenced above.

EC ½ point: Provide `Pair` with "getters" `a` and `b`, to fetch the two values held.

Answers

1. Write a Ruby class `Pair`.

```
class Pair
  def initialize(a,b)
    @a,@b = a,b
  end

  def eqsize
    @a.size == @b.size
  end

  def to_s
    "#{@a.size} / #{@b.size}"
  end

  attr_reader :a, :b
end
```

EC ½ point: Tell me something about writing classes in Ruby not evidenced above.

Class variables are denoted with a double sigil: @@

EC ½ point: Provide `Pair` with "getters" `a` and `b`, to fetch the two values held.

Included above, via `attr_reader`.

Quiz 11; April 9, 2018; 3'30"; ½ point per answer; 4 points total

1. With Prolog in mind, write an example of each of the following:
 - An atom
 - A structure with two terms
 - A variable
 - A query
2. A Prolog query has one or more _____. (Hint: starts with "g".)
3. What's the command to run SWI Prolog on lectura?
4. What is the output of the following?

```
?- write(3+4).
```

5. Consider the following fragment of interaction with SWI Prolog. If the user types a semicolon, what will happen?

```
?- food(X).  
X = apple
```

EC ½ point: Within ten years, when was Prolog created?

Answers

1. With Prolog in mind, write an example of each of the following:

- An atom

`a`

- A structure with two terms

`a(1, 2)`

`3+4`

- A variable

`A`

- A query

`food(X).`

2. A Prolog query has one or more goals.

3. What's the command to run SWI Prolog on lectura?

`swipl`

4. What is the output of the following?

```
?- write(3+4).
```

Anything that doesn't have 7 will be counted as correct.

5. Consider the following fragment of interaction with SWI Prolog. If the user types a semicolon, what will happen?

```
?- food(X).  
X = apple
```

It will print the next value of **X** for which **food(X)** can be proven.

EC ½ point: Within ten years, when was Prolog created?
1972

Quiz 12; April 18, 2018; 4'; ½ point per answer; 3 points total

1. What are the names of the four ports of the port model for goal evaluation that we've discussed? Hint: Here's the first letter of each: **c**, **e**, **r**, **f** (½ point for all four)
2. We think of *backtracking* as moving through the goals of a query or rule in a particular direction. What is that direction?
3. For each of the following queries, indicate whether they succeed or fail.

?- length([1,2,3],Len).

?- append(A,B,[1,2,3]), writeln(A), fa

?- member(X,[1,2,3]), member(X,[3,4,5])

4. Write a predicate $p(N)$ that prints the numbers between 1 and N inclusive. It always succeeds.

```
?- p(3).
```

```
1
```

```
2
```

```
3
```

```
true.
```

Answers

1. What are the names of the four ports of the port model for goal evaluation that we've discussed?

call, exit, redo, fail

2. We think of *backtracking* as moving through the goals of a query or rule in a particular direction. What is that direction?

right to left

3. For each of the following queries, indicate whether they succeed or fail.

```
?- length([1,2,3],Len).  
Len = 3.
```

```
?- append(A,B,[1,2,3]), writeln(A), fa  
[]  
[1]  
[1,2]  
[1,2,3]  
false.
```

```
?- member(X,[1,2,3]), member(X,[3,4,5])  
X = 3 .
```

4. Write a predicate $p(N)$ that prints the numbers between 1 and N inclusive.

```
p(Max) :- between(1,Max,N), writeln(N)
p(_).
```