Quiz 1, January 15, 2014

2 minutes; ½ pt/answer; 2½ pts total

1. In what decade was the oldest language still in use created?

2. How many programming languages are there? (pick one: dozens, hundreds, thousands)

3. Who founded the UA CS department?

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" for the course?

Quiz 2, January 22, 2014
2 minutes; ½ pt/answer; 2 pts total

1. What are two characteristics of languages that support imperative programming? (Hint: two words is enough!)

2. With **ghci**, after evaluating **2+3** what is **it+it**?

3. What do you type at the **ghci** prompt to see the type of the function **fizz**?

4. Extra credit (½ point): What's the name of the book that **whm** claims popularized the term "paradigm", as used in the phrase "programming paradigm"? (Or, who's the author of the book?)

Quiz 3, January 27, 2014

3 minutes; 1/1/2 points; 4 points total + ½ point extra credit

1. Write a function **f** such that **f x** returns **7**, no matter what the value of **x** is.

2. Given a function **g** with type **Char -> (Int -> Bool)**, what is the type of each of the following two expressions?

   ```
   g 'x'
   g 'x' 5
   ```

3. Using guards to handle cases, write a function **roman** such that

   roman 5 returns **'V'**

   roman 10 returns **'X'**

   roman anything else returns **'?'**

4. Extra credit (½ point): What's the type of **roman**?

1. f x = 7

2. > let g char ordval = ord char == ordval

   > :type g
   g :: Char -> Int -> Bool

   > :type g 'x'
   g 'x' :: Int -> Bool

   > :type g 'x' 5
   g 'x' 5 :: Bool

3.
```
roman  n
    | n == 5 = 'V'
    | n == 10 = 'X'
    | otherwise = '?'

> roman 5
'V'
> roman 10
'X'
> roman 3
'?'
> :type roman
roman :: (Eq a, Num a) => a -> Char
```

Quiz 4, Feb 3, 2014
2 minutes; 2 points for taking it

1. Write **sum list**, which computes the sum of the numbers in **list**.

2. Write **member x list**, which returns **True** iff **x** is in **list**.

3. Write **last list**, which returns the last element of **list**. Return **undef** for the empty list.

Quiz 5, Feb 10, 2014
3 minutes; 1+½+1+1+½ points

1. Write **sum list**, which computes the sum of the numbers in **list**.

2. Given **let x:y:z = [(1,2), (3,4)]**, what is the value of **y**?

3. Name any one of the characteristics that would cause a function to be considered to be a higher-order function.

4. Recall **map**:

```
> map length ["just", "testing"]
[4,7]
```

   Write **map**.

5. What is the type of **map**?

6. (½ point EC) Write another function that would be considered to be higher-order.

Quiz 6, March 3, 2014

3 minutes; ½ point each #1-5, 1 point for #6, 3.5 points total

1. What's the Ruby analog for Haskell's **`ghci`**? That is, what program can we use to evaluate Ruby expressions interactively?

2. Cite one of the many significant differences between working with strings in Java and strings in Ruby.

3. Cite another Java vs. Ruby string difference.

4. Given a string **`s`**, write a Ruby expression to create a string **`s2`** containing the first and last characters of **`s`**. Assume **`s.size > 0`**.

5. Assume that **`s`** is all digits, like **`"100"`**. Write an expression to convert the value to an integer, so that **`s.....*2 == 200`**

6. Write a Ruby program that reads lines from standard input, printing lines that are longer than 20 characters. (Ok to be off-by-one or -two!)

Quiz 7, March 10, 2014

3½ minutes; ½ point each; 3.5 points total

1. Briefly describe the essential characteristic of "duck typing".

2. Write a method **f** that returns its argument unless it's called without an argument, in which case it returns **8**.

3. Using the iterator **each**, print all the elements in an array **a**.

4. Write Ruby code to create a **Hash** named **h**.

5. Write Ruby code to store the key/value pair **"X"/10** in **h**.

6. What's the Ruby keyword that an iterator uses to invoke a block?

7. If an array has **N** elements, how many times will an iterator on the array invoke its block?

   (a) **N** times
   (b) **1 <= times <= N**
   (c) **N-1** times
   (d) It might depend on the elements

Quiz 8, March 14, 2014
90 seconds; 1½ points

1. Write a method **ifn(s)** that returns true iff the string **s** is a full name, like "John Q. Smith" and nothing more! Hint: Use anchors!

EC ½ point: Write your answer for #1 so that the middle initial is optional.

Solution: (see next page)

There are lots of details that come to mind but here's a reasonable solution for the problem as stated.

```
def ifn(s)
    !! (s =~ /^[a-z]+ ([a-z]\. )?[a-z]+$/i)
end
```

One of the details is capitalization—should the first letters of all parts be required to be capitals? Can a capital appear in the middle (McIntosh)? The solution above takes a liberal view by using **`/.../i`** to make it case-insensitive.

Another detail: the above doesn't accommodate hyphenated names like Mary B. Smith-Fox or names with multiple whitespace like Jim de Stefano.

In practice there's so much variety in names that expecting a full name to be in any particular format creates more headaches than it's worth.

p.s. I wrote this quiz between 9:53 and 9:59, and now I wish I'd thought it through a little more! :)

Quiz 9, March 31, 2014
3 Minutes; 3½ points

1. In what decade or what country was Prolog created?

2. Write an example of a fact.

3. Syntactically, how are facts distinguished from queries?

4. Given a bunch of clauses like **food(apple)** and **food('Big Mac')**, how can we use Prolog to display the foods?

5. **apple** and **'Big Mac'** are examples of _____.

6. Cite a predicate that was mentioned.

7. Write an example of a structure with two terms.

Quiz 10, April 9, 2014
3 Minutes; 1+1+1+½ points

```
thing(apple, red, yes).
thing(grape, purple, yes).
thing(dirt, brown, no).
```

1. Using **thing/3** above right, write a Prolog query that shows the names of the green foods.

2. Draw the box of the four-port model and label the ports. Be sure to include the arrows for the ports.

3. Consider the query **A = B, B = 3, writeln(C), A = 1.**
   What does it output, if anything?
   Does it succeed or fail? (That is, does it say **true.** or **false.**?)

4. Write a rule **hello/0** that prints "Hello!".

```
?- hello.
Hello!
```

It doesn't matter whether **hello.** succeeds or fails.

Quiz 11, April 18, 2014

3 Minutes; 3 points + **½ point E.C. if no singletons!**

1. Write a predicate **middle(+List, ?Middle)** where **Middle** is **List** with the first and last elements removed.

```
?- middle([a,b,c,d,e],M).
M = [b, c, d] ;
false.

?- middle([1,2],[]).
true.

?- middle([1],M).
false.
```

As a reminder, here's the documentation for append:

```
?- help(append/3).
append(?List1, ?List2, ?List1AndList2)
    List1AndList2 is the concatenation of List1 and List2
```

Usage:

```
?- middle([a,b,c,d,e],M).
M = [b, c, d] ;
false.
```

Solution:

```
middle(L,M) :-
    append([ _ ], Rest, L),
    append(M, [ _ ], Rest).
```

Quiz 12, April 25, 2014
3 Minutes; 3 points

1. What does the following query output?

```
?- member(X, [1,2,3]), writeln(X), X = 2, !,
   member(Y, [a,b,c]), writeln(Y), !, fail.
```

2. Write a query that prints "hello" an endless number of times.

3. What is the output of the following query?

```
?- assert(a(1)), retractall(a(2)), assert(a(2)), a(X),
writeln(X), fail.
```

Answers

Question 1:

```
?- member(X,[1,2,3]), writeln(X), X = 2, !,
   member(Y,[a,b,c]), writeln(Y), !, fail.
1
2
a
false.
```

Question 2:

```
?- repeat, write(hello), fail.
```

Question 3:

```
?- dynamic(a/1).
true.

?- assert(a(1)), retractall(a(2)), assert(a(2)), a(X), writeln(X), fail.
1
2
false.
```

Quiz 13, April 30, 2014
3:30; 4 points (1+1+2)

Regarding the brick laying puzzle...

1. **layrow** uses **getone** instead of **member** to select bricks one at a time. Why wouldn't **member** work just as well as **getone**?

2. The following goal is in **laybricks**. What does **BricksLeft** represent?

   layrow(Bricks, RowLen, BricksLeft, Row),

Regarding the Zebra puzzle...

3. What facts are represented by the following goal? (Ok to write as one sentence.)

   Houses = [_, house(_, zebra, _,_, green) | _]

Answers

1. Along with producing each element in turn, **`getone`** also produces a copy of the list with that element removed.

2. **`BricksLeft`** represents the bricks from **`Bricks`** that were not used to lay **`Row`**.

3. The second house is green and has a zebra.