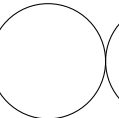
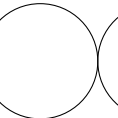
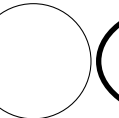
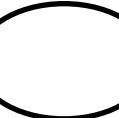


\$Id: cmps112-2014q4-exam3.mm,v 1.115 2014-12-11 13:23:31-08 - - \$

page 1	page 2	page 3	page 4	page 5	Total / 54	<i>Please print clearly :</i>
						Name :
						Login : @ucsc.edu

No books ; No calculator ; No computer ; No email ; No internet ; No notes ; No phone. Neatness counts ! Do your scratch work elsewhere and enter only your final answer into the spaces provided.

1. **Ocaml.** Define the function **reverse** which reverses a list. Use an inner function that is tail recursive. Do not use a higher-order function. **[2✓]**

```
val reverse : 'a list -> 'a list
```

2. **Ocaml.** Define the functions **reverse** and **sum** using **fold_left**. (Fill in the blanks.) **[3✓]**

```
# List.fold_left;;
- : ('a -> 'b -> 'a) -> 'a -> 'b list -> 'a

# let reverse = List.fold_left _____;;
val reverse : 'a list -> 'a list

# let sum = List.fold_left _____;;
val sum : int list -> int

# let length = List.fold_left _____;;
val length : 'a list -> int
```

3. **Ocaml.** The Collatz conjecture states that for any positive integer n , if it is replaced by $n/2$ when even and $3n+1$ when odd, that the sequence eventually converges on 1. Write a function which uses a tail-recursive inner function to return a list of all integers starting from the argument and ending with 1. The inner function produces the list in reverse order, but the result is reversed by the outer function. **[3✓]**

```
# collatz;;
- : int -> int list
# collatz 1;;
- : int list = [1]
# collatz 2;;
- : int list = [2; 1]
# collatz 3;;
- : int list = [3; 10; 5; 16; 8; 4; 2; 1]
```

4. **Scheme.** Write a function **take** in Scheme which will make a copy of the first n elements of a list. If there are fewer than n elements in the list, it returns the complete list. If $n \leq 0$ it returns the empty list. **[2✓]**

```
(define (take n list)
```

5. **Ocaml**, or any functional language. Rules about type checking.
- (i) Every expression has exactly one type.
 - (ii) When an expression is evaluated, exactly one of four general things may happen. List them. [2✓]
- (a)
- (b)
- (c)
- (d)

6. **λ -calculus**. For both applicative and normal order, perform β -reduction on the following expression. [2✓]

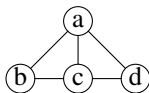
Applicative order	Normal order
$(\lambda x. * x x) (+ 2 3)$	$(\lambda x. * x x) (+ 2 3)$

7. **Ocaml**. Given the function at the left, fill in the table at the right giving the Ocaml types for each item. [2✓]

```
let fac n =
  if n < 0
  then failwith "fac n | n < 0"
  else let rec fac' n' m' =
        if n' = 0
        then m'
        else fac' (n' - 1) (n' * m')
      in fac' n 1
;;
```

fac	
n	
0	
fac'	
n'	
m'	
-	
*	

8. **Prolog**. Given the graph shown here, write **edge** facts to describe it. Write a rule **adjacent** which uses **edge** to determine if two nodes are adjacent. [2✓]



9. **Scheme**. Define the functions **map** and **filter**. [2✓]

```
> (map (lambda (n) (+ 1 n)) '(3 4 5))
(4 5 6)
> (filter (lambda (n) (> n 4)) '(3 4 5 6 7))
(5 6 7)
```

10. **Perl.** Write a script in Perl which will iterate over all of the input files given on the command line and print the contents of all of the files to the standard output. If no files are specified, copy the standard input. You are limited to *one* statement only. Hint: Use <>. [1✓]

```
#!/usr/bin/perl
```

11. **Prolog.** Given facts like the ones presented at the left, define the rules **father** and **mother** where the first argument is the parent and the second argument is the child. [2✓]

```
parents(henry_vii,elizabeth_of_york,henry_viii).
parents(henry_viii,catherine_of_aragon,mary_i).
parents(henry_viii,anne_boylen,elizabeth_i).
parents(henry_viii,jane_seymour,edward_vi).
| ?- father(X,henry_viii).
X = henry_vii
| ?- father(henry_viii,X).
X = mary_i
X = elizabeth_i
X = edward_vi
```

12. Define a function **range** with two integer arguments and which returns a list of all arguments in order including the two arguments. If the first number is larger than the second, return an empty list. [3✓]

(a) **Ocaml.**

```
# range 2 7;;
- : int list = [2; 3; 4; 5; 6; 7]
# range 7 2;;
- : int list = []
```

(b) **Scheme.**

```
> (range 2 7)
(2 3 4 5 6 7)
> (range 7 2)
()
```

(c) **Perl.**

```
print "[@{[range(2,7)]}]\n";
[2 3 4 5 6 7]
print "[@{[range(7,2)]}]\n";
[]
```

13. Write a function that takes two lists as arguments and which returns a single list where each element is a list of corresponding pairs. If the lists are of different lengths, trailing elements of the longer list are ignored.

(a) **Scheme.** [2✓]

```
> (pairthem '(1 2 3 4) '(a b c d e))
((1 a) (2 b) (3 c) (4 d))
> (pairthem '(1 2 3 4 5) '(a b))
((1 a) (2 b))
```

(b) **Ocaml.** [2✓]

```
# pairthem [1;2;3] ['a';'b';'c';'d'];;
- : (int * char) list = [(1, 'a'); (2, 'b'); (3, 'c')]
# pairthem [1;2;3;4;5] ['a';'b'];;
- : (int * char) list = [(1, 'a'); (2, 'b')]
```

Multiple choice. To the *left* of each question, write the letter that indicates your answer. Write **Z** if you don't want to risk a wrong answer. Wrong answers are worth negative points. [12✓]

number of correct answers		$\times 1 =$	$= a$
number of wrong answers		$\times \frac{1}{2} =$	$= b$
number of missing answers		$\times 0 =$	0
column total $c = \max(a - b, 0)$	12		$= c$

- What is **not** to be avoided in a pure functional language?
 - goto
 - lambda
 - loops
 - variables
- Which languages do not have parametric polymorphism?
 - C++
 - Java
 - Ocaml
 - Smalltalk
- What is the expected running time of fold left and fold right on a list of length n ?
 - fold left $O(1)$; fold right $O(1)$
 - fold left $O(1)$; fold right $O(n)$
 - fold left $O(n)$; fold right $O(1)$
 - fold left $O(n)$; fold right $O(n)$
- What is the required amount of stack space for fold left and fold right on a list of length n ?
 - fold left $O(1)$; fold right $O(1)$
 - fold left $O(1)$; fold right $O(n)$
 - fold left $O(n)$; fold right $O(1)$
 - fold left $O(n)$; fold right $O(n)$
- If **guess** is a predicate that searches a database to return one of its elements, and **verify** checks to see if the thing found is good, then we may define the predicate **find**, which returns a valid entry from the database as:
 - find**(X) :- **guess**(X), **verify**(X).
 - find**(X) :- **guess**(X).
find(X) :- **verify**(X).
 - find**(X) :- **verify**(X), **guess**(X).
 - guess**(X) :- **find**(X), **verify**(X).
- What is the type of **List.map**?
 - $('a \rightarrow 'b \rightarrow 'a) \rightarrow 'a \rightarrow 'b \text{ list} \rightarrow 'a$
 - $('a \rightarrow 'b \rightarrow 'b) \rightarrow 'a \text{ list} \rightarrow 'b \rightarrow 'b$
 - $('a \rightarrow 'b) \rightarrow 'a \text{ list} \rightarrow 'b \text{ list}$
 - $('a \rightarrow \text{bool}) \rightarrow 'a \text{ list} \rightarrow 'a \text{ list}$
- In Smalltalk, the expression **3+4** means:
 - The message **+4** is sent to the object 3.
 - The message 3 is sent to the function +, the result of which is a function to which the message 4 is sent.
 - The same as **(+)3 4**.
 - The tuple message **(3,4)** is sent to the object +.
- What is the type of the Ocaml function


```
let g () = 3;;
```

 - int * unit**
 - int -> unit**
 - unit * int**
 - unit -> int**
- Unification is a part of the static type checking algorithm used by compilers of?
 - C and C++
 - Java and Smalltalk
 - ML and Ocaml
 - Perl and Python
- When a garbage collector forms the closure of the root set, it identifies all _____ objects on the heap.
 - dead
 - live
 - reachable
 - unreachable
- A C++ compiler does object-oriented dynamic dispatch via:
 - default memory allocator
 - help-allocated closure
 - inferential type indicator
 - virtual function table
- “Structured Programming with goto Statements”
 - Edsger Dijkstra
 - C.A.R. Hoare
 - Donald Knuth
 - Niklaus Wirth

Multiple choice. To the *left* of each question, write the letter that indicates your answer. Write **Z** if you don't want to risk a wrong answer. Wrong answers are worth negative points. **[12✓]**

number of correct answers		$\times 1 =$	$= a$
number of wrong answers		$\times \frac{1}{2} =$	$= b$
number of missing answers		$\times 0 =$	0
column total $c = \max(a - b, 0)$	12		$= c$

- Which language is not completely strongly typed?
 - C++
 - Perl
 - Scheme
 - Smalltalk
- What is a comment in Ocaml?
 - `(*...*)`
 - `/*...*/`
 - `//...`
 - `;;...`
- What kind of garbage collection fails on cyclic data structures?
 - copying with semispaces
 - generational
 - mark and sweep
 - reference counting
- In Ocaml, what is the type of `(/.)`?
 - `float * float * float`
 - `float * float -> float`
 - `float -> float * float`
 - `float -> float -> float`
- In Java or C++, which statement can cause control to pass from the current function to the calling function, or perhaps the caller of the caller, or perhaps even all the way back to the main function?
 - `break`
 - `continue`
 - `return`
 - `throw`
- A static (access) link is:
 - a pointer to the instruction which called the current function.
 - a pointer to the next free byte of storage on the heap.
 - a pointer to the stack frame in which the current function is nested.
 - a pointer to the stack frame of the caller of the current function.
- Which is a fully curried lazy functional language with overloading?
 - Haskell
 - Ocaml
 - Scheme
 - Smalltalk
- Given the Smalltalk statement `a:=[:x|x+1]`. What expression would return the number 4?
 - `3 to: a.`
 - `a 3.`
 - `a at: 3.`
 - `a value: 3.`
- In Perl, what will print the current date and time?
 - `print "date";`
 - `print 'date';`
 - `print (date);`
 - `print `date`;`
- What is not false in Perl?
 - 0
 - 0.0/0.0
 - `""`
 - `undef`
- In Smalltalk, what is 9?
 - `(4 + 5) value.`
 - `[4 + 5] value.`
 - `"4 + 5" value.`
 - `{4 + 5} value.`
- "Go To Statement Considered Harmful"
 - Edsger Dijkstra
 - C.A.R. Hoare
 - Donald Knuth
 - Niklaus Wirth