$Id: cmps112-2011q2-exam3.mm,v 1.67 2011-06-09 14:06:15-07 - - $

page 1 page 2 page 3 page 4 page 5 Total / 52

Please print clearly :

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. Note : In the interaction samples provided, computer output is shown using Courier font, and user input is shown in **Courier-Bold** font.

1. ***Ocaml :*** Define a function **evenlen** in Ocaml which returns true if the list's length is even and false if not. It must be tail-recursive and may not use the function **List.length** or any of the folding functions. **[2✔]**

```
# evenlen [];;
- : bool = true
# evenlen [1];;
- : bool = false
# evenlen [1;2;3;4];;
- : bool = true
```

2. ***Prolog :*** Define some facts or rules such that the predicate **oddlen/1** succeeds if the length of its list is odd and fails otherwise. Do not do any computation on the length of the list. **[2✔]**

```
| ?- oddlen([]).
no
| ?- oddlen([1]).
yes
| ?- oddlen([1,2,3,4]).
no
```

3. What are the four general things that a function may do when called ? **[2✔]**

4. ***Smalltalk :*** Define a class **List** in which supports the class messages **new** and **cons:with:** ; and the instance messages **car**, **cdr**, and **setcar:setcdr:**. Note that **cons:with:** has to call **setcar:setcdr:** to initialize the instance just created. **[4✔]**

```
st> a := List cons: 3 with: (List cons: 4 with: List new).
a List
st> a car.
3
st> a cdr.
a List
st> a cdr car.
4
st> a cdr cdr.
nil
```

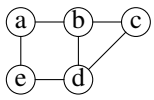
5. ***Haskell :*** Consider the following two Ocaml functions. Define the functions **filter** and **map** in Haskell in terms of list comprehensions. **[2✔]**

```
# filter;;
- : ('a -> bool) -> 'a list -> 'a list = <fun>
# filter ((>)4) [3; 1; 4; 1; 5; 9];;
- : int list = [3; 1; 1]
# map;;
- : ('a -> 'b) -> 'a list -> 'b list = <fun>
# map ((-)8) [3; 1; 4; 1; 5; 9];;
- : int list = [5; 7; 4; 7; 3; -1]
```

6. ***Scheme :*** Using **apply** and **max**, define the Scheme function **depth**. The depth of anything that is not a list is 0. The depth of a list is one more than the maximum depth of its constituent elements. **[2✔]**

```
> (depth '(1 2 (3 4 (5 6)) 88))
3
> (depth '(a b c))
1
> (depth '())
1
> (depth 7)
0
```

7. ***Prolog :*** Given the following graph, Define several facts called **edge** which define the graph. Also, define a rule **adjacent** which can be used to test whether or not two nodes are adjacent to each other. **[2✔]**



8. ***Perl :*** Write a program in Perl that uses **<>** to read all of the input lines. At end of the last file, it prints the number of characters, words, and lines found in the file. A word is any sequence of characters that does not match white space. (A word matches **\S+**). **[2✔]**

```
bash-3.2$ (echo this is a test; \
> echo 2 lines in the file) \
> | wc.perl
2 9 35
```

9. ***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, eventually it converges to the integer 1. Write a function which accepts any integer and returns the number of steps necessary to reach the value of 1. Your solution must be tail-recursive. Do not handle a case where $n < 1$ or is larger than the maximum integer. **[2✔]**

```
# collatz 1;;
- : int = 0
# collatz 2;;
- : int = 1
# collatz 3;;
- : int = 7
# collatz 10;;
- : int = 6
```

10. ***Smalltalk :*** Write some smalltalk code to create a `SortedCollection` and store it in a variable called `sc`. Then use cascaded (chained) messages to add the numbers `23`, `498`, `33`, `87`, and `10`, in that order. Then print out all the numbers, one per line, in sorted order. **[2✔]**

11. ***Scheme :*** Define the function `iota`, which returns a list of all integers from 1 to the argument given. It returns an empty list for an argument less than 1. **[2✔]**

```
> (iota -5)
()
> (iota 0)
()
> (iota 1)
(1)
> (iota 8)
(1 2 3 4 5 6 7 8)
```

12. ***Prolog :*** Define `after/3`, which returns in its third argument all of the elements of the second argument that appear after the first argument. Return `[]` if not found. **[2✔]**

```
| ?- after(3,[1,2,3,4,5,6],X).
X = [4,5,6] ?
yes
| ?- after(0,[1,2,3,4,5,6],X).
X = [] ?
yes
| ?- after(6,[],X).
X = [] ?
yes
```

13. Enter the names of these programming languages in the appropriate box : ***C***, ***C++***, ***Haskell***, ***Java***, ***Ocaml***, ***Perl***, ***Prolog***, ***Scheme***. **[1✔]**

	strong typing	weak typing
static types		
dynamic types		

14. ***Ocaml :*** Define the function `zipwith` whose arguments are a function of two curried arguments, and two lists. The lists must have element types acceptable to the function, and the result is a single list computed by applying the function to pair elements of the lists. Do not compute the lengths of the lists. If the lists are of different lengths, `raise (Invalid_argument "zipwith")`. **[3✔]**

```
# zipwith;;
- : ('a -> 'b -> 'c) -> 'a list -> 'b list -> 'c list = <fun>
# zipwith (+);;
- : int list -> int list -> int list = <fun>
# zipwith (+) [1;3;5] [2;4;6];;
- : int list = [3; 7; 11]
# zipwith (+) [1;3;5] [2];;
Exception: Invalid_argument "zipwith".
```

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. **[11✔]**

number of correct answers		× 1 =	$= a$
number of wrong answers		× ½ =	$= b$
number of missing answers		× 0 =	0
column total $c = \max(a - b, 0)$	11		$= c$

1. In Smalltalk, the expression `3+4` means :
   (A) The message `+` is sent to the number `3`, the result of which is a function that accepts the message `4`.
   (B) The message `+4` is sent to the number `3`.
   (C) The message `3+` is sent to the number `4`.
   (D) The messages `3` and `4` are sent to the operator `+`.

2. In Ocaml, the expression `3+4` means :
   (A) The same as the expression `(3) (+) (4)`.
   (B) The operands `3` and `4` are pushed on a stack, and the operator `+` pops the stack and pushes the sum.
   (C) The operator `+` is applied to the operand `3`, the result of which is a function which is applied to the number `4`.
   (D) The operator `+` is applied to the operands `3` and `4`.

3. If ***M*** = memory leak, ***D*** = dangling references, and ***U*** = unsafe type conversions, which is possible in Java ?
   (A) all of them
   (B) none of them
   (C) ***D*** but neither ***M*** nor ***U***
   (D) ***M*** but neither ***D*** nor ***U***

4. Some early languages, like PL/I, allowed non-local `goto`s, i.e., the ability to use a `goto` to transfer control to a different function. A structured way of doing this in Java is with :
   (A) `break`
   (B) `continue`
   (C) `throw`
   (D) `try`

5. Unification is an important algorithm in determining the flow of control in :
   (A) Ocaml
   (B) Prolog
   (C) Scheme
   (D) Smalltalk

6. A ***closure*** is :
   (A) the address of the local variables that are passed to another function during a function call.
   (B) automatically closing all opened files when the **exit** function is called.
   (C) a heap allocated structure which points at a function and contains the values of all non-local variables used by that function.
   (D) a structure which holds an unevauated expression used when parameters are passed in normal form.

7. A ***thunk*** is :
   (A) the address of the local variables that are passed to another function during a function call.
   (B) automatically closing all opened files when the **exit** function is called.
   (C) a heap allocated structure which points at a function and contains the values of all non-local variables used by that function.
   (D) a structure which holds an unevauated expression used when parameters are passed in normal form.

8. A Perl pattern that matches one or more white space characters is :
   (A) `\d+`
   (B) `\s+`
   (C) `\t+`
   (D) `\w+`

9. In C++, templates such as `stack<int>` are an example of what kind of polymorphism ?
   (A) ad-hoc conversion
   (B) ad-hoc overloading
   (C) universal inclusion
   (D) universal parametric

10. A static link is a pointer to the :
    (A) call instruction that activated the current function.
    (B) segment in an executable binary containing all of a C program's static variables.
    (C) stack frame of the calling function.
    (D) stack frame of the most recent function activation for the function in which the current function is nested.

11. The first computer bug was so named in 1947 at :
    (A) Carnegie-Mellon
    (B) Harvard
    (C) Princeton
    (D) Stanford

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. **[11✔]**

number of correct answers		× 1 =	$= a$
number of wrong answers		× ½ =	$= b$
number of missing answers		× 0 =	0
column total $c = \max(a - b, 0)$	11		$= c$

1. In Ocaml, what is the depth of recursive non-tail calls for the following functions, where n is the length of the list argument ?
   (A) `fold_left` : $O(1)$; `fold_right` : $O(1)$
   (B) `fold_left` : $O(1)$; `fold_right` : $O(n)$
   (C) `fold_left` : $O(n)$; `fold_right` : $O(1)$
   (D) `fold_left` : $O(n)$; `fold_right` : $O(n)$

2. If a balanced binary search treeis implemented in a purely functional language, the insertion of one new node will take how long ?
   (A) $O(1)$
   (B) $O(\log_2 n)$
   (C) $O(n)$
   (D) $O(n \log_2 n)$

3. If `%h` is a hash in Perl, and `$k` is its key, the value is obtained by the expression :
   (A) `$h{$k}`
   (B) `%h{$k}`
   (C) `&h{$k}`
   (D) `@h{$k}`

4. Which expression will cause Scheme to print :
   `(3)`
   (A) `(caar '(1 2 3))`
   (B) `(cadr '(1 2 3))`
   (C) `(cdar '(1 2 3))`
   (D) `(cddr '(1 2 3))`

5. In Prolog, if `guess` is a function that searches a database to return one of its elements, and `verify` checks to see if the selection is valid, then we may define the function `find` to look up a valid entity in the database.
   (A) `find(X) :- guess(X), verify(X).`
   (B) `guess(X) :- verify(X), find(X).`
   (C) `find(X) :- guess(X).`
   `find(X) := verify(X).`
   (D) `verify(X) :- guess(X), find(X).`

6. Ocaml does ***not*** have :
   (A) applicative order evaluation
   (B) operator overloading
   (C) parametric polymorphism
   (D) type inference

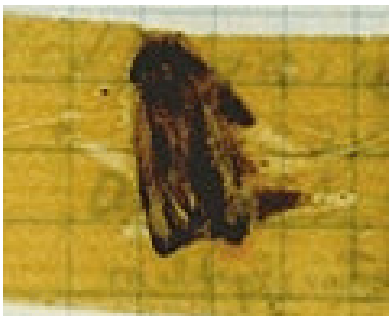
7. In Perl, the name of the script being run is :
   (A) `$!`
   (B) `$0`
   (C) `$ARGV[0]`
   (D) `$ENV{SCRIPT}`

8. What kind of function is
   `let f x y = x + y`
   (A) curried
   (B) thunked
   (C) tupled
   (D) unified

9. In Ocaml, the type of `(+)` is :
   (A) `int * int * int`
   (B) `int * int -> int`
   (C) `int -> int * int`
   (D) `int -> int -> int`

10. In Ocaml, what is the type of `List.map` ?
    (A) `('a -> 'b -> 'a) -> 'a -> 'b list -> 'a`
    (B) `('a -> 'b) -> 'a list -> 'b list`
    (C) `('a -> bool) -> 'a list -> 'a list`
    (D) `('a -> bool) -> 'a list -> bool`

11. In Ocaml, what is the type of `List.fold_left` ?
    (A) `('a -> 'b -> 'a) -> 'a -> 'b list -> 'a`
    (B) `('a -> 'b) -> 'a list -> 'b list`
    (C) `('a -> bool) -> 'a list -> 'a list`
    (D) `('a -> bool) -> 'a list -> bool`



The First "Computer Bug". Moth found trapped between points at Relay #70, Panel F, of the Mark II Aiken Relay Calculator while it was being tested at Harvard University, 9 September 1947. The operators affixed the moth to the computer log, with the entry: "First actual case of bug being found". They put out the word that they had "debugged" the machine, thus introducing the term "debugging a computer program". In 1988, the log, with the moth still taped by the entry, was in the Naval Surface Warfare Center Computer Museum at Dahlgren, Virginia.
[`http://en.wikipedia.org/wiki/File:H96566k.jpg`]