$Id: cmps112-2017q2-midterm.mm,v 1.31 2017-05-04 13:47:18-07 - - $

page 1 page 2 page 3 page 4 Total / 42

Please print clearly :

Name :

Login : @ucsc.edu

No books ; No calculator ; No computer ; No email ; No internet ; No notes ; No phone. Do your scratch work elsewhere and enter only your final answer into the spaces provided. Points will be deducted for messy answers. Unreadable answers will be presumed incorrect.

1. What are the four very general things a function might do when called ? **[2✔]**

2. ***Ocaml.*** Fill in the blanks : **[2✔]**

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

# let sum = _____________________________________________________________________
val sum : int list -> int = <fun>
# sum [1;2;3;4;5];;
- : int = 15

# let length = _____________________________________________________________________
val length : '_a list -> int = <fun>
# length [1;2;3;4;5];;
- : int = 5
```

3. ***Ocaml.*** Define the function `fold_left`.

   (a) ***Ocaml.*** **[2✔]**

   (b) ***Scheme.*** **[2✔]**

4. Define the function `zipwith` that takes a function and two lists and uses the function to join the lists. If the lists are of different lengths, ignore the excess elements of the longer list. **[2✔]**

```
# zipwith;;
- : ('a -> 'b -> 'c) -> 'a list -> 'b list -> 'c list = <fun>
# zipwith (+) [1;2;3] [4;5;6;7];;
- : int list = [5; 7; 9]
```

5. ***Ocaml.*** Define the function **`filter`** which takes a predicate and a list and returns a list in the same order as the original list, but with only those elements for which the predicate is true. **[2✔]**

```
# filter;;
- : ('a -> bool) -> 'a list -> 'a list = <fun>
# filter ((>)0) [1;2;-3;-4;8;-3];;
- : int list = [-3; -4; -3]
```

6. ***Ocaml.*** Rewrite the second filter statement above using a **`fun`** (i.e., lambda) expression instead of `((>)0)`. **[1✔]**

7. ***Smalltalk.*** List the kinds of messages that can be sent to a Smalltalk object, and give an example of each. **[3✔]**

   (a) Highest precedence :

   (b) Middle precedence :

   (c) Lowest precedence :

8. Fill in the following table. Some possible answers : John Backus. Alonzo Church. Grace Hopper. John Kemeny & Thomas Kurtz. John McCarthy. Dennis Ritchie. Bjarne Stroustrup. **[1✔]**

Fortran	Cobol	Basic	Lisp

9. ***Scheme.*** Define the function **`sum`**. You may use the function **`foldl`** or explicitly write a tail-recursive function. **[1✔]**

10. ***Java.*** Give an example of how memory leak might happen in Java. **[2✔]**

11. ***Ocaml.*** Define the function **max** which returns **Some** maximum element of a list as determined by its function argument. Return **None** if the list is empty. **[3✔]**

```
# type 'a opt = None | Some of 'a;;
type 'a opt = None | Some of 'a
# max;;
- : ('a -> 'a -> bool) -> 'a list -> 'a opt = <fun>
# max (>) [3;1;4;1;5;9];;
- : int opt = Some 9
# max (<) [3;1;4;1;5;9];;
- : int opt = Some 1
# max (>) [];;
- : 'a opt = None
```

12. ***Scheme.*** Write a tail-recursive function called **reverse** that reverses a list. Do not use a higher-order function. **[2✔]**

```
> (reverse '(1 2 3 4 5))
(5 4 3 2 1)
```

13. ***Scheme.*** Define the function **map2** which takes three arguments : a binary function and two lists. It returns a list with the two lists merged using the function. If the lists are of different lengths, excess elements of the long list are ignored. **[2✔]**

```
> (map2 + '(1 2 3) '(4 5 6 7))
(5 7 9)
> (map2 * '(1 2 3 4 5) '(6 7 8))
(6 14 24)
```

14. ***Ocaml.*** Define **mul'** as per the project specifications. **[3✔]**

```
# add' [1;2;3] [4;5;6;7] 0;;
- : int list = [5; 7; 9; 7]
# add' [4;5;6;7] [2] 0;;
- : int list = [6; 5; 6; 7]
# add' [9;9;9;9] [1] 0;;
- : int list = [0; 0; 0; 0; 1]
```

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		× 1 =	$= a$
number of wrong answers		× ½ =	$= b$
number of missing answers		× 0 =	0
column total $c = \max(a - b, 0)$	12		$= c$

1. Replacing $(\lambda x\ .\ +\ x\ 1)\ 2$ by $(+\ 2\ 1)$ is an example of
   (A) α-reduction
   (B) β-reduction
   (C) η-reduction
   (D) λ-reduction

2. The λ-calculus was invented by :
   (A) Ada Lovelace
   (B) Alan Turing
   (C) Alonzo Church
   (D) Grace Hopper

3. A garbage collector automatically recycles what kind of objects ?
   (A) dead
   (B) live
   (C) reachable
   (D) unreachable

4. What kind of polymorphism is `vector<string>` an example of ?
   (A) conversion
   (B) inclusion
   (C) overloading
   (D) parametric

5. Two kinds of universal polymorphism are :
   (A) conversion & overloading
   (B) inclusion & parametric
   (C) overloading & inclusion
   (D) parametric & overloading

6. Which of the following functions can be written tail-recursively without the need for reversing the list ?
   (A) filter
   (B) fold_left
   (C) fold_right
   (D) map

7. Edsger Dijkstra published a paper entitled "____ statement considered harmful".
   (A) continue
   (B) goto
   (C) switch
   (D) throw

8. Which was a language designed in the 1950s for use mainly in business data processing ?
   (A) 1957 FORTRAN
   (B) 1958 Algol 58
   (C) 1958 Lisp
   (D) 1959 COBOL

9. The address of a static variable in C and C++ is determined when ?
   (A) compile time
   (B) link time
   (C) exec time
   (D) when a function is called

10. Java generics and C++ templates are an example of ____ polymorphism.
    (A) conversion
    (B) inclusion
    (C) overloading
    (D) parametric

11. Which C operator uses normal order evaluation ?
    (A) `++`
    (B) `<<`
    (C) `==`
    (D) `||`

12. How will Ocaml respond to
    `# (+);;`
    (A) `- : int * int * int = <fun>`
    (B) `- : int * int -> int = <fun>`
    (C) `- : int -> int * int = <fun>`
    (D) `- : int -> int -> int = <fun>`