$Id: cmps112-2014q4-exam2.mm,v 1.96 2014-11-12 15:45:13-08 - - $

page 1 page 2 page 3

Total / 32

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.

1. For each language described here, fill in the name of the language. Choose from among the following languages : Algol 60, AWK, Basic, C++, C, COBOL, FORTRAN, Haskell, Intercal, Java, Lisp, ML, OCaml, Pascal, Perl, PL/I, Prolog, Python, Scheme, Simula 67. (Grading : Deduct 1/2 point for each wrong or missing answer, but do not assign a score less than 0.) **[2✔]**

	Language which uses primarily lazy evaluation, based on the λ-calculus.
	Kemeny and Kurtz designed this language included in the IBM PC ROM in 1980.
	Scripting language designed by Aho, Weinberger, and Kernighan.
	Van Rossum designed this scripting language named after Monty's Flying Circus.
	Steele and Sussman designed this functional language with lexical scoping rules.
Intercal †	Parody language with the **come from**, **maybe**, and **forget** control structures.

† "Abandon All Sanity, Ye Who Enter Here." — **http://catb.org/esr/intercal/**

2. ***Perl.*** Write a program in Perl which will use **<>** to read in lines. At end of file, print the number of characters, words, and lines. A word is any sequence of characters delimited by white space. **[2✔]**

3. ***Smalltalk.*** Extend class **Object** with the message **fibonacci:**, which returns an array filled with the first *n* Fibonacci numbers. **[3✔]**

Expected output.

```
st> a := Object new.
an Object
st> a fibonacci: 10.
(0 1 1 2 3 5 8 13 21 34 )
```

Your answer.

```
Object extend [
   fibonacci: n [
```

4. ***Scheme.*** Define the function **merge** which takes two lists of numbers sorted in ascending order and returns a list of these numbers sorted into ascending order. **[3✔]**

Expected output.

```
>(merge '(1 3 5) '(2 4 6 8))
(1 2 3 4 5 6 8)
>(merge '(0 1 1 2 3) '(1 4 9))
(0 1 1 1 2 3 4 9)
```

Your answer.

```
(define (merge list1 list2)
```

5. ***Scheme.*** Write the function `product` in tail recursive accumulator style so that it computes the product of a list of numbers. **[1✔]**

6. ***Scheme.*** Define the higher-order function `foldl` whose arguments are a binary function, an identity element, and a list in that order. It folds them from left to right into a single value. **[2✔]**

```
(define (foldl fn unit list)




)
```

7. ***Scheme.*** Using `foldl`, define the function `product`, which computes the product of a list of numbers. **[1✔]**

8. ***Scheme.*** Using `foldl`, define the function `length`, which returns the length of a list. **[1✔]**

9. ***Smalltalk.*** Define a block called product so that when given a vector as its `value:` argument, it returns the product of the vector. **[2✔]**

```
product := [




].
product value: #(1 2 3 4 5).
120
```

10. ***Perl.*** Write a line of code that uses `map` to compute the product of an array. **[1✔]**

11. ***Scheme.*** Define the function `reverse`, which reverses a list. Do not use any higher-order functions. Do not use `append`. **[2✔]**

```
(define (reverse list)
```

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. A thunk is :
   (A) A heap allocated value yet to be initialized.
   (B) A pointer to a virtual function table used in dynamic dispatch.
   (C) An unevaluated expression passed into a function which may be evaluated by the function if needed.
   (D) The sound of dead code dropping.

2. What is `2` ?
   (A) `(caar '(1 2 3))`
   (B) `(cadr '(1 2 3))`
   (C) `(cdar '(1 2 3))`
   (D) `(cddr '(1 2 3))`

3. What is `((lambda (x) x) (+ 2 3))` ?
   (A) `(+ 2 3)`
   (B) `+`
   (C) `5`
   (D) `x`

4. What is the parenthesized equivalent of the Smalltalk expression `a b c: d` ?
   (A) `((a b) c: d)`
   (B) `(a (b c: d))`
   (C) `(a b) (c: d)`
   (D) `a ((b c:) d)`

5. In the λ-calculus expression $(\lambda x. + x\ y)$:
   (A) x is bound and y is bound.
   (B) x is bound and y is free.
   (C) x is free and y is bound.
   (D) x is free and y is free.

6. Java supports :
   (A) single inheritance only.
   (B) multiple (mixin) inheritance of functions (methods) but not fields.
   (C) multiple inheritance of functions (methods) and shared inheritance of fields.
   (D) multiple inheritance of functions (methods) and repeated inheritance of fields.

7. What is `((lambda (x) x) '(+ 2 3))` ?
   (A) `(+ 2 3)`
   (B) `+`
   (C) `5`
   (D) `x`

8. Static type checking in C and C++ is done :
   (A) by the preprocessor
   (B) by the compiler
   (C) by the linker
   (D) at run time

9. Which of these functions can be written in a tail-recursive style ?
   (A) `append`
   (B) `fold_right`
   (C) `map`
   (D) `reverse`

10. In Smalltalk, how does one compute $\sqrt{2}$?
    (A) `2 **: 0.5`
    (B) `2 sqrt`
    (C) `Number sqrt: 2`
    (D) `sqrt 2`

11. In Smalltalk and Scheme, type checking is :
    (A) strong and dynamic.
    (B) strong and static.
    (C) weak and dynamic.
    (D) weak and static.

12. Lisp was designed when, by whom, and where ?
    (A) 1953, John Backus, IBM.
    (B) 1958, John McCarthy, MIT.
    (C) 1959, Grace Hopper, *et al.*, Department of Defense.
    (D) 1964, John Kemeny & Thomas Kurtz, Dartmouth College.