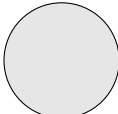
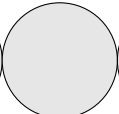
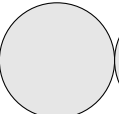
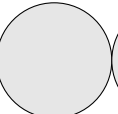
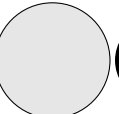



page 1	page 2	page 3	page 4	page 5	Total / 50	Please PRINT using keyboard letters : Name : Login : @ucsc.edu
						

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. Enter the names of the following languages in the appropriate places in the table : [2✓]

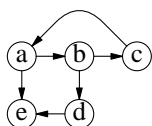
C, C++, Java, Ocaml, Perl, Prolog, Scheme.

	static types	dynamic types
strong types		
weak types		

2. Define the function `zipwith` in Ocaml. It takes a binary function and applies it pairwise to its two argument lists. If the lists are not of the same length, raise the `Invalid_argument` exception. [2✓]

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

3. Write some facts in Prolog to describe the following directed graph. Call the predicate `link`, with the first parameter pointing at the second parameter. [2✓]



4. What is normal order evaluation ? What is applicative order evaluation ? [2✓]

5. Fill in the blanks with an appropriate expression which will generate the results shown. Do not write a definition. A `fun` or `λ`-expression, respectively, is required. [2✓]

(a) Ocaml :

```
# List.map ( _____ ) [1;2;3;4];;
- : int list = [2; 4; 6; 8]
```

(b) Scheme :

```
> (map ( _____ ) '(1 2 3 4))
(2 4 6 8)
```

6. Define a function `split` in Ocaml which takes a single list as an argument and returns a tuple of two lists of the same type each containing half of the elements of the original. If the list has an odd number of elements, one of the output lists will be one element longer than the other. It does not matter which elements go in which output lists. [2✓]

```
# split;;
- : 'a list -> 'a list * 'a list = <fun>
# split [1;2;3;4;5;6];;
- : int list * int list = ([1; 3; 5], [2; 4; 6])
```

7. Define a Scheme function `split` which takes a single list as an argument and returns a list of length two, of which the `car` is a list of half of the elements and the `cadr` is a list of the other half. The actual selection of elements for each output list is arbitrary. [2✓]

```
> (split '(1 2 3 4 5 6 7 8))
((1 3 5 7) (2 4 6 8))
```

8. In either Ocaml or Scheme, using the higher-order functions `map` and `foldl`, and without defining any functions (use `fun` or `lambda`), write two expressions: [2✓]

(a) compute the sum of a list of integers.

(b) add one to each element of a list of integers and return it as a new list.

9. Define the predicate `mem` in Prolog, which succeeds if its first argument can unify with one of the elements of its second argument, which is a list. [2✓]

```
?- mem(3,[1,2,3,4]).
Yes
?- mem(9,[1,2,3,4]).
No
```

10. Define the predicate `sum` in Prolog, which sets its first argument to the sum of the integers in the list of its second argument. [2✓]

```
?- sum(X,[]).
X = 0
?- sum(X,[1,2,3,4]).
X = 10
```

11. In C++, provide the definition of class **box** compatible with the usage shown in **main**. Define the ctor of one parameter, **put**, which deposits a thing in the box, and **get** which copies it out. It must be a template class. Since the functions are trivial, code them all as inline functions. [2✓]

```
int main () {
    box <int> bi (3);
    bi.put (6);
    cout << bi.get () << endl;
    box <string> bs ("foo");
    bs.put ("bar");
    cout << bs.get () << endl;
    return 0;
}
```

12. Write a program in Perl which uses `<>` to read lines of input. Each line is split on white space. Each word thus found is added to the total and at final EOF, the sum of all the words is printed. All words are guaranteed to look to Perl like numbers. [2✓]

13. Write a function in Perl called **search** which takes another function as its first argument, and a pointer to an array as its second argument. It searches the array for the earliest element for which the function returns true and returns that element. If none are found, it returns **undef**. [2✓]

```
% tail -2 search.perl
$a = search sub{$_[0]<0}, [1, 2, -3, -4, 5];
print "$a\n";
% search.perl
-3
```

14. Write a function in Scheme called **grep** whose first argument is a function and whose second argument is a list. The result is a sublist of the argument, in the same order, for which the list elements accepted by the function. Do not use a higher-order function. Use a **let** to avoid calling **car** or **cdr** more than once. [2✓]

```
> (grep (lambda (x) (> x 0)) '(1 2 -3 -4 5))
(1 2 5)
```

15. Draw a picture of the following Scheme list. For each cons cell, draw a box divided into two parts. For any half of such a box: If it is null, write the Greek letter ϕ in it. If it contains an atom, write that atom inside the half box. Otherwise draw an arrow from the half box to the cell that it points at. [2✓]

```
((a b c) (d ((e))) (f . g) h)
```

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		$\times 1 =$	$= a$
number of wrong answers		$\times \frac{1}{2} =$	$= b$
number of missing answers		$\times 0 =$	0
column total $c = \max(a - b, 0)$	11		$= c$

- The Scheme function **map** and the Ocaml function **List.map** take how much time on a list of length n ?
 - $O(1)$
 - $O(\log_2 n)$
 - $O(n)$
 - $O(n \log_2 n)$
- Which statement will cause Ocaml to print
`val f : int -> int -> int = <fun>`
 - `let f (x, y) = x + y;;`
 - `let f = 3, 4, 5;;`
 - `let f x = x, x, x;;`
 - `let f x y = x + y;;`
- What is the type of the Ocaml expression
`List.map (fun x -> x+x) [1;2;3]`
 - `('a -> 'b) -> 'a list -> 'b list`
 - `int -> int`
 - `int list`
 - `int list -> int list`
- Allowing partial parameterization of a function is called
 - currying
 - lambda lifting
 - tupling
 - unification
- What is the time efficiency of

```
let rec f n = match n with
  | 0 -> 0 | 1 -> 1
  | n -> f (n - 1) + f (n - 2)
```

 - $O(n)$
 - $O(n \log_2 n)$
 - $O(n^2)$
 - $O(2^n)$
- The language PL/I allowed a non-local **goto**, that is a **goto** that transferred control from one function into another by allowing statement labels to be passed as parameters. A structured form of non-local **goto** is accomplished in C++ and Java with what statement?
 - break**
 - catch**
 - continue**
 - throw**
- What kind of polymorphism does ANSI C support?
 - coercion
 - inclusion
 - overloading
 - parametric
- What kind of polymorphism is provided by an object-oriented language?
 - coercion
 - inclusion
 - overloading
 - parametric
- Algol 60's parameter passing mechanism by name uses a parameterless procedure called a
 - closure
 - curry
 - horn
 - thunk
- Which of the following higher order functions would be useful in returning a list containing the squares of all the integers in a list passed to it as an argument?
 - compose**
 - foldl**
 - foldr**
 - map**
- The Norwegian lives in the first house.

```
% einstein.pl
```

 The _____ owns the fish.
 - brit
 - dane
 - german
 - swede

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		$\times 1 =$	$= a$
number of wrong answers		$\times \frac{1}{2} =$	$= b$
number of missing answers		$\times 0 =$	0
column total $c = \max(a - b, 0)$	11		$= c$

- Unification is used as part of the static type checking algorithm used in
 - C++
 - Ocaml
 - Prolog
 - Scheme
- In C++, in order to make a member of a class available only to other members of the same class and also to any derived class, but not to any other classes, what keyword is used?
 - friend
 - private
 - protected
 - public
- In order to suppress a C++ constructor with one parameter being used in automatic type conversion, one precedes the definition of that constructor with what keyword?
 - automatic
 - explicit
 - implicit
 - static
- What is the type of


```
let f x y = y x;;
```

 - 'a * 'b -> 'b * 'a
 - 'a -> 'b -> 'c -> 'a
 - 'a -> ('a -> 'b) -> 'b
 - ('a -> 'b) -> 'a -> 'b
- Which of these memory management mechanisms is most friendly to the page tables?
 - copying collector with semispaces
 - explicit use of free
 - mark and sweep in-place collection
 - reference counting

- The Ocaml function `List.filter` is of type `('a -> bool) -> 'a list -> 'a list`. A function to extract positive integers from a list can be defined as
 - let pos = List.filter ((<)0)
 - let pos = List.filter ((>)0)
 - let pos = List.filter (0(<))
 - let pos = List.filter (0(>))
- When an Ocaml function has a type like the following, what kind of polymorphism is thus illustrated?


```
('a -> 'b) -> 'a list -> 'b list
```

 - coercion
 - inclusion
 - overloading
 - parametric
- Which of the following C++ functions will compile without type-checking errors, but return a dangling pointer when called?
 - int p() { int x = 0; return x; }
 - int &p() { int x = 0; return x; }
 - int *p() { int x = 0; return x; }
 - int *p() { int *x = 0; return x; }
- Suppose we have a generic associative container in C++ called `hash`. What would be correct syntax for declaring such with keys as strings and values as integers?
 - hash <string, int> m;
 - hash m <string, int>;
 - hash m of string and int;
 - m: (string * int) hash;
- Given the query `car(L,X)`, what Prolog statement will properly extract the first element of list `L` and assign it to `X`?
 - car([Head | Tail], Tail).
 - car([Head | Tail], _).
 - car([Head | _], Head).
 - car([Head, Tail], Head).

11.

- 2264
- 2462
- 4622
- 6224

