

3. (5 points) State Java's subtyping rules on method overriding in terms of the signatures of overriding and overridden methods.

4. (5 points) Define the following terms of AspectJ:

- Join point
- Pointcut
- Advice

5. (5 points) Define a Haskell list containing all non-negative, even numbers that are also multiples of 3, i.e., 0, 6, 12, 18, 24, ...

```
numbers :: [Int]
```

6. (10 points) Evaluate the following Haskell expressions by showing the main steps of the evaluations.

(a) $(\lambda x \rightarrow (\lambda y \rightarrow (x\ y))) (\lambda x \rightarrow x) (\lambda x \rightarrow x)$

(b) $f (\lambda x \rightarrow (\lambda y \rightarrow y)) [10, 20, 30]$ where

$f\ x1\ [x2] = x2$

$f\ x1\ (h1: h2: t) = f\ x1\ ((x1\ h1\ h2) : t)$

7. (10 points) Consider the following Haskell function named `mystery`.

```
mystery y = foldl1 (+) (map (\x->(x * x)) (filter (\x->(x > 0)) y))
```

- (a) Write the signature of the above function?
- (b) Define the above function directly without using any helper functions such as `foldl1`, `map`, and `filter`.

8. (5 points) There are three general approaches for formally describing the meanings of programming languages. What are they, and which one has the following characteristics?

- based on the recursive function theory
- most abstract semantic description approach

9. (5 points) Define an axiomatic semantics for the following `if` statement of Java, where E is a Boolean expression without side-effects and S is a statement.

`if (E) S`

10. (5 points) Calculate the weakest precondition for the following code:

```
x = x + y;  
y = x - y;  
x = x - y;  
{x > 0}
```

11. (5 points) State the fundamental differences between imperative (or procedural) and logic (or declarative) programming.

12. (10 points) Assume that you are planning a party and want to speculate about who might dance with whom. Assume the following Prolog clauses are already written.

```
boy(X). /* X is a boy. */  
girl(X). /* X is a girl */  
tall(X). /* X is tall. */  
short(X). /* X is short. */  
hobby(X, Y). /* X's hobby is Y. */
```

Define a Prolog clause `possible_pair(X, Y)`. A pair is a *possible pair* if:

- (a) one of the pair is a boy and the other is a girl, and
- (b) both of the pair have the same hobby or the boy is taller than the girl.

13. (5 points) Discuss the relative advantages and disadvantages of pass-by-value-result and pass-by-reference parameter passing mechanisms.
14. (5 points) In Haskell, one can represent an “infinite” data structure, e.g., a list of all positive numbers. To achieve this, which parameter passing mechanism will you use for Haskell? Justify your answer by stating the reason. Remember that there are five different parameter passing mechanisms possible (see Problem 15 below).
15. (15 points) Consider the following Java-like skeletal code:

```
int[] a = new int[] { 3, 2, 1, 0 };

void guessWhat(int x, int y) {
    x = x + 2;
    y = y + 1;
    a[x] = a[y] + 2;
}

int b = 1;
guessWhat(b, a[b]);
```

What are the values of variables **a** and **b**, respectively, after the call to the subprogram **guessWhat** returns when:

- (a) Pass-by-value is used,
- (b) Pass-by-result is used,
- (c) Pass-by-value-result is used,
- (d) Pass-by-reference is used, and
- (e) Pass-by-name is used?

Assume that the language uses 0-based indices for arrays and an uninitialized variable is set to a default value, e.g., 0 for **int** variables. You should explicitly state all other assumptions, if any.

Method	Variable a	Variable b
pass-by-value		
pass-by-result		
pass-by-value-result		
pass-by-reference		
pass-by-name		