Principles of Programming Languages (CS329)
Computer Science and Engineering
Indian Institute of Technology Bombay

Final exam
2003-11-25 09:30–12:30

NAME ______________________ ROLL ______________

This quiz has 6 page/s. You can use the FWH book and your own class notes only. Write your roll number on each sheet. Perform rough work elsewhere and write only the final answer in the spaces provided and on the last blank page, if any. Clarity of writing may influence your scores. Do not attach rough work. Do not write inside the boxes marked for grading. Do not use red or green ink. Even if a question does not require an explanation, providing some explanation may qualify you for partial credit in case your final answer is wrong. Use the marks alongside each question for time management.

1. Using pure lambda expressions without recursion or Y, design a function which takes as input a positive number n in the form of a Church numeral and outputs the smallest positive number k such that $k!$ (the factorial of k) exceeds n (i.e., $k! > n$). Try to find a short and simple solution.

	4

2. In a Scheme-like programming language, the following code

```
(let ((x 2))
  (cons (set! x (1+ x))
        (list (set! x (1+ x)))))
```

evaluates to the list `(3 2)`. Write down standard semantics (using continuations and store) for the construct `(cons` E_1 E_2`)` in this language. Assume that lists are represented in memory by cells holding pairs, where the left item in the pair is a list element and the right item is the location of the next pair.

	5

3. Exactly what will the following Java code print when executed? Explain why in up to three lines.

```
public class Stack {
  static void rec(int x) throws Exception {
    if ( x > 5 ) throw new Exception("exc=" + x);
    rec(x+1);
    System.err.println("out=" + x);
  }
  public static final void main(String args[]) {
    try { rec(1); }
    catch ( Exception pex ) { System.err.println(pex); }
  }
}
```

	3

4. Suppose you have to write programs in a preliminary release of Java which is missing the `finally` construct. In Java, if you write

```
try { Et } finally { Ef }
```

you want `Ef` to execute no matter what, after `Et` completes either normally or abnormally with an exception. Give the simplest (least typing for the programmer) workaround recipe to achieve the effect of `finally` using any other Java features.

	2

5. What will be the result of interpreting the following expressions in Scheme? Give up to three lines of explanations for each case.

(a)
```
(let ((r 1) (top (call-with-current-continuation (lambda (c) c))))
  ( (top (lambda (x) x)) r ) )
```

	2

(b)
```
(let ((r 1) (top (call-with-current-continuation (lambda (c) c))))
  ( (top top) (begin (display r) (newline) (set! r (1+ r)) ) ) )
```

	2

(c) Make minimal modifications to the code above so that it prints integers between 1 and 100 and then terminates. Complete the code shown below and write it out in full separately (do not fill the blanks in place). Some blanks may need to remain empty.

```
( ______
  (let (______ (top (call-with-current-continuation (lambda (c) c))))
    ( ______
      ( (______) (begin (display r) (newline)
                         (set! r (1+ r))) ) ) ) )
```

	3

6. Infer the most specific type of all variables and subexpressions of the following expression:

```
(proc (f g x y) (if (f 3 y) (f x "static") (g x)))
```

	4

7. In class we saw how to reconstruct ($\mathcal{R}$) the type of the expression `(proc I` E`)`.

   (a) Extend $\mathcal{R}$ so that it can reconstruct types for expressions of the form `(rec I` E_p`)`.

	2

   (b) Draw an abstract syntax tree for the following expression and mark type variables on appropriate nodes. Also write down all constraints between type expressions induced by the syntax tree. Pick as few fresh type variables as possible.

```
(rec fib (proc (n)
           (if (< 2 n)
               1
               (+ (call fib (- n 1)) (call fib (- n 2))))))
```

	2

(c) Show important steps during the reconstruction of the type of variables and subexpressions of the above expression.

	2

8. In some language with array support, the following code

```
let
  f = proc(x, y) { y = y+1; x = 10; };
  a = array [ 200, 201, 202 ]; // first index is 0
  i = 0;
begin
  f(a[i], i);     print a;
  // f(i, a[i]);  print i;  print a;
```

prints

```
[ 200, 10, 202 ]
```

(a) Does the language have applicative or normal order evaluation? Justify your answer in two lines.

	2

(b) What would be printed if the last line were to be uncommented?

	2

Total: 35