Programming Languages Lab (CS389) **Assignment 1**
Computer Science and Engineering **Out: 2002-08-06**
Indian Institute of Technology Bombay **Due: 2002-08-20**

For some of these experiments, it may be a good idea to use the `limit` or `ulimit` command in your shell to control the growth of your stack segment.

1. In your Foundations course, you wrote recursive factorial using `lambda` as

```
( (lambda (f x) (f f x))
  (lambda (fact n)
    ( if (= 0 n) 1 (* n (fact fact (- n 1))) )  )
  5 )
```

   Test if this runs correctly using the Scheme interpreter.

2. In this class, we wrote factorial as

```
( (lambda m (m m))
  (lambda fact
    (lambda n
      (IF (= 0 n) 1 (* n ((fact fact) (- n 1))) ) )  )
  5 )
```

   Does the above code execute correctly using Scheme?

3. The functional F for the factorial function was defined as

```
(lambda fact
  (lambda n  (if (= 0 n) 1 (* n (fact (- n 1))))) )
```

   Note that there is no `(fact fact)` in the body, only a single `fact`. We also defined the Y operator such that `(`Y F`)` returns the factorial function. Y was defined in class as

```
( (lambda y (lambda f (f ((y y) f))) )
  (lambda y (lambda f (f ((y y) f))) )  )
```

   (a) Does `((`Y F`) 5)` compute the factorial of 5 correctly using Scheme? Explain why it does or does not.

   (b) If it does not, can you modify the definition of Y (without modifying F) so that the problem is resolved?