CS117 (Fall 2017) **Lab Assignment** **11/27,30/2017**

Practice for functional programming with Scheme

1. Download and install any Scheme interpreter in your computer system.
   One suggested site (MIT/GNU Scheme for both Mac and Windows) is:
   http://www.gnu.org/software/mit-scheme/
   You should follow the installation guide provided at the site.

2. Interactive mode operations

   Invoke the Scheme session: $>./mit-scheme

   Try the following operations:

```
1]=> (+ 2  3)
1]=> '(+ 2  3)
1]=> (car '(1  2  3))
1]=> (cdr '(1  2  3))
1]=> (* (+ 2  3) 7)
```

   Sample function definition and usage:

```
1]=> (define (square x) (* x  x))      ; square function definition
1]=> (square  5)                       ; function application with argument 5
1]=> (map  square  '(1  2  3  4  5))   ; function mapping on input values
```

   Quit the Scheme session: type Crl-D (or, Crl-C and q)

3. Working with source code file

   (1) Create a file and type the Scheme source code, i.e., function definitions;
   Save this file – in MIT Scheme, no file extension is needed;
   (2) Invoke Scheme session: $> ./mit-schme
   (3) Load the Scheme source code file:
   1]=> (load "file_name")
   (4) Type expressions (function applications), e.g.,
   1]=> (square 5) ; assume that the square function is defined in the file
   (5) Quit the Scheme session: type Crl-D (or, Crl-C and q)

4. Implement the following three functions in Scheme, i.e., create a text file containing the code and save it with any meaningful file name (extension is not needed), and run them.

```
Fibo (x)   ; returns the x^th Fibonacci number
           ; ex)  Fibo(0) => 1;  Fibo(1) => 1;  Fibo(2) => 2;  Fibo(3) => 3;  Fibo(4) => 5

Fibo_list (y)   ; returns 0^th ~ y^th Fibonacci numbers
           ; ex)  Fibo_list(0) => 1;  Fibo_list(1) => 1, 1;  Fibo_list(2) => 1, 1, 2;
           ;      Fibo_list(5) => 1, 1, 2, 3, 5, 8

Fibo_list_20 ( )   ; returns the first 20 Fibonacci numbers; i.e. 0^th, 1^st, 2^nd, .. , 20^th
           ; ex)  Fibo_list_20 => 1, 1, 2, 3, 5, 8, 13, ……. 6765, 10946
           ; Use "map" and previously defined function "Fibo".
```

```
; For Step4, you should define the following 3 functions.
; Create a file (for example, named "fibonacci") and type the code and save it.

  (define (fibo x)
    ....
    ....
   )

  (define (fibo_list y)
    ....
    ....
   )

  (define (fibo_list2_20)
    ....
    ....
   )

; after making/saving the program file, invoke the Scheme session:  $>./mit-scheme
; Load the program file first, then call functions that you defined in the file, e.g.,
; 1]=> (load "fibonacci")
; 1]=> (fibo  5)
; 1]=> (fibo_list  5)
; 1]=> (fibo_list_20)
; . . . more testings
```