**CS 352 Spring 2018 -- Programming Assignment #3 (solution for your reference only)**

***Do both problem in both Python and Lisp.***

***Problem 1:*** Create a list of integer values with list size of 20 and values in the range [0, 100] randomly generated, and create another list of float values with list size of 20 and values in the range [10.0, 80.0) (\*correction made here) with values randomly generated.

Python:

from random import \*

>>> [randint(0,101) for num in range(20)]

[10, 33, 100, 76, 100, 14, 82, 49, 60, 33, 57, 40, 59, 32, 70, 15, 23, 14, 63, 7]

>>> [10+random()\*70 for num in range(20)]

[46.263941982389525, 38.66149239495219, 46.23390057005086, 33.88551733543066, 53.99589016981416, 58.14071338803991, 36.53274940246892, 15.641889864838193, 34.057162832396955, 75.35612944588402, 11.191465156183424, 71.93714653298025, 63.06727810905378, 79.35291915279555, 41.77645928515747, 70.24668949196749, 70.30104109014312, 27.721547214024767, 58.41619663101585, 59.20938517565762]

Lisp:

(setf ivalues nil)

(loop for x from 1 to 20

do (setf ivalues (cons (random 100) ivalues)))

(setf fvalues nil)

(loop for x from 1 to 20

do (setf fvalues (cons (+ 10 (random 70.0)) fvalues)))

[2]> (load "test\_random.lsp")

;; Loading file test\_random.lsp ...

(75 8 64 24 2 58 45 36 15 48 29 92 15 2 29 56 83 90 73 60)

(62.906933 60.5113 15.358604 14.579068 63.864727 12.04698 66.89256 25.740803

78.20657 64.6177 79.11749 14.100323 52.682514 10.653228 40.870308 68.909195

27.848398 40.32419 10.55058 41.22928)

;; Loaded file test\_random.lsp

T

**Problem 2:** Consider polynomials in one variable, **x** represented as lists of the monomials. Thus, if **(3 2)** represents the term and **(-1 0)** represents the term then the list **((3 2)(-1 0))** stands for the polynomial 

Here are some examples:

|  |  |  |
| --- | --- | --- |
| math notation | internal representation | printed output form |
|  | **Nil** | 0 |
|  | **((2 1)(1 0))** | + 2x + 1 |
|  | **((3 2)(-1 0))** | 2  + 3x - 1 |
|  | **((-4 1)(5 2)(1 0))** | 2  – 4x + 5x + 1 |
|  | **((7 14)(9 3)(-3 2)(7 1))** | 14 3 2  + 7x + 9x - 3x + 7x |

Write a function **WRITE-POLY**, which takes an internal representation of a polynomial and produces the appropriate printed output form. You may print xk as x^k.

***Required test cases:***

**NIL**

**((2 1) (1 0))**

**((3 2) (-1 0))**

**((5 2) (-4 1) (1 0))**

**((7 14) (11 13) (-3 2) (7 1) (-5 0))**

**((1 0) (2 1) (-5 3) (-3 1) (7 0))**

Python:

def write\_poly\_main (L) :

if len (L) == 0 :

print("nil")

else :

print(write\_poly(L))

def write\_poly (L) :

if len(L) == 0 :

return ""

else :

pair = L[0]

if pair[0] >= 0 :

if pair[1] == 0 :

return "+" + str(pair[0]) + write\_poly(L[1:])

elif pair[1] == 1 :

return "+" + str(pair[0]) + "x" + write\_poly(L[1:])

else :

return "+" + str(pair[0])+"x^" + str(pair[1]) + write\_poly(L[1:])

else :

if pair[1] == 0 :

return str(pair[0]) + write\_poly(L[1:])

elif pair[1] == 1 :

return str(pair[0])+"x" + write\_poly(L[1:])

else :

return str(pair[0])+"x^" + str(pair[1]) + write\_poly(L[1:])

write\_poly\_main([])

write\_poly\_main([[2,1],[1,0]])

write\_poly\_main([[3,2], [-1, 0]])

write\_poly\_main([[5,2], [-4, 1], [1,0]])

write\_poly\_main([[7,14], [11,13], [-3,2], [7,1], [-5,0]])

write\_poly\_main([[1,0],[2,1],[-5,3],[-3,1],[7,0]])

'''

=============== RESTART: Z:/CS/Classes/CS352/PA/write\_poly.py ===============

nil

+2x+1

+3x^2-1

+5x^2-4x+1

+7x^14+11x^13-3x^2+7x-5

+1+2x-5x^3-3x+7

>>>

'''

Lisp:

(defun write-poly-main (L)

(cond ((null L) (print nil) (format t "~%" #\return))

(T (write-poly L) (format t "~%" #\return))))

(defun write-poly (L)

(cond ((null L) nil)

(T (setf pair (car L))

(setf p1 (car pair))

(setf p2 (car (cdr pair)))

(cond ((> p1 0) (format t "+~A" p1))

(T (format t "~A" p1))

)

(cond ((= p2 0) nil)

((= p2 1) (format t "x" p2))

(T (format t "x^~A" p2))

)

(write-poly (cdr L))

)

)

)

(write-poly-main 'nil)

(write-poly-main '((2 1) (1 0)))

(write-poly-main '((3 2) (-1 0)))

(write-poly-main '((5 2) (-4 1) (1 0)))

(write-poly-main '((7 14) (11 13) (-3 2) (7 1) (-5 0)))

(write-poly-main '((1 0) (2 1) (-5 3) (-3 1) (7 0)))

[13]> (load "write-poly.lsp")

;; Loading file write-poly.lsp ...

NIL

+2x+1

+3x^2-1

+5x^2-4x+1

+7x^14+11x^13-3x^2+7x-5

+1+2x-5x^3-3x+7

;; Loaded file write-poly.lsp

T