Requerimientos Programa 2

# Contador de Lineas de Código LOC:

El programa debe contar las líneas de un programa, dada la ruta en donde está ubicado. El programa al que se van a contar las líneas de código y los objetos es un programa Python. Los siguientes son los requerimientos para el contador de LOC:

1. Contar todas las líneas, excepto las líneas que contienen solo comentarios y las líneas vacías. Los comentarios están identificados por tres doble comilla “”” ó un # al inicio de la línea.

Ej:

|  |
| --- |
| 1. def get\_next(self):   “”” Este es un bloque de comentarios  que no se debe contar  “”” |
| 1. next = self.i+1   # Este es otro comentario que no se debe contar   1. return next |

1. El programa debe contar las líneas de código en total.
2. El programa debe contar las líneas de código de cada elemento: módulo, clase, función, etc.
3. Cada clase está identificada con la palabra class al principio de la línea
4. Cada función está identificada con la palabra def al principio de la línea
5. Según el scope en el que se encuentra la línea, se cuenta dentro del módulo, clase ó función.
6. La salida del programa debe mostrar las líneas en total y las líneas por objeto en la misma pantalla o en un archivo plano.

Ej:

Total: 150

module linked\_list.py : 90

class Node: 30

class LinkedList: 60

def get\_prior: 15

def get\_next: 15

def get\_lenght: 20

def \_\_init\_\_: 10

module main.py:20

Conceptual Design

lines\_of\_code.py

Units = [Unit] Recursive get\_modules\_of\_path(path)-> return Units[]

get\_lines\_of\_module()->lines

inspect\_module(u Unit) /Recursively get the values of units nested to u, in global Unit. First time read the file and record line by line counting and searching for class or def line/-> Unit

unit.py

Type= Module, class or def

Lines= Number of lines

Units = [Unit] Recursive nested units

Path = in case of modules the path of the file

main.py

/\*Catch the path to evaluate. Use the lines\_of\_code class\*/

Lines\_of\_code.py =