**SEGUNDA PRÁCTICA DE LÓGICA II (2019/2)**

La práctica se desarrollará en equipos de dos estudiantes. Serán 8 prácticas en las cuales la única diferencia será el tipo de lista en que la debe implementar:

* Práctica 1: listas simplemente ligadas.
* Práctica 2: listas simplemente ligadas circulares.
* Práctica 3: listas simplemente ligadas circulares con nodo cabeza.
* Práctica 4: listas simplemente ligadas con nodo cabeza.
* Práctica 5: listas doblemente ligadas.
* Práctica 6: listas doblemente ligadas circulares.
* Práctica 7: listas doblemente ligadas circulares con nodo cabeza.
* Práctica 8: listas doblemente ligadas con nodo cabeza.

Para saber cuál práctica le corresponde basta con sumar los números correspondientes a los dos últimos dígitos de sus cédulas, al resultado le hace módulo 8 y le suma 1. El resultado de esa operación le indica la práctica que le toca. Por ejemplo, si las cédulas de los integrantes son 32486738 y 70456816 se suma 38 + 16 = 54, 54%8 = 6 a este módulo le sumo 1 y obtengo 7, por tanto les corresponde la práctica 7.

No se aceptan errores en este cálculo. Es decir quienes desarrollen la práctica que no es, porque se equivocaron haciendo este cálculo se les calificará la práctica con 0.0. OK ?

Su programa debe ser totalmente interactivo y en ambiente gráfico.

Prácticas iguales se califican con 0.0

Prácticas con sospecha de copia se hará sustentación.

La práctica debe enviarse en formato  .zip y debe nombrarse con el siguiente formato [Estudiante1][Estudiante2][#practica].zip, y que en el correo de entrega se especifiquen los integrantes con nombres y apellidos.

Fecha de vencimiento: Mayo 22 del 2020 a las 11:59 pm

**ENUNCIADO DE LA PRACTICA**

Construya programa de computador para manipular listas ligadas. Su programa de computador deberá estar en forma gráfica y aceptar en forma interactiva la entrada de datos. Los datos serán numéricos y los entrará el usuario.

Con base en los datos que vaya entrando el usuario, su programa deberá construir 4 listas ligadas: una con los datos pares (L2), otra con los datos múltiplos de tres (L3), otra con los datos múltiplos de 5 (L5) y una cuarta (L0) con los datos que no correspondan a alguna de las anteriores características.

La lista con los datos pares la debe construir insertando datos siempre al final de la lista, la lista con los múltiplos de tres la construye insertando datos siempre al principio de la lista, la lista con los datos múltiplos de 5 la construye de tal forma que los datos queden ordenados ascendentemente a medida que la va construyendo y la cuarta lista la construye de tal manera que los datos queden ordenados descendentemente a medida que la va construyendo.

Si el dato entrado es par y múltiplo de 3 deberá quedar en L2 y en L3, si es par y múltiplo de 5 deberá quedar en L2 y en L5, si es múltiplo de 3 y múltiplo de 5 deberá quedar en L3 y L5, y si es par, múltiplo de 3 y múltiplo de 5 deberá quedar en las tres listas.

Por ejemplo, si el número entrado es 16, sólo deberá quedar en la lista L2; si el número entrado es 21, sólo deberá quedar en la lista L3; si el número entrado es 35, sólo deberá quedar en la lista L5; si el número entrado es 24, el 24 deberá quedar en L2 y también en L3; si el número entrado es 30, deberá quedar en L2, L3 y L5.

Cuando el usuario decida no entrar más datos su programa deberá mostrar por pantalla las listas construidas y hacer lo siguiente con ellas:

* Construir una nueva lista que sea la intersección de L2 con L3.
* Construir una nueva lista que sea la intersección de L2 con L5.
* Construir una nueva lista que sea la intersección de L3 con L5.
* Construir una nueva lista que sea la intersección de L2, L3 y L5.
* Mostrar por pantalla las listas originales y las que acabó de construir.
* Mostrar por pantalla los datos de L2 que no está ni en L3 ni en L5.
* Mostrar por pantalla los datos de L3 que no están ni en L2 ni en L5.
* Mostrar por pantalla los datos de L5 que no están ni en L2 ni en L3.
* Construir una nueva lista (LZ) que sea la unión de los datos de las listas L0, L2, L3 y L5.
* Mostrar por pantalla la lista LZ.
* Ordenar la lista LZ de tal manera que la primera mitad de los datos queden ordenados ascendentemente y la segunda mitad descendentemente.
* Mostrar por pantalla la lista LZ después de ordenada.

Para implementar todo lo pedido debe definir una clase con la lista que le corresponde. Dicha clase deberá incluir métodos necesarios para efectuar todas las operaciones solicitadas. Su programa principal deberá hacer uso de esa clase con sus métodos.

Para efectos de la práctica defina la clase de la lista que le corresponda, es decir no la defina derivada de ninguna otra. OK?