Parcial Algoritmos y Estructuras de datos CB100 Curso Schmidt (3° instancia) 28 de Noviembre 2024

Padrón: **Apellido y Nombre:**

Mail:

Puntero) Indicar la salida

```
/* Datos:
 *   '@' es 64
 *   'A' es 65 'B' es 66
 */
public static void main(String[] args) {
    int A, C, F = 0;
    char G;
    int H;
    int[] L = new int[7];
    int[] M = L;
    H = 71;
    G = 'D';
    A = H;
    L[0] = A;
    M[1] = L[0];
    System.out.println(A + "-" + G + "-" + M[0] + "-" + L[1]);
    G = (char) H;
    M[2] = L[0] + A;
    System.out.println(G + "-" + H + "-" + M[1] + "-" + L[2]);

    while (L[0] > 10) {
        H--;
        A--;
        C = A;
        System.out.println(C + "-" + H + "-" + M[1] + "-" + F);
        F = A;
        C--;
        M[0] -= 30;
    }
    //Que hace el garbage collector y que puntero libera
}
```

TDA) Implementar una clase que modele una Subasta:
a) Una Subasta debe ser creada con la descripción del artículo y opcionalmente el precio base de la misma. Si no se especifica, el precio base es $ 0.
b) La Subasta debe poseer métodos para realizar una oferta y conocer cuál es, hasta el momento, la oferta ganadora (la mayor de todas las ofertas realizadas).
c) Inicialmente la Subasta no está abierta (está pendiente, por tanto, no recibe ofertas) y debe proveer un método para abrirla (a partir de lo cual recibe ofertas).
d) La Subasta debe proveer un método para cerrarla, a partir de lo cual no recibe más ofertas. Una vez cerrada no puede ser abierta nuevamente.

LISTA) Un Blog está compuesto por un conjunto de Artículos. Cada Artículo posee un texto.
Implementar el método buscarArticulos de la clase Blog a partir de la siguiente especificación de su interfaz:

```
class Blog {

  public:
    /**
     * post: el Blog queda creado sin Artículos.
     */
    Blog();
    /**
     * post: devuelve los Artículos que componen el Blog.
     */
    Lista<Articulo> getArticulos();
    /**
     * post: devuelve una nueva Lista con los Artículos que tengan en su texto todas las palabras claves en ese
orden y aparecen solo una vez cada una.
     */
    Lista<Articulo> buscarArticulos(Lista<String> palabrasClaves);
}
```

Ayuda:

```
String texto = "Hola mundo";
int indice = texto.indexOf("mundo"); //=5
String temp = texto.substring(indice);//=mundo
```

Los alumnos que tienen aprobado algún punto, no deben hacerlo.
Para aprobar es necesario tener al menos el 60% de los ejercicios correctos y completos.
Todas las implementaciones deben respetar la teoría de TDA.
Duración del examen : 2 horas y 30 minutos