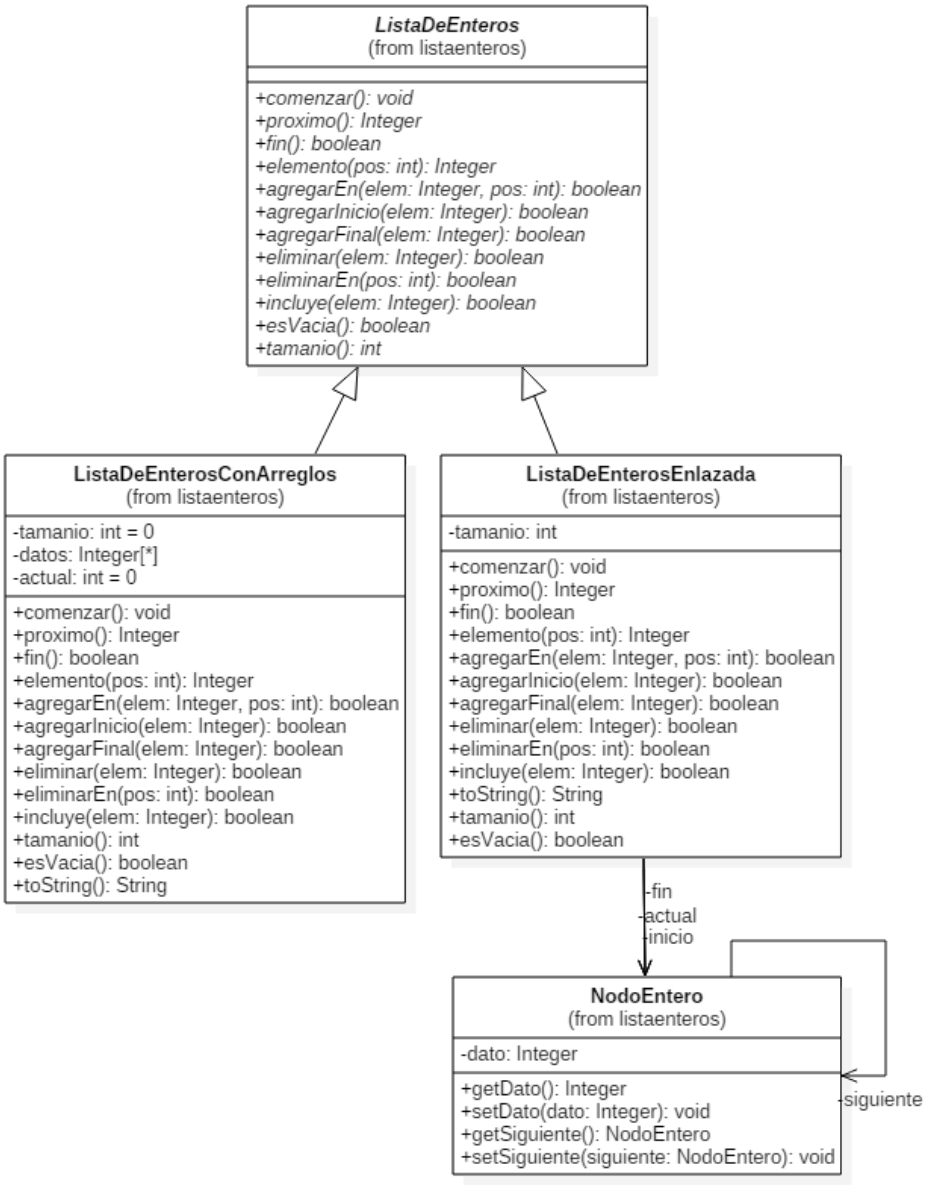
Programación III
TEMA 3: Listas de Enteros
Práctica nº 3 - A

Tema: Abstracción, Encapsulamiento, Herencia, Tipos Genéricos. Listas.

Importante

- Descargue el material disponible en el sitio de la cátedra.
- En esta práctica necesitará la implementación de la cátedra de **ListaDeEnteros**. En particular se trabajará con la implementación de una subclase, la **ListaDeEnterosEnlazada**. Importe el archivo de la cátedra en eclipse**.** Se recomienda trabajar a partir de esta práctica en un mismo proyecto (Programacion3).



1. **Teniendo en cuenta la implementación propuesta:**
    a. Indique 2 motivos por los cuales la clase ListaDeEnteros se define como abstracta. Note que una subclase implementa la lista usando un arreglo de tamaño fijo y la otra usando nodos enlazados.
    b. La clase **ListaDeEnteros** está definida como clase **abstracta**. ¿Podría ud. agregar comportamiento en algún método de la clase **ListaDeEnteros**?
    c. Escriba una clase llamada **ListaDeEnterosEnlazadaTestBasico** (con método main) en el paquete prog3.listaenteros.test. Pruebe el comportamiento de la clase **ListaDeEnterosEnlazada** creando una instancia de la misma e invocando al método agregar (indique valores en el mismo código), luego recorra la lista e imprima los elementos de la misma.
    d. Escriba un método **recursivo** que imprima los elementos de una lista en sentido inverso. La lista la recibe por parámetro.
    e. Observe la implementación interna de la clase **ListaDeEnterosEnlazada**, ¿qué diferencias existen entre agregar un nodo al principio de la lista, agregar un nodo en el medio y agregar un nodo al final, respecto de la cantidad de operaciones que debe realizarse?

2. **Método "ordenar" de ListaDeEnterosEnlazada.** Implemente un método de instancia llamado *ordenar,* que asumiendo que la lista contiene valores mayores a 0 y que todos sus elementos son diferentes, devuelve una nueva lista ordenada usando la siguiente estrategia: *"seleccionar el menor elemento de la lista y colocarlo al final de la lista resultado"*. Tenga en cuenta que la lista original NO debe modificarse.
    a. La firma del método deberá ser la siguiente:

```
public ListaDeEnterosEnlazada ordenar();
```

    b. Escriba una clase llamada TestOrdenamientos y verifique el correcto funcionamiento del método *ordenar*.
    c. Suponga que la lista a ordenar contiene N elementos. Indique ¿cuántos elementos recorrió de la lista original y en el peor de los casos, para generar la nueva lista resultante?

3. **Método "combinarOrdenado" de ListaDeEnterosEnlazada.** Implemente un método llamado *combinarOrdenado* que reciba 1 lista de elementos ordenada y devuelve una nueva lista también ordenada conteniendo los elementos de las 2 listas (la lista receptora del mensaje y la lista recibida por parámetro) . Como precondición, la lista que recibe el mensaje "*combinarOrdenado*" también estará ordenada.
    a. La firma del método deberá ser la siguiente:

```
public ListaDeEnterosEnlazada combinarOrdenado(ListaDeEnterosEnlazada listaParam);
```

    b. Escriba en la clase TestOrdenamientos escriba el código necesario para verificar el correcto funcionamiento del método *combinarOrdenado*.

c. Suponga que las listas a combinar tienen N y M elementos respectivamente. Indique ¿cuantos elementos recorrió para generar la nueva lista resultante?

4. **MergeSort.** La estrategia del MergeSort consiste en ordenar una lista dividiendo "el problema" (la lista a ordenar) recursivamente hasta llegar a un punto en que no se puede dividir más. Luego, a medida que se vuelve de la recursión, se devuelven listas ya ordenadas y simplemente se combinan. Puede leer más acerca del mergesort en:
   - https://www.khanacademy.org/computing/computer-science/algorithms/merge-sort/a/overview-of-merge-sort

      i. Cree en el paquete prog3.util una clase llamada UtilitariosLista, e implemente el método **mergeSort.**

5. **JUnit (OPCIONAL)**

- Las librerías en el proyecto, fueron descargadas del sitio https://github.com/junit-team/junit/releases y corresponen a JUnit, un conjunto de clases que permiten escribir "pruebas".
- En caso que no se encuentre incluída en su poyecto, incluya dicha librería (cree una carpeta lib de modo que la librería quede dentro de su proyecto)
- Ejecute la clase ListaDeEnterosEnlazadaJUnitTest y verifique que los Test se ejecutan exitosamente.