

Asignatura	Datos del alumno	Fecha
Estructura de Datos	Apellidos: Pilay ***	11/12/2025
	Nombre: Tyrone ****	

Actividad en Contacto con el Docente

TAREA DE UNIDAD III

Indicaciones:

Este documento fue desarrollado con la finalidad de guiar al estudiante en las tareas de los contenidos tratados en la unidad III, posee una serie de instrucciones que el estudiante debe aplicar para reforzar los conocimientos profesionales correspondiente al nivel de estudio de la carrera.

Objetivos

El trabajo de esta asignatura consistirá en desarrollar su capacidad de conceptualización para complementar los temas tratados en la Unidad No.3.

Descripción de la actividad y pautas de elaboración

- Para realizar esta tarea deberá consultar el cronograma de actividades del aula virtual.
- La tarea es individual y por lo tanto cada estudiante asumirá la responsabilidad de autoría, si existen dos o más trabajos iguales, la calificación será de cero (0) para el número de alumnos que incurran en esta falta.
- En caso de que el estudiante tenga dudas en los temas tratados en clase, podrá utilizar las diapositivas, compendio y videos que se adjuntan en el aula virtual.
- Queda prohibido el uso de librerías no estudiadas en clases, tales como StringBuilder, Stack, ArrayList entre otras, de hacerlo su nota será de cero (0).
- **Adjuntar carpeta comprimida con los códigos de programación**

Facultad de Ciencias Informáticas | Carrera de Tecnologías de la Información

Asignatura	Datos del alumno	Fecha
Estructura de Datos	Apellidos: Pilay ***	11/12/2025
	Nombre: Tyrone ****	

Actividades por desarrollar

La Facultad de Ciencias Informática ha realizado una evaluación para cada estudiante que aspira un cupo en esta Facultad.

De cada Estudiante la facultad tiene el código único del postulante (de tipo entero), el nombre (String) y el puntaje sobre 100 puntos (puede ser float)

Estos datos se registran en una lista doble, cuando el proceso de registro finaliza, se deberá seleccionar a los primeros 50 estudiantes que obtuvieron un puntaje de postulación superior a los 75 puntos (nota tope), en caso de no cubrir los 50 cupos deberá completar los 50 cupos con los estudiantes con los puntajes más próximos a la nota tope.

Se requiere:

- **Procedimiento Insertar:** que genere una lista de estudiantes con 75 códigos de estudiantes (códigos consecutivos de 1, 2, 3 hasta llegar a los 75 que sería el último estudiante ingresado) y así mismo se genera una puntuación lograda (genere las calificaciones de forma aleatoria) en un rango de valor entre 20 y 100 puntos.
- Genera la selección de los 50 postulantes admitidos.
- Muestre todos los postulantes generados (Código único, Nombre y puntaje)
- Muestre la selección de los 50 cupos (Código único, Nombre y puntaje)

Asignatura	Datos del alumno	Fecha
Estructura de Datos	Apellidos: Pilay ***	11/12/2025
	Nombre: Tyrone ****	

Realizar pruebas y validar mi solución propuesta

¿Por qué elegí esta estructura para conseguir la solución al problema?

Para esta tarea elegí una lista doblemente enlazada con su NodoDoble porque me permitía recorrer la lista en ambos sentidos, aunque en la práctica solo la recorrió del inicio al final. Aun así, dejé los métodos porque más adelante podría mejorarla o ampliarla.

También sentí que al hacerla doble iba a aprender más y aplicar lo enseñado por el profesor, por lo que ese fue otro motivo para elegirla.

Explicación de la creación de la lista, el Nodo y la entidad

Primero empecé con la creación de la entidad, que en este caso se llama **Estudiante**.

La tarea me pedía que tuviera tres datos:

- un identificador,
- su nombre,
- y el puntaje de postulación.

En mi caso, el identificador lo creé con una variable llamada id de tipo int.

El otro atributo lo llamé nombre de tipo String.

Y el último atributo lo puse como puntaje de tipo float.

Los tres atributos los establecí como private.

Después creé los constructores (vacío y con datos), los métodos **getters** y **setters**, y sobreescribí el método **toString()** para mostrar bien los datos que pedía la tarea.

```
package com.tyrone.tarea6;

/**
 *
 * @author tyron
 */
public class Estudiante {
    private int id;
    private String nombre;
    private float puntaje;

    public Estudiante() {
```

Asignatura	Datos del alumno	Fecha
Estructura de Datos	Apellidos: Pilay *** Nombre: Tyrone ****	11/12/2025

}

```

public Estudiante(int id, String nombre, float puntaje) {
    this.id = id;
    this.nombre = nombre;
    this.puntaje = puntaje;
}

public int getId() {
    return id;
}

public void setId(int id) {
    this.id = id;
}

public String getNombre() {
    return nombre;
}

public void setNombre(String nombre) {
    this.nombre = nombre;
}

public float getPuntaje() {
    return puntaje;
}

public void setPuntaje(float puntaje) {
    this.puntaje = puntaje;
}

```

Asignatura	Datos del alumno	Fecha
Estructura de Datos	Apellidos: Pilay *** Nombre: Tyrone ****	11/12/2025

```

@Override
public String toString() {
    String nombreLimpio = (nombre.charAt(0) == ' ') ? nombre.substring(1)
: nombre;
    return "{" + "id= " + id + ", nombre= " + nombreLimpio + ", puntaje=
" + puntaje + '}';
}

}

```

Ahora que ya tengo la entidad que contendrá los datos, procedí a crear el Nodo. En mi caso escogí el Nodo doble por la practicidad que mencioné antes. Primero creé la clase con el nombre **NodoDoble**, luego creé los atributos **ante** y **sigue** del mismo tipo de clase, y otro atributo de nombre **dato** que contiene un objeto Estudiante.

Todos los atributos son **private**.

Luego hice los constructores (vacío y con datos) y los getters y setters.

```

package com.tyrone.tarea6;

/**
 *
 * @author tyron
 */
public class NodoDoble {
    private NodoDoble ante;
    private Estudiante dato;
    private NodoDoble sigue;

    public NodoDoble() {
    }

    public NodoDoble(NodoDoble ante, Estudiante dato, NodoDoble sigue) {
        this.ante = ante;
    }
}

```

Asignatura	Datos del alumno	Fecha
Estructura de Datos	Apellidos: Pilay *** Nombre: Tyrone ****	11/12/2025

```

    this.dato = dato;
    this.sigue = sigue;
}

public NodoDoble getAnte() {
    return ante;
}

public void setAnte(NodoDoble ante) {
    this.ante = ante;
}

public Estudiante getData() {
    return dato;
}

public void setData(Estudiante dato) {
    this.dato = dato;
}

public NodoDoble getSigue() {
    return sigue;
}

public void setSigue(NodoDoble sigue) {
    this.sigue = sigue;
}

}

```

Asignatura	Datos del alumno	Fecha
Estructura de Datos	Apellidos: Pilay *** Nombre: Tyrone ****	11/12/2025

Ya teniendo las dos clases, procedí a crear la clase **ListaDoble**, donde implementé los métodos característicos.

Primero creé dos atributos llamados inicio y fin, ambos de tipo NodoDoble.

Luego puse los constructores y a continuación los métodos principales.

El primero que uso es **ListaVacia()**, que devuelve un boolean dependiendo si la variable inicio es nula.

```
package com.tyrone.tarea6;
```

```
/*
 *
 * @author tyron
 */
public class ListaDoble {
    //atributos
    private NodoDoble inicio;
    private NodoDoble fin;

    //constructor

    public ListaDoble() {
    }

    public ListaDoble(NodoDoble inicio, NodoDoble fin) {
        this.inicio = inicio;
        this.fin = fin;
    }

    //Metodos
    //Lista vacia
    public boolean ListaVacia(){
        return inicio == null;
    }
}
```

Asignatura	Datos del alumno	Fecha
Estructura de Datos	Apellidos: Pilay ***	11/12/2025
	Nombre: Tyrone ****	

Luego creé el método llamado **InsertInicio()**, en el cual primero genero un nuevo nodo llamado **nuevo**. Ese nodo lo creo llamando al constructor **NodoDoble(null, dato, null)**, lo cual significa que inicialmente sus dos extremos (el anterior y el siguiente) están en **null**, y en la parte del centro le paso la variable **dato**, que en este caso son los datos del Estudiante.

Después de crear este nodo, hago una validación con un **if** usando el método **ListaVacia()** para saber si la lista está vacía. Si la lista está vacía, entonces simplemente hago que **inicio** sea igual a **nuevo** y también **fin** sea igual a **nuevo**, porque cuando hay un solo elemento, inicio y fin apuntan al mismo nodo.

Si la lista **no** está vacía (o sea, entra al **else**), entonces tengo que enlazar manualmente los extremos del nuevo nodo con el inicio actual. Para eso realizo los siguientes pasos:

1. Con **nuevo.setDatos(dato)** me aseguro de colocar el dato del estudiante dentro del nodo nuevo.
2. Luego, con **nuevo.setSigue(inicio)** hago que el nodo nuevo apunte al nodo que antes era el inicio.
3. Despues, con **inicio.setAnte(nuevo)** le digo al nodo que antes era el inicio que su anterior ahora sea el nodo nuevo.
4. Finalmente asigno **inicio = nuevo**, lo que significa que el nodo nuevo ahora es oficialmente el primer nodo de la lista básicamente lo actualizo.

```
public void InsertInicio(Estudiante dato){
    NodoDoble nuevo = new NodoDoble(null, dato, null);
    if (ListaVacia()) {
        inicio = nuevo;
        fin = nuevo;
    } else {
        nuevo.setDatos(dato);
        nuevo.setSigue(inicio);
        inicio.setAnte(nuevo);
        inicio = nuevo;
    }
}
```

Facultad de Ciencias Informáticas | Carrera de Tecnologías de la Información

Asignatura	Datos del alumno	Fecha
Estructura de Datos	Apellidos: Pilay ***	11/12/2025
	Nombre: Tyrone ****	

También hice los métodos InsertatFinal() que es similar al método InsertInicio() por lo que no lo explicare, EliminarFinal() y EliminarInicio() esos tampoco los explicare porq no los ocupe en la tarea.

```

public void InsertFinal(Estudiante dato){
    NodoDoble nuevo = new NodoDoble(null, dato, null);
    if (ListaVacia()) {
        inicio = nuevo;
        fin = nuevo;
    } else {
        fin.setSigue(nuevo);
        nuevo.setAnte(fin);
        fin = nuevo;
    }
}

// Eliminar inicio o final
public String EliminarFinal(){
    if (ListaVacia()) {
        return "Lista vacia";
    } else {
        String n = fin.getDato().getNombre();
        fin = fin.getAnte();
        fin.setSigue(null);
        return n;
    }
}

public Estudiante EliminarInicio(){
    if (ListaVacia()) {
        System.out.println("Lista vacia");
        return null;
    } else {

```

Asignatura	Datos del alumno	Fecha
Estructura de Datos	Apellidos: Pilay *** Nombre: Tyrone ****	11/12/2025

```

Estudiante n = inicio.getData();
inicio = inicio.getSigue();
inicio.setAnte(null);
return n;
}
}

```

Posteriormente creé el método **Imprimir()**, en el cual primero verifico si la lista está vacía usando un if con el método **ListaVacia()**.

Si está vacía, entonces simplemente muestro un mensaje por consola indicando que la lista está vacía.

Pero si no está vacía, entonces procedo a recorrerla.

Para eso creo un nodo auxiliar llamado **aux** y le asigno el valor de inicio, para comenzar el recorrido desde el primer nodo.

Luego utilizo un **while** que seguirá ejecutándose mientras **aux** sea diferente de null. Dentro del ciclo:

- imprimo los datos del postulante usando **aux.getData()**
- luego actualizo el auxiliar haciendo **aux = aux.getSigue()**, lo cual permite recorrer nodo por nodo hasta que finalmente llegue al final donde será null y terminará el ciclo.

```

public void Imprimir(){
    if(ListaVacia()) {
        System.out.println("Lista esta vacia");
    } else {
        NodoDoble aux = inicio;
        while(aux != null){
            System.out.println("Postulante " + aux.getData());
            aux = aux.getSigue();

        }
    }
}

```

Asignatura	Datos del alumno	Fecha
Estructura de Datos	Apellidos: Pilay ***	11/12/2025
	Nombre: Tyrone ****	

Parte de la explicación del método de ordenamiento y la selección de 50 alumnos

Aquí implementé una lógica para ordenar los datos sin alterar la lista original. Para eso creé el método **ListaPuntajeAscendente()** que devuelve un objeto de tipo ListaDoble, porque necesito retornar la lista ya ordenada.

Primero creo una ListaDoble llamada **copia**, que utilizaré para guardar la copia exacta de todos los estudiantes sin modificar nada de la lista original.

Después verifico con un if usando **ListaVacia()** si la lista está vacía. Si está vacía, imprimo un mensaje. Pero si no está vacía, procedo con la lógica.

Primero declaro un nodo auxiliar llamado **aux** y le asigno el inicio de la lista actual.

Luego hago un while que recorre toda la lista original desde el inicio hasta que **aux** sea null. En cada vuelta del ciclo:

- uso **copia.InsertFinal(aux.getData())** que inserta al final de la lista copia el dato del estudiante
- después actualizo el auxiliar con **aux = aux.getSigue()**

De esa forma, al finalizar ese **while**, tengo una lista copia con exactamente los mismos elementos que la lista original pero sin alterar nada en ella.

Luego viene la parte del ordenamiento.

Para ordenar la lista usé dos nodos:

- **nodoComparacion1**

Toma el valor de **copia.inicio** y servirá para ir comparando elemento por elemento.

- **nodoComparacion2**

Este nodo toma el valor de **nodoComparacion1.getSigue()** y sirve para comparar el nodo actual con todos los que vienen después.

Primero hago un **while** para que **nodoComparacion1** recorra toda la lista.

Dentro de ese while creo **nodoComparacion2** para que empiece desde el siguiente nodo hacia adelante.

Dentro del segundo **while** comparo los puntajes.

La condición es:

Si el puntaje de **nodoComparacion2** es mayor que el de **nodoComparacion1**, entonces intercambio los datos.

Para hacer ese intercambio:

Asignatura	Datos del alumno	Fecha
Estructura de Datos	Apellidos: Pilay ***	11/12/2025
	Nombre: Tyrone ****	

1. guardo el dato de nodoComparacion1 en una variable temporal tipo Estudiante llamada **temp**.
2. luego asigno a nodoComparacion1 el dato de nodoComparacion2.
3. finalmente asigno a nodoComparacion2 el dato de **temp**.

Así se hace un intercambio básico de valores.

Luego actualizo nodoComparacion2 con **getSigue()** para que siga recorriendo su propio ciclo. Al terminar ese ciclo, actualizo nodoComparacion1 también con **getSigue()**.

Finalmente retorno la lista copia ya ordenada.

```

public ListaDoble ListaPuntajeAscendente() {

    ListaDoble copia = new ListaDoble();

    if (ListaVacia()) {
        System.out.println("Lista vacía");

    }else{

        NodoDoble aux = inicio;

        while (aux != null) {
            copia.InsertFinal(aux.getDato());
            aux = aux.getSigue();
        }
    }

    NodoDoble nodoComparacion1 = copia.inicio;

    while (nodoComparacion1 != null) {
        NodoDoble nodoComparacion2 = nodoComparacion1.getSigue();

        while (nodoComparacion2 != null) {
    
```

Asignatura	Datos del alumno	Fecha
Estructura de Datos	Apellidos: Pilay ***	11/12/2025
	Nombre: Tyrone ****	

```

        if (nodoComparacion2.getData().getPuntaje() >
nodoComparacion1.getData().getPuntaje()) {
    Estudiante temp = nodoComparacion1.getData();
    nodoComparacion1.setData(nodoComparacion2.getData());
    nodoComparacion2.setData(temp);
}
nodoComparacion2 = nodoComparacion2.getSigue();
}

nodoComparacion1 = nodoComparacion1.getSigue();
}
}

return copia;
}

```

Con el método anterior ya tengo una lista ordenada, por lo que ahora creé el método **SelecTop50MejAlumnosEnCalif()** para escoger los 50 estudiantes con puntajes mayores o iguales a 75 y, si no alcanza, completar con los siguientes más cercanos al puntaje tope.

Primero verifico si la lista está vacía.

Si no está vacía:

- creo una ListaDoble llamada **seleccionados**
- creo otra ListaDoble llamada **ordenada** que captura el retorno de **ListaPuntajeAscendente()**
- creo un NodoDoble llamado **aux** que apunta al inicio de la lista ordenada
- creo dos variables:
 - cupos = 50
 - cuposActuales = 0

Luego uso un ciclo for desde **i = 1** hasta lo q contenga la variable **cupos**.

Uso **i** desde 1 para ir numerando los postulantes aceptados.

Dentro del **for** hago una condición:

Si el puntaje de aux es mayor o igual a 75

Entones lo inserto a la lista **seleccionados**, actualizo **cuposActuales = cupos - 1** e imprimo el dato de ese **aux**.

Facultad de Ciencias Informáticas | Carrera de Tecnologías de la Información

Asignatura	Datos del alumno	Fecha
Estructura de Datos	Apellidos: Pilay ***	11/12/2025
	Nombre: Tyrone ****	

Si el puntaje es menor a 75

También lo inserto porque la tarea manda a completar los cupos con los más cercanos.
Actualizo cuposActuales e imprimo, básicamente hago lo mismo que lo que tiene la primera condición.

Si no se cumplen las condiciones (algo raro o cantidad de postulante menor a 50)

Muestro cuántos cupos sobraron.

Al final de cada iteración actualizo aux con **aux.getSigue()**.

Y aunque tenga esa lista con esos postulantes seleccionados, no la retornare y dejare la lógica como esta ahora, pero en un futuro cuando vuelva a descargar mis apuntes practicaré y cambiare el tipo de dato de este método y retornare la lista

```
public void SelecTop50MejAlumnosEnCalif(){

    if (ListaVacia()) {
        System.out.println("Lista vacía");

    }else{
        ListaDoble seleccionados = new ListaDoble();
        ListaDoble ordenada = ListaPuntajeAscendente();
        NodoDoble aux = ordenada.inicio;

        int cupos = 50, cuposActuales=0;

        for (int i = 1; i <= cupos; i++) {
            if (aux.getData().getPuntaje()>=75) {

                seleccionados.InsertFinal(aux.getData());
                cuposActuales = cupos-1 ;
                System.out.println(i+"~ Estudiante "+ aux.getData());

            } else if (aux.getData().getPuntaje()<75) {
```

Asignatura	Datos del alumno	Fecha
Estructura de Datos	Apellidos: Pilay *** Nombre: Tyrone ****	11/12/2025

```

    seleccionados.InsertFinal(aux.getDato());
    cuposActuales = cupos-1 ;
    System.out.println(i+"~ Estudiante "+ aux.getDato());

} else{
    System.out.println("sobraron"+ cuposActuales + " cupos");
}

aux = aux.getSigue();

}

}

```

Cómo ocupo y funciona mi solución

Uso este código en mi clase Main llamada MainPuntajeMejoresPostulacion.

Primero creo una ListaDoble llamada **listaDoble**.

Luego muestro un mensaje por consola simulando el registro de datos.

Después uso un ciclo **for** que se repetirá 75 veces, desde i = 1 hasta i = 75.

Lo puse así porque la tarea pide 75 códigos con datos de estudiantes y además necesito que el ID vaya del 1 al 75 y por eso inicie **i** en 1 para poder usarlo.

Dentro del for:

1. Creo el puntaje aleatorio

- Uso Math.random() multiplicado por 8000 y dividido para 100.0
- Esto me da un número con decimales
- Lo redondeo con Math.round
- Y le sumo 20 para asegurar que no baje de ese valor

2. Simulo nombres

Hice un String muy largo llamado nombres.

Para sacar una parte de ese string uso substring, pero para eso necesito índices válidos.

Creo la variable indice usando Math.random() multiplicado por la longitud del string

Asignatura	Datos del alumno	Fecha
Estructura de Datos	Apellidos: Pilay ***	11/12/2025
	Nombre: Tyrone ****	

menos 12 caracteres (para no salirme del rango cuando lo sume posteriormente). Le sumo i para intentar que no se repitieran, pero al final eso no importa mucho porque en el `toString` se limpia el espacio inicial y el índice al caer en un espacio se nota como diferente pero al eliminar ese espacio en el `toString` no retorna un valor igual a un número anterior o posterior.

Luego creo el nombre con:

`nombres.substring(indice, indice + 12)`

3. Creo un objeto Estudiante con estas variables

`Estudiante estudiante = new Estudiante(i, nombre.toUpperCase(), puntaje);`

4. Lo inserto a la lista

`listaDoble.InsertInicio(estudiante);`

5. Simulo la barra de carga

`System.out.print("=");`

Cuando el ciclo llega a 75 muestra el mensaje “Datos cargados con éxito”.

Después imprimo un mensaje por consola “Todos los postulantes” y a todos los postulantes usando el método `Imprimir()`, y finalmente muestro por consola el mensaje “Selección de los 50 cupos:” y llamo al método `SelecTop50MejAlumnosEnCalif()` para mostrar quienes alcanzaron estos cupos.

```
package com.tyrone.tarea6;

/*
 *
 * @author tyron
 */
public class MainPuntajeMejoresPostulacion {

    public static void main(String[] args) {
        ListaDoble listaDoble = new ListaDoble();

        System.out.println("Procesando registros de datos...");

        for (int i = 1; i <= 75; i++) {

```

Facultad de Ciencias Informáticas | Carrera de Tecnologías de la Información

Asignatura	Datos del alumno	Fecha
Estructura de Datos	Apellidos: Pilay *** Nombre: Tyrone ****	11/12/2025

```
float puntaje = (float)((Math.round(Math.random() * 8000) / 100.0)+ 20.0);
```

```
String nombres = "Tyrone Joel Juan Gabriel Pedro Luis Carlos Maria Elena Pepa Sofía Alejandra
Ana";
```

```
int indice = ((int)(Math.random() * (nombres.length() - 12)) + i) % (nombres.length() - 12);
```

```
String nombre = nombres.substring(indice, indice + 12);
```

```
Estudiante estudiante = new Estudiante(i, nombre.toUpperCase(), puntaje);
```

```
listaDoble.InsertInicio(estudiante);
```

```
System.out.print("=");
```

```
if (i == 75) {
```

```
    System.out.println("\n====Datos Cargados con éxito====");
```

```
}
```

```
}
```

```
System.out.println("Todos los postulantes:");
```

```
listaDoble.Imprimir();
```

```
System.out.println("Selección de los 50 cupos:");
```

```
listaDoble.SelecTop50MejAlumnosEnCalif();
```

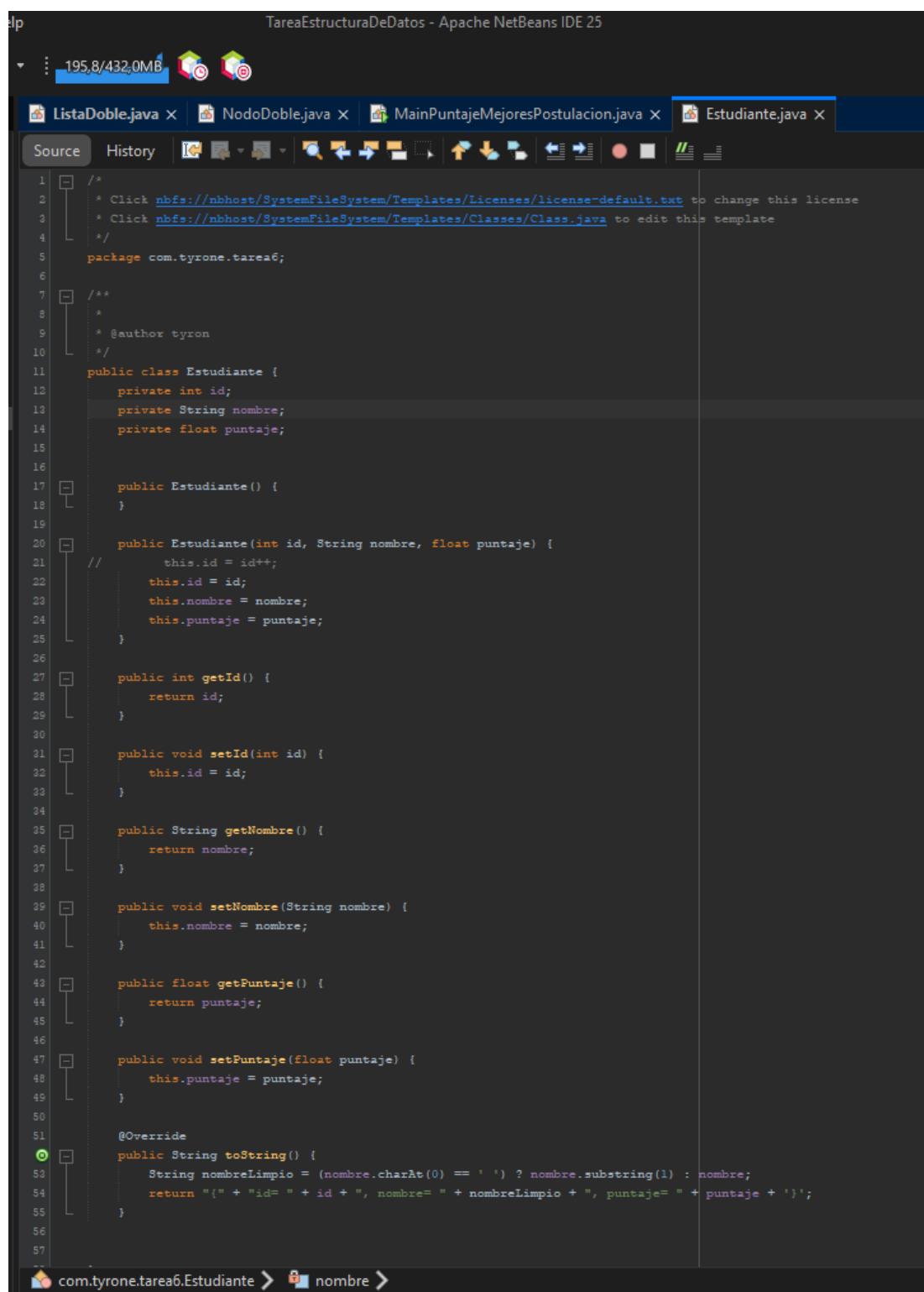
```
}
```

```
}
```

Asignatura	Datos del alumno	Fecha
Estructura de Datos	Apellidos: Pilay ***	11/12/2025
	Nombre: Tyrone ****	

Captura de las clases

Clase Estudiante:



The screenshot shows the Apache NetBeans IDE 25 interface with the title bar "TareaEstructuraDeDatos - Apache NetBeans IDE 25". The left sidebar shows a file tree with "195.8/432.0MB" and icons for files and folders. The main editor window displays the "Estudiante.java" source code. The code defines a class "Estudiante" with private attributes "id", "nombre", and "puntaje", and methods for their manipulation. It also overrides the "toString" method to return a string representation of the object.

```

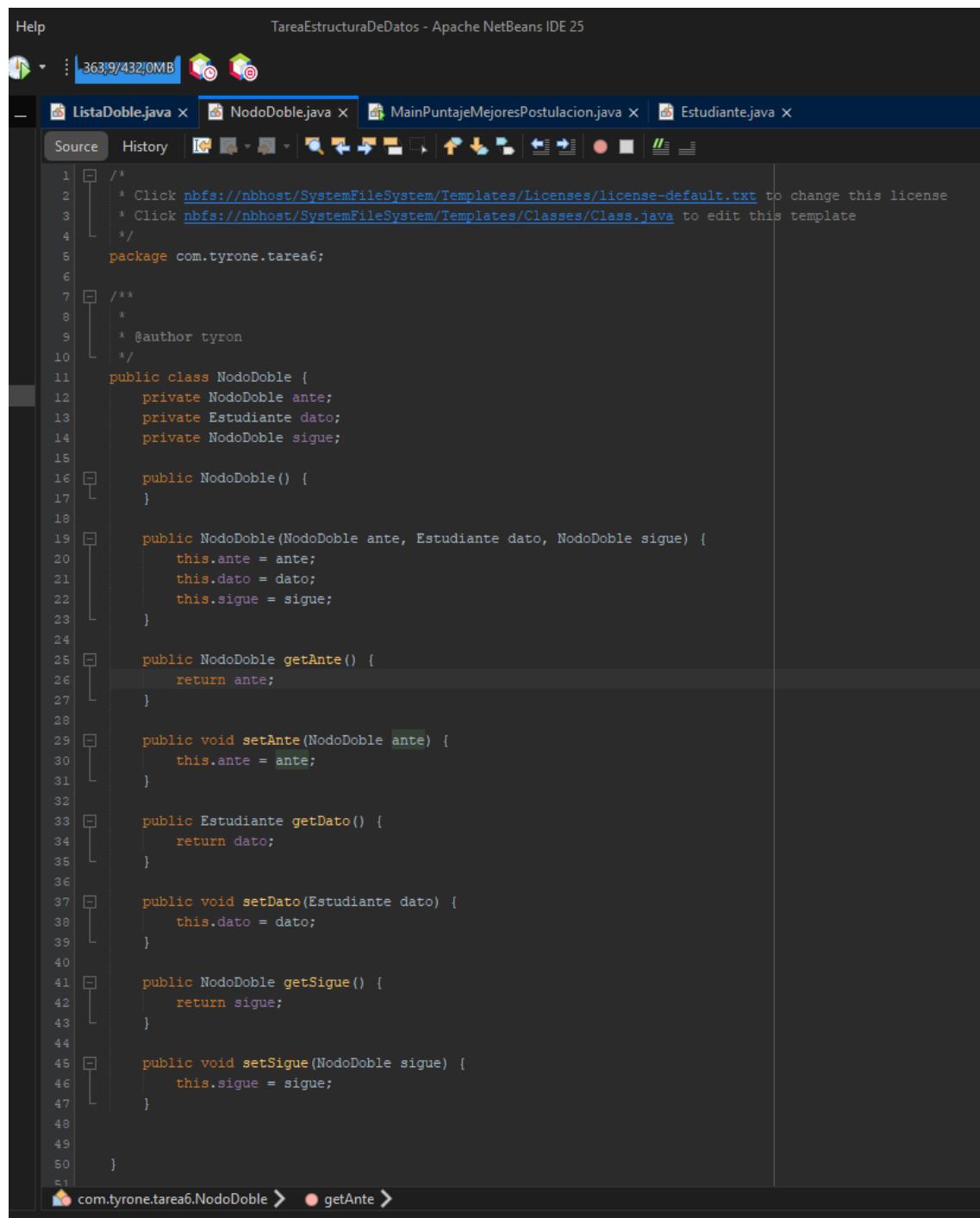
1  /*
2   * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3   * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
4   */
5  package com.tyrone.tarea6;
6
7  /**
8   * @author tyron
9   */
10 public class Estudiante {
11     private int id;
12     private String nombre;
13     private float puntaje;
14
15
16     public Estudiante() {
17     }
18
19
20     public Estudiante(int id, String nombre, float puntaje) {
21         this.id = id++;
22         this.id = id;
23         this.nombre = nombre;
24         this.puntaje = puntaje;
25     }
26
27     public int getId() {
28         return id;
29     }
30
31     public void setId(int id) {
32         this.id = id;
33     }
34
35     public String getNombre() {
36         return nombre;
37     }
38
39     public void setNombre(String nombre) {
40         this.nombre = nombre;
41     }
42
43     public float getPuntaje() {
44         return puntaje;
45     }
46
47     public void setPuntaje(float puntaje) {
48         this.puntaje = puntaje;
49     }
50
51     @Override
52     public String toString() {
53         String nombreLimpio = (nombre.charAt(0) == ' ') ? nombre.substring(1) : nombre;
54         return "(" + "id= " + id + ", nombre= " + nombreLimpio + ", puntaje= " + puntaje + ")";
55     }
56
57 }

```

com.tyrone.tarea6.Estudante > nombre >

Asignatura	Datos del alumno	Fecha
Estructura de Datos	Apellidos: Pilay ***	11/12/2025
	Nombre: Tyrone ****	

Clase NodoDoble



```

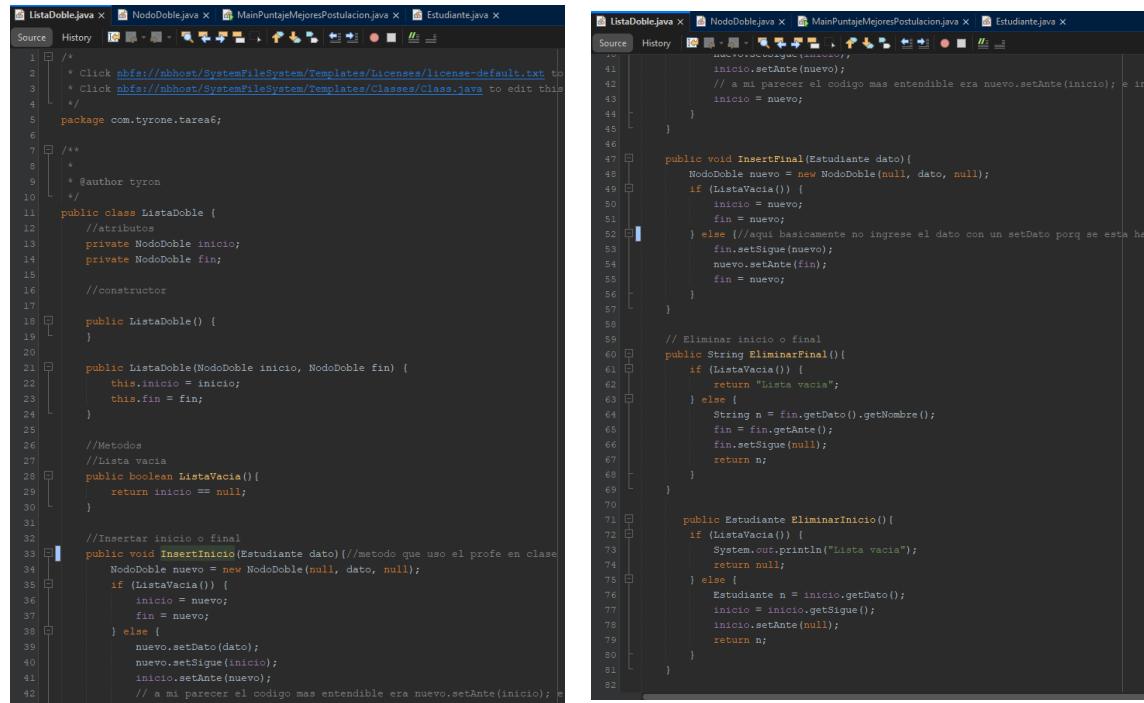
Help TareaEstructuraDeDatos - Apache NetBeans IDE 25
363,9/432,0MB
ListaDoble.java X NodoDoble.java X MainPuntajeMejoresPostulacion.java X Estudiante.java X
Source History
1 /**
2  * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3  * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
4  */
5 package com.tyrone.tarea6;
6
7 /**
8  *
9  * @author tyron
10 */
11 public class NodoDoble {
12     private NodoDoble ante;
13     private Estudiante dato;
14     private NodoDoble sigue;
15
16     public NodoDoble() {
17     }
18
19     public NodoDoble(NodoDoble ante, Estudiante dato, NodoDoble sigue) {
20         this.ante = ante;
21         this.dato = dato;
22         this.sigue = sigue;
23     }
24
25     public NodoDoble getAnte() {
26         return ante;
27     }
28
29     public void setAnte(NodoDoble ante) {
30         this.ante = ante;
31     }
32
33     public Estudiante getData() {
34         return dato;
35     }
36
37     public void setData(Estudiante dato) {
38         this.dato = dato;
39     }
40
41     public NodoDoble getSigue() {
42         return sigue;
43     }
44
45     public void setSigue(NodoDoble sigue) {
46         this.sigue = sigue;
47     }
48
49
50 }

```

com.tyrone.tarea6.NodoDoble > getAnte >

Asignatura	Datos del alumno	Fecha
Estructura de Datos	Apellidos: Pilay *** Nombre: Tyrone ****	11/12/2025

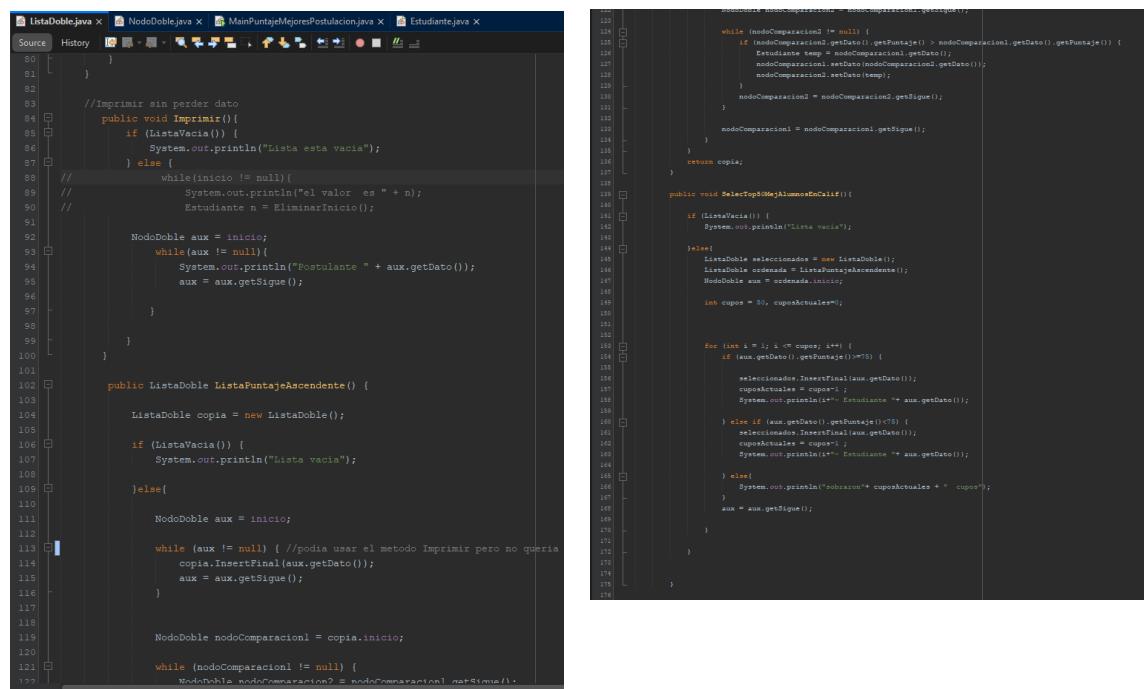
Clase ListaDoble:



```

1  /*
2  * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to
3  * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this
4  */
5  package com.tyrone.tarea6;
6
7  /**
8  *
9  * @author tyron
10 */
11 public class ListaDoble {
12     //atributos
13     private NodoDoble inicio;
14     private NodoDoble fin;
15
16     //constructor
17
18     public ListaDoble() {
19     }
20
21     public ListaDoble(NodoDoble inicio, NodoDoble fin) {
22         this.inicio = inicio;
23         this.fin = fin;
24     }
25
26     //Metodos
27     //Lista vacia
28     public boolean ListaVacia(){
29         return inicio == null;
30     }
31
32     //Insertar inicio o final
33     public void InsertarFinal(Estudiante dato){//metodo que uso el profe en clase
34         NodoDoble nuevo = new NodoDoble(null, dato, null);
35         if (ListaVacia()) {
36             inicio = nuevo;
37             fin = nuevo;
38         } else {
39             nuevo.setdato(dato);
40             nuevo.setSigue(inicio);
41             inicio.setAnte(nuevo);
42             // a mi parecer el codigo mas entendible era nuevo.setAnte(inicio); e
43             nuevo.setAnte(nuevo);
44         }
45     }
46
47     public void InsertarFinal(Estudiante dato){
48         NodoDoble nuevo = new NodoDoble(null, dato, null);
49         if (ListaVacia()) {
50             inicio = nuevo;
51             fin = nuevo;
52         } else { // aqui basicamente no ingrese el dato con un setdato porq se esta ha
53             fin.setSigue(nuevo);
54             nuevo.setAnte(fin);
55             fin = nuevo;
56         }
57     }
58
59     // Eliminar inicio o final
60     public String EliminarFinal(){
61         if (ListaVacia()) {
62             return "Lista vacia";
63         } else {
64             String n = fin.getdato().getNombre();
65             fin = fin.getAnte();
66             fin.setSigue(null);
67             return n;
68         }
69     }
70
71     public Estudiante EliminarInicio(){
72         if (ListaVacia()) {
73             System.out.println("Lista vacia");
74             return null;
75         } else {
76             Estudiante n = inicio.getdato();
77             inicio = inicio.getSigue();
78             inicio.setAnte(null);
79             return n;
80         }
81     }
82 }

```



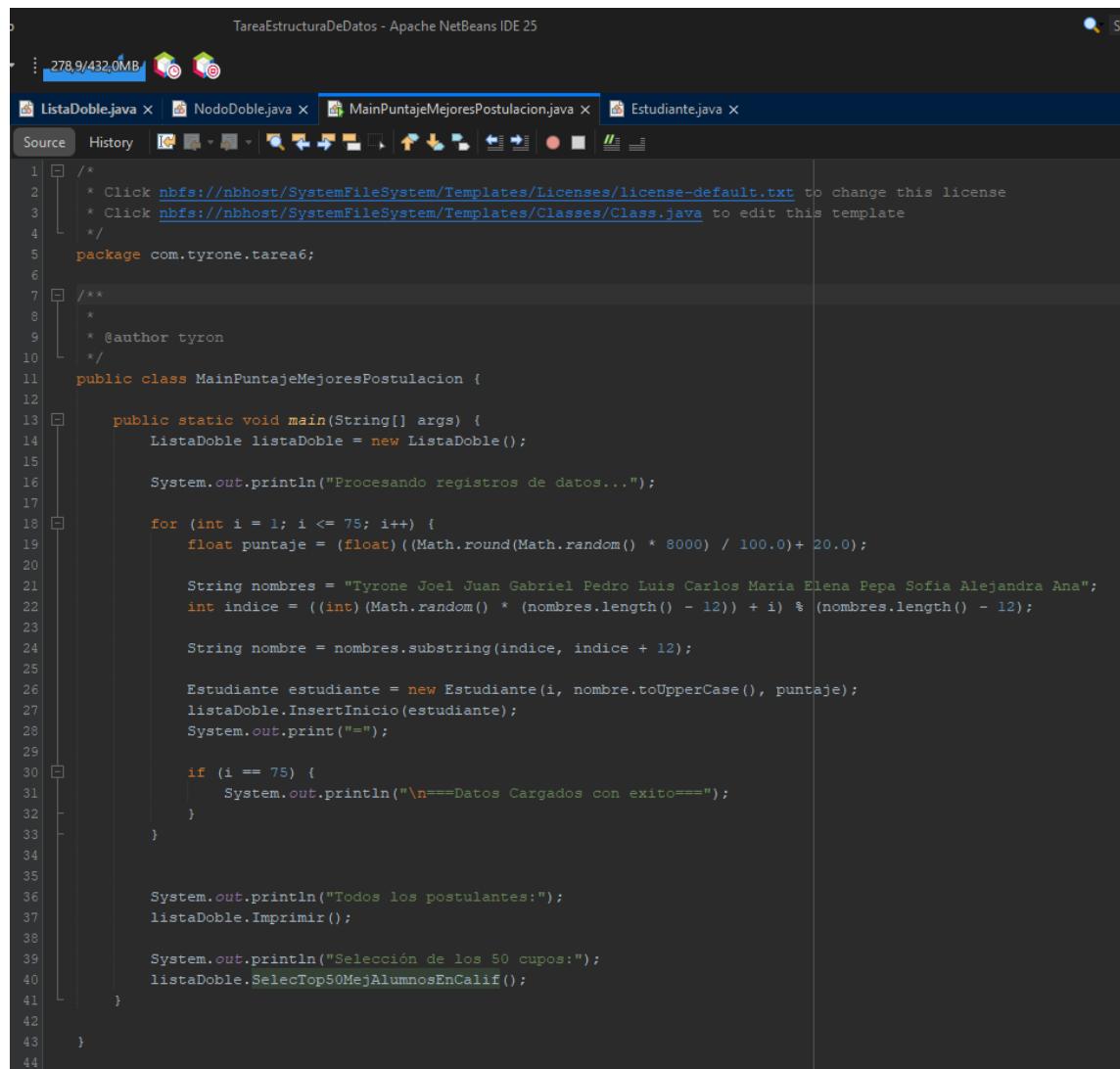
```

1  package com.tyrone.tarea6;
2
3  public class MainPuntajeMejoresPostulacion {
4
5      public static void main(String[] args) {
6
7          //Imprimir sin perder dato
8          public void Imprimir(){
9              if (ListaVacia()){
10                  System.out.println("Lista esta vacia");
11              } else {
12                  while(inicio != null){
13                      System.out.println("el valor es " + n);
14                      Estudiante n = EliminarInicio();
15
16                      NodoDoble aux = inicio;
17                      while(aux != null){
18                          System.out.println("Postulante " + aux.getdato());
19                          aux = aux.getSigue();
20
21                      }
22
23                  }
24
25              }
26
27          }
28
29          public ListaDoble ListaPuntajeAscendente() {
30
31              ListaDoble copia = new ListaDoble();
32
33              if (ListaVacia()){
34                  System.out.println("Lista vacia");
35              }else{
36                  NodoDoble aux = inicio;
37
38                  while (aux != null) { //podia usar el metodo Imprimir pero no queria
39                      copia.InsertarFinal(aux.getdato());
40                      aux = aux.getSigue();
41
42                  }
43
44                  NodoDoble nodoComparacionl = copia.inicio;
45
46                  while (nodoComparacionl != null) {
47                      NodoDoble nodoComparacionr = nodoComparacionl.getSigue();
48
49                      if (nodoComparacionr != null) {
50                          if (nodoComparacionl.getdato().getPuntaje() > nodoComparacionr.getdato().getPuntaje()) {
51                              Estudiante temp = nodoComparacionl.getdato();
52                              nodoComparacionl.setdato(nodoComparacionr.getdato());
53                              nodoComparacionr.setdato(temp);
54
55                          nodoComparacionl = nodoComparacionl.getSigue();
56
57                      }
58
59                  }
60
61                  copia = copia.InsertarFinal(nodoComparacionl.getdato());
62
63                  nodoComparacionl = copia.inicio;
64
65              }
66
67              return copia;
68
69          }
70
71          public void SelectTopNME(ArrayList<Estudiante> seleccionados) {
72
73              if (ListaVacia()){
74                  System.out.println("Lista vacia");
75              } else{
76                  ListaDoble seleccionados = new ListaDoble();
77                  ListaDoble ordenada = ListaPuntajeAscendente();
78                  NodoDoble aux = ordenada.inicio;
79
80                  int cupos = 50, cuposactuales=0;
81
82
83                  for (int i = 1; i <= cupos; i++) {
84                      if (aux.getdato().getPuntaje()>75) {
85
86                          seleccionados.InsertarFinal(aux.getdato());
87                          cuposactuales = cupos-1;
88                          System.out.println(" "+ aux.getdato().getNombre() + " " + aux.getdato());
89
90                      } else if (aux.getdato().getPuntaje()>70) {
91                          seleccionados.InsertarFinal(aux.getdato());
92                          cuposactuales = cupos-1;
93                          System.out.println(" "+ aux.getdato().getNombre() + " " + aux.getdato());
94
95                      } else{
96                          System.out.println("subcupos" + cuposactuales + " " + cupos);
97
98                      }
99
100                     aux = aux.getSigue();
101
102                 }
103
104             }
105
106         }
107
108     }
109
110 }
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176

```

Asignatura	Datos del alumno	Fecha
Estructura de Datos	Apellidos: Pilay ***	11/12/2025
	Nombre: Tyrone ****	

Clase Main:



```

1  /*
2  * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3  * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
4  */
5  package com.tyrone.tarea6;
6
7 /**
8 *
9 * @author tyron
10 */
11 public class MainPuntajeMejoresPostulacion {
12
13     public static void main(String[] args) {
14         ListaDoble listaDoble = new ListaDoble();
15
16         System.out.println("Procesando registros de datos...");
17
18         for (int i = 1; i <= 75; i++) {
19             float puntaje = (float) ((Math.round(Math.random() * 8000) / 100.0) + 20.0);
20
21             String nombres = "Tyrone Joel Juan Gabriel Pedro Luis Carlos Maria Elena Pepa Sofia Alejandra Ana";
22             int indice = ((int) (Math.random() * (nombres.length() - 12)) + i) % (nombres.length() - 12);
23
24             String nombre = nombres.substring(indice, indice + 12);
25
26             Estudiante estudiante = new Estudiante(i, nombre.toUpperCase(), puntaje);
27             listaDoble.InsertInicio(estudiante);
28             System.out.print("==");
29
30             if (i == 75) {
31                 System.out.println("\n==Datos Cargados con exito===");
32             }
33         }
34
35
36         System.out.println("Todos los postulantes:");
37         listaDoble.Imprimir();
38
39         System.out.println("Selección de los 50 cupos:");
40         listaDoble.SelecTop50MejAlumnosEnCalif();
41     }
42
43 }
44

```

Asignatura	Datos del alumno	Fecha
Estructura de Datos	Apellidos: Pilay ***	11/12/2025
	Nombre: Tyrone ****	

Capturas de la ejecución:

```
Output - Run (MainPuntajeMejoresPostulacion)
--- exec:3.1.0:exec (default-cli) @ TarealEstructuraDeDatos ---
Procesando registros de datos...
=====
====Datos Cargados con exito===
Todos los postulantes:
Postulante {id= 75, nombre= PEDRO LUIS C, puntaje= 46.09}
Postulante {id= 74, nombre= UAN GABRIEL , puntaje= 94.98}
Postulante {id= 73, nombre= OEL JUAN GAB, puntaje= 85.88}
Postulante {id= 72, nombre= IA ALEJANDRA, puntaje= 66.26}
Postulante {id= 71, nombre= OFIA ALEJAND, puntaje= 58.53}
Postulante {id= 70, nombre= ALEJANDRA AN, puntaje= 84.23}
Postulante {id= 69, nombre= TYRONE JOEL , puntaje= 65.68}
Postulante {id= 68, nombre= RO LUIS CARL, puntaje= 77.92}
Postulante {id= 67, nombre= LUIS CARLOS, puntaje= 64.42}
Postulante {id= 66, nombre= SOFIA ALEJAN, puntaje= 35.58}
Postulante {id= 65, nombre= LENA PEPA SO, puntaje= 38.65}
Postulante {id= 64, nombre= L JUAN GABRI, puntaje= 98.19}
Postulante {id= 63, nombre= A ALEJANDRA , puntaje= 74.67}
Postulante {id= 62, nombre= LUIS CARLOS, puntaje= 29.71}
Postulante {id= 61, nombre= LOS MARIA EL, puntaje= 34.51}
Postulante {id= 60, nombre= AN GABRIEL P, puntaje= 84.22}
Postulante {id= 59, nombre= OFIA ALEJAND, puntaje= 38.8}
Postulante {id= 58, nombre= TYRONE JOEL , puntaje= 68.27}
Postulante {id= 57, nombre= ABRIEL PEDRO, puntaje= 67.72}
Postulante {id= 56, nombre= NE JOEL JUAN, puntaje= 31.32}
Postulante {id= 55, nombre= ALEJANDRA A, puntaje= 62.14}
Postulante {id= 54, nombre= RIA ELENA PE, puntaje= 30.1}
Postulante {id= 53, nombre= N GABRIEL PE, puntaje= 78.56}
Postulante {id= 52, nombre= ARIA ELENA P, puntaje= 42.5}
Postulante {id= 51, nombre= CARLOS MARIA, puntaje= 40.1}
Postulante {id= 50, nombre= TYRONE JOEL , puntaje= 34.5}
Postulante {id= 49, nombre= AN GABRIEL P, puntaje= 32.36}
Postulante {id= 48, nombre= ALEJANDRA A, puntaje= 88.42}
Postulante {id= 47, nombre= UIS CARLOS M, puntaje= 36.59}
Postulante {id= 46, nombre= DRO LUIS CAR, puntaje= 23.54}
Postulante {id= 45, nombre= OFIA ALEJAND, puntaje= 73.98}
Postulante {id= 44, nombre= AN GABRIEL P, puntaje= 40.18}
Postulante {id= 43, nombre= NA PEPA SOFI, puntaje= 33.9}
Postulante {id= 42, nombre= ARIA ELENA P, puntaje= 35.79}
Postulante {id= 41, nombre= LUIS CARLOS , puntaje= 94.86}
Postulante {id= 40, nombre= SOFIA ALEJAN, puntaje= 96.98}
Postulante {id= 39, nombre= OFIA ALEJAND, puntaje= 21.23}
Postulante {id= 38, nombre= SOFIA ALEJAN, puntaje= 23.51}
Postulante {id= 37, nombre= RIA ELENA PE, puntaje= 89.75}
Postulante {id= 36, nombre= LENA PEPA SO, puntaje= 70.95}
Postulante {id= 35, nombre= PEDRO LUIS C, puntaje= 48.88}
Postulante {id= 34, nombre= FIA ALEJANDR, puntaje= 85.61}
Postulante {id= 33, nombre= ELENA PEPA , puntaje= 60.28}
Postulante {id= 32, nombre= SOFIA ALEJA, puntaje= 91.21}
Postulante {id= 31, nombre= LUIS CARLOS , puntaje= 55.54}
```

Facultad de Ciencias Informáticas | Carrera de Tecnologías de la Información

Asignatura	Datos del alumno	Fecha
Estructura de Datos	Apellidos: Pilay ***	11/12/2025
	Nombre: Tyrone ****	

```

Output - Run (MainPuntajeMejoresPostulacion)
Postulante {id= 32, nombre= SORIA ALEJA, puntaje= 91.21}
Postulante {id= 31, nombre= LUIS CARLOS , puntaje= 55.54}
Postulante {id= 30, nombre= LENA PEPA SO, puntaje= 88.44}
Postulante {id= 29, nombre= UAN GABRIEL , puntaje= 82.66}
Postulante {id= 28, nombre= NA PEPA SOFI, puntaje= 29.5}
Postulante {id= 27, nombre= PEDRO LUIS , puntaje= 95.63}
Postulante {id= 26, nombre= ARIA ELENA P, puntaje= 66.68}
Postulante {id= 25, nombre= JUAN GABRIE, puntaje= 56.57}
Postulante {id= 24, nombre= JOEL JUAN GA, puntaje= 89.39}
Postulante {id= 23, nombre= ONE JOEL JUA, puntaje= 94.1}
Postulante {id= 22, nombre= BRIEL PEDRO , puntaje= 96.56}
Postulante {id= 21, nombre= DRO LUIS CAR, puntaje= 32.54}
Postulante {id= 20, nombre= UAN GABRIEL , puntaje= 61.46}
Postulante {id= 19, nombre= ELENA PEPA , puntaje= 35.68}
Postulante {id= 18, nombre= IS CARLOS MA, puntaje= 33.46}
Postulante {id= 17, nombre= A SOFIA ALEJ, puntaje= 34.95}
Postulante {id= 16, nombre= YRONE JOEL J, puntaje= 46.46}
Postulante {id= 15, nombre= S MARIA ELEN, puntaje= 75.88}
Postulante {id= 14, nombre= OFIA ALEJAND, puntaje= 86.18}
Postulante {id= 13, nombre= O LUIS CARLO, puntaje= 77.83}
Postulante {id= 12, nombre= RIA ELENA PE, puntaje= 28.71}
Postulante {id= 11, nombre= EDRO LUIS CA, puntaje= 80.95}
Postulante {id= 10, nombre= CARLOS MARI, puntaje= 20.08}
Postulante {id= 9, nombre= EL JUAN GABR, puntaje= 50.68}
Postulante {id= 8, nombre= ALEJANDRA A, puntaje= 21.99}
Postulante {id= 7, nombre= ARIA ELENA P, puntaje= 43.8}
Postulante {id= 6, nombre= PEPA SOFIA , puntaje= 81.17}
Postulante {id= 5, nombre= CARLOS MARI, puntaje= 51.46}
Postulante {id= 4, nombre= CARLOS MARI, puntaje= 49.97}
Postulante {id= 3, nombre= E JOEL JUAN , puntaje= 43.41}
Postulante {id= 2, nombre= RONE JOEL JU, puntaje= 33.49}
Postulante {id= 1, nombre= EPA SOFIA AL, puntaje= 84.03}
Selección de los 50 cupos:
1~ Estudiante {id= 64, nombre= L JUAN GABRI, puntaje= 98.19}
2~ Estudiante {id= 40, nombre= SOFIA ALEJAN, puntaje= 96.98}
3~ Estudiante {id= 22, nombre= BRIEL PEDRO , puntaje= 96.56}
4~ Estudiante {id= 27, nombre= PEDRO LUIS , puntaje= 95.63}
5~ Estudiante {id= 74, nombre= UAN GABRIEL , puntaje= 94.98}
6~ Estudiante {id= 41, nombre= LUIS CARLOS , puntaje= 94.86}
7~ Estudiante {id= 23, nombre= ONE JOEL JUA, puntaje= 94.1}
8~ Estudiante {id= 32, nombre= SOFIA ALEJA, puntaje= 91.21}
9~ Estudiante {id= 37, nombre= RIA ELENA PE, puntaje= 89.75}
10~ Estudiante {id= 24, nombre= JOEL JUAN GA, puntaje= 89.39}
11~ Estudiante {id= 30, nombre= LENA PEPA SO, puntaje= 88.44}
12~ Estudiante {id= 48, nombre= ALEJANDRA A, puntaje= 88.42}
13~ Estudiante {id= 14, nombre= OFIA ALEJAND, puntaje= 86.18}
14~ Estudiante {id= 73, nombre= OEL JUAN GAB, puntaje= 85.88}
15~ Estudiante {id= 34, nombre= FIA ALEJANDR, puntaje= 85.61}
16~ Estudiante {id= 70, nombre= ALEJANDRA AN, puntaje= 84.23}
17~ Estudiante {id= 60, nombre= AN GABRIEL P, puntaje= 84.22}
18~ Estudiante {id= 1, nombre= EPA SOFIA AL, puntaje= 84.03}
19~ Estudiante {id= 29, nombre= UAN GABRIEL , puntaje= 82.66}

```

Facultad de Ciencias Informáticas | Carrera de Tecnologías de la Información

Asignatura	Datos del alumno	Fecha
Estructura de Datos	Apellidos: Pilay ***	11/12/2025
	Nombre: Tyrone ****	

```
Output - Run (MainPuntajeMejoresPostulacion)

7~ Estudiante {id= 23, nombre= ONE JOEL JUA, puntaje= 94.1}
8~ Estudiante {id= 32, nombre= SOFIA ALEJA, puntaje= 91.21}
9~ Estudiante {id= 37, nombre= RIA ELENA PE, puntaje= 89.75}
10~ Estudiante {id= 24, nombre= JOEL JUAN GA, puntaje= 89.39}
11~ Estudiante {id= 30, nombre= LENA PEPA SO, puntaje= 88.44}
12~ Estudiante {id= 48, nombre= ALEJANDRA A, puntaje= 88.42}
13~ Estudiante {id= 14, nombre= OFIA ALEJAND, puntaje= 86.18}
14~ Estudiante {id= 73, nombre= OEL JUAN GAB, puntaje= 85.88}
15~ Estudiante {id= 34, nombre= FIA ALEJANDR, puntaje= 85.61}
16~ Estudiante {id= 70, nombre= ALEJANDRA AN, puntaje= 84.23}
17~ Estudiante {id= 60, nombre= AN GABRIEL P, puntaje= 84.22}
18~ Estudiante {id= 1, nombre= EPA SOFIA AL, puntaje= 84.03}
19~ Estudiante {id= 29, nombre= UAN GABRIEL , puntaje= 82.66}
20~ Estudiante {id= 6, nombre= PEPA SOFIA , puntaje= 81.17}
21~ Estudiante {id= 11, nombre= EDRO LUIS CA, puntaje= 80.95}
22~ Estudiante {id= 53, nombre= N GABRIEL PE, puntaje= 78.56}
23~ Estudiante {id= 68, nombre= RO LUIS CARL, puntaje= 77.92}
24~ Estudiante {id= 13, nombre= O LUIS CARLO, puntaje= 77.83}
25~ Estudiante {id= 15, nombre= S MARIA ELEN, puntaje= 75.88}
26~ Estudiante {id= 63, nombre= A ALEJANDRA , puntaje= 74.67}
27~ Estudiante {id= 45, nombre= OFIA ALEJAND, puntaje= 73.98}
28~ Estudiante {id= 36, nombre= LENA PEPA SO, puntaje= 70.95}
29~ Estudiante {id= 58, nombre= TYRONE JOEL , puntaje= 68.27}
30~ Estudiante {id= 57, nombre= ABRIEL PEDRO, puntaje= 67.72}
31~ Estudiante {id= 26, nombre= ARIA ELENA P, puntaje= 66.68}
32~ Estudiante {id= 72, nombre= IA ALEJANDRA, puntaje= 66.26}
33~ Estudiante {id= 69, nombre= TYRONE JOEL , puntaje= 65.68}
34~ Estudiante {id= 67, nombre= LUIS CARLOS, puntaje= 64.42}
35~ Estudiante {id= 55, nombre= ALEJANDRA A, puntaje= 62.14}
36~ Estudiante {id= 20, nombre= UAN GABRIEL , puntaje= 61.46}
37~ Estudiante {id= 33, nombre= ELENA PEPA , puntaje= 60.28}
38~ Estudiante {id= 71, nombre= OFIA ALEJAND, puntaje= 58.53}
39~ Estudiante {id= 25, nombre= JUAN GABRIE, puntaje= 56.57}
40~ Estudiante {id= 31, nombre= LUIS CARLOS , puntaje= 55.54}
41~ Estudiante {id= 5, nombre= CARLOS MARI, puntaje= 51.46}
42~ Estudiante {id= 9, nombre= EL JUAN GABR, puntaje= 50.68}
43~ Estudiante {id= 4, nombre= CARLOS MARI, puntaje= 49.97}
44~ Estudiante {id= 35, nombre= PEDRO LUIS C, puntaje= 48.88}
45~ Estudiante {id= 16, nombre= YRONE JOEL J, puntaje= 46.46}
46~ Estudiante {id= 75, nombre= PEDRO LUIS C, puntaje= 46.09}
47~ Estudiante {id= 7, nombre= ARIA ELENA P, puntaje= 43.8}
48~ Estudiante {id= 3, nombre= E JOEL JUAN , puntaje= 43.41}
49~ Estudiante {id= 52, nombre= ARIA ELENA P, puntaje= 42.5}
50~ Estudiante {id= 44, nombre= AN GABRIEL P, puntaje= 40.18}

-----
BUILD SUCCESS

-----
Total time: 4.020 s
Finished at: 2025-12-10T22:14:27-05:00
-----
```