



# Introducción a la Programación Orientada a Objetos

DCIC – UNS

25 de noviembre de 2023



Examen Promoción 10:30 hs.

| Apellido y Nombre | N°Libreta | e-mail |
|-------------------|-----------|--------|
|-------------------|-----------|--------|

- 1- Defina los conceptos de variable polimórfica y asignación polimórfica. ¿Qué restricciones establece el chequeo de tipos de JAVA al polimorfismo?
- 2- Defina Ligadura Dinámica de Código.
- 3- Dado el siguiente diagrama de clases:

| ColeccionGenerica                                                                                                                                                                     |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| << Atributos de instancia >><br>c: [ ] Elemento<br>cant: entero                                                                                                                       |
| << constructor >><br>ColeccionGenerica(n: entero)                                                                                                                                     |
| << Comandos >><br>insertar(e: Elemento)                                                                                                                                               |
| << Consultas >><br>obtener(p: entero): Elemento<br>cantElementos(): entero<br>nConsecutivosEquivalentes(n: entero): boolean<br>suplementar(col: ColeccionGenerica): ColeccionGenerica |

| Punto                                                                                                                         |
|-------------------------------------------------------------------------------------------------------------------------------|
| << Atributos de instancia >><br>x: real<br>y: real                                                                            |
| << constructor >><br>Punto(px, py: real)                                                                                      |
| << Consultas >><br>obtenerX(): real<br>obtenerY(): real<br>suplementar(e: Elemento): Elemento<br>equals(e: Elemento): boolean |

a) Implemente la clase ColeccionGenerica considerando las siguientes funcionalidades y responsabilidades:

- **ColeccionGenerica(n: entero)** crea una Coleccion de  $n$  componentes y la liga a  $c$ . Requiere  $n$  mayor a 0.
- **insertar(e: Elemento)** inserta el elemento  $e$  en la primera posición libre.
- **nConsecutivosEquivalentes(n: entero): boolean** retorne true si en la colección hay al menos una secuencia de exactamente  $n$  elementos consecutivos equivalentes. Requiere  $n > 1$ .
- **suplementar(col: ColeccionGenerica): ColeccionGenerica** retorna una nueva ColeccionGenerica que contendrá los elementos que resulten de suplementar los elementos de la colección que recibe el mensaje con los elementos de la colección  $col$  pasada por parámetro. Por ejemplo: el elemento ligado a la primera posición de la colección resultante será aquel que se obtenga de suplementar el elemento ligado a la primera posición de la colección que recibe el mensaje con el elemento ligado a la primera posición de  $col$ , el elemento ligado a la segunda posición de la colección resultante será aquel que se obtenga de suplementar el elemento ligado a la segunda posición de la colección que recibe el mensaje con el elemento ligado a la segunda posición de  $col$ , y así siguiendo. En caso de que las colecciones no tengan la misma cantidad de elementos almacenados, se completa la colección resultante con los elementos de la colección que tiene más elementos ligados y están en posiciones donde la otra colección no tiene elementos ligados. Esto es, suponga que la colección que recibe el mensaje tiene los siguientes elementos  $[e1, e2, null]$  y la colección pasada por parámetro tiene los siguientes elementos  $[e3, e4, e5, null, null]$ , la colección resultante tendrá los siguientes elementos  $[suplemento(e1, e3), suplemento(e2, e4), e5]$ .

b) Implemente la clase Elemento con él o los servicios que requiere la clase ColeccionGenerica.

c) Implemente la clase Punto que extiende a Elemento, considerando que el suplemento de  $p1$  con  $p2$  es el punto  $p3$  donde el valor  $x$  de  $p3$  resulta de sumar los valores  $x$  de  $p1$  y  $p2$ , y el valor  $y$  de  $p3$  resulta de



## Introducción a la Programación Orientada a Objetos

DCIC – UNS

25 de noviembre de 2023



Examen Promoción 10:30 hs.

sumar los valores  $y$  de  $p1$  y  $p2$ . Adicionalmente, un punto es equivalente a otro, si tienen los mismos valores en  $x$  e  $y$ .

- d) Implemente una clase `ColeccionPuntos` que extienda a `ColeccionGenerica` e implemente un método **`mismosNullsQueDifCero(): boolean`** que retorne true si en la colección hay tantas componentes no ligadas como puntos ligados cuya diferencia entre sus valores  $x$  e  $y$  es cero. La clase cliente de `ColeccionPuntos` se compromete a garantizar que cuando se inserten los valores de una instancia de `ColeccionPuntos`, el parámetro va a ser un objeto de clase `Punto`.
- e) Implemente un tester que permita verificar el servicio **`mismosNullsQueDifCero(): boolean`**.

Envíe el proyecto empaquetado a **`ipoo.dic@gmail.com`**, asegúrese de que su nombre figura en el proyecto y que el nombre del archivo tenga su nombre de la siguiente manera: **`apellido-nombre.zip`**