LABORATORIO N° 1
CLASE CIUDAD

Dada la siguiente definición de la clase *Ciudad*, parcialmente implementada en Java (en **negrita** se listan los métodos sin implementar):

Ciudad
<< Atributos de instancia >> CP: entero poblacion: entero superficie: real
<< Constructores >> Ciudad (c: entero) Ciudad (c: entero, p: entero, s: entero) **<< Comandos >>** establecerPoblacion (pob: entero) **establecerSuperficie (sup: real)** **aumentarPoblacion (cre: entero)** **<< Consultas >>** obtenerCP(): entero **obtenerPoblacion(): entero** **obtenerSuperficie(): real** **obtenerDensidad(): real** **toString(): String**

1. Completar la implementación de la clase *Ciudad*, teniendo en cuenta la siguiente especificación para los métodos no triviales:

   - **aumentarPoblacion(cre: entero)**: Se incrementa la población de acuerdo al valor recibido por parámetro. Puede recibir un valor negativo, en ese caso la población se reduce. Tener en cuenta, en este último caso, que nunca una población puede ser negativa. Es responsabilidad de la clase *Ciudad* controlarlo, y en caso que se intente restar una cantidad mayor a la población actual, se debe fijar la misma en cero.

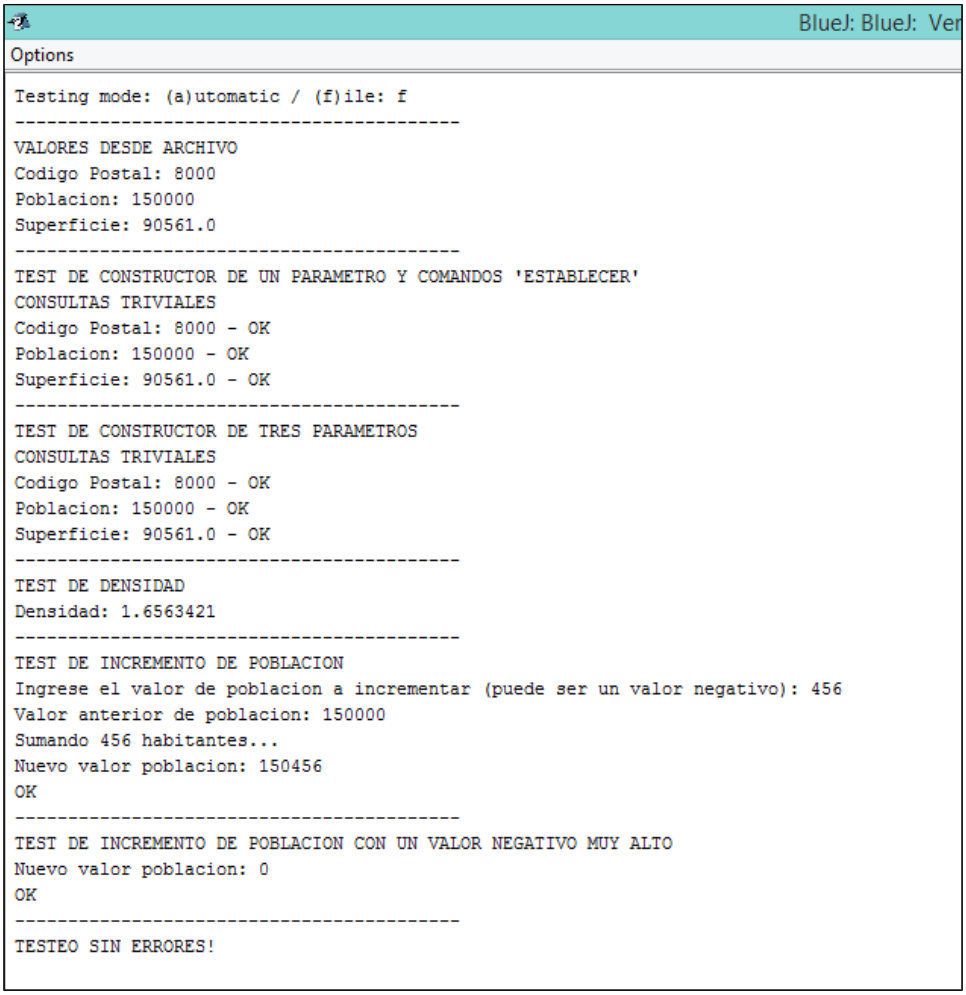
   - **obtenerDdensidad(): real**: Calcula la densidad como el cociente entre la población y la superficie. Requiere que la superficie sea distinta de cero, es decir que es responsabilidad de la clase cliente (en nuestro caso de trabajo, sería la clase *Tester*) realizar dicho control.

   - **toString(): String**: Obtiene una cadena de caracteres representando el estado interno del objeto. Ejemplo: “Codigo Postal: 8000 - Poblacion: 300000 habitantes - Superficie: 10 km2”

2. Estudiar y completar la implementación parcial de la clase Tester dada. Para ello se deberá:

a. Analizar la clase Tester dada, estudiando sus dos modos alternativos de funcionamiento (automático y desde archivo).

b. Completar el código incluyendo casos de prueba para los métodos triviales. Esto es, chequear que los métodos `obtenerCP`, `obtenerPoblacion` y `obtenerSuperficie` devuelvan los valores correctos.



Resultado ejecución exitosa.

Tips:

- Usar siempre los archivos base que distribuimos a los efectos del laboratorio, que ya contienen código pre-implementado. No iniciar la codificación desde cero.
- Antes de iniciar la codificación, leer completamente el enunciado y estudiar detalladamente el código fuente distribuido.
- La implementación de clase Ciudad debe respetar las especificaciones de nombres (con las mayúsculas y minúsculas correspondientes) y parámetros, ya que los mismos son asumidos de ese modo por la clase *Tester* (el cliente), y el mismo debe ejecutarse exitosamente.
- Analizar la clase *Tester* distribuida. Allí se muestran técnicas de prueba a incorporar para futuras construcciones de tests propios.
- El uso de archivos para el testeo es un hábito que se debe incorporar gradualmente, recuerden siempre que la forma en la que se lee el archivo depende cómo se encuentren estructurados los datos en el mismo.