Reflexión y Revisión DPOO Taller 1

José Cristóbal Arroyo 202011404 - Juan Alegría 202011282

February 2021

Tipos Básicos de Java

1. ¿Cuáles tipos básicos de Java se utilizaron en el taller y qué se puede representar con cada uno?

En el taller se utilizaron los siguientes tipos básicos de Java:

- boolean: utilizado para representar si un libro tiene portada, si una categoría es de ficción, si hay un libro de un autor en una categoría, si hay un autor en varias categorías, o si existe un archivo en la carpeta data.
- double: utilizado para representar la calificación promedio de los libros en una categoría o catálogo, y sumas/promedios generales,
- integer: utilizado para contar la cantidad de libros al calcular el promedio o al buscar aquellos sin portada

null

2. ¿A qué elemento de Python se parece el valor null de Java?

El valor null de Java es parecido al None de Python, ya que ambos son valores especiales para indicar que algo es nulo.

3. ¿Qué diferencias hay entre el valor null de Java y el valor que respondió para la pregunta anterior?

La diferencia es que null es una instancia de nada (no es un objeto), mientras que None es un objeto sin métodos que denota falta de valor, se podría convertir a otras representaciones de datos, y evita errores de programación como NullPointerExceptions en Java.

Arreglos y listas

4. ¿Cómo puede averiguar el tamaño de un arreglo?

Si tenemos un arreglo llamado array podríamos utilizar `array.length;` para averiguar su tamaño.

5. ¿Cómo puede averiguar el tamaño de una lista?

Para averiguar el tamaño de una lista, podríamos utilizar el método size(). Ejemplo: `arrlist.size();`

6. ¿Cuál es la principal restricción de un arreglo con respecto a las listas?

Los arreglos normales tienen un tamaño fijo, esto implica que si quisieramos cambiar el tamaño tendríamos que hacerlo desde el código fuente, sin embargo las listas (`ArrayList<>`) poseen una estructura que les permite cambiar el tamaño en tiempo de ejecución.

7. ¿Cómo se especifica el tipo de una lista de números enteros?

Para especificar el tipo en una lista de tamaño variable, es necesario ingresar el tipo dentro de los corchetes. Para ello se declara de la siguiente manera: `ArrayList<[TIPO]> [NOMBRE];`

8. Haga una lista de los métodos de los arreglos que le hayan sido de utilidad para este taller. ¿Algún comentario?

- `.addAll()`: Fue útil para añadir todos los nuevos elementos de un arreglo a otro
- `.contains()`: Su utilidad fue encontrar si existe un elemento o no dentro del arreglo.
- `.length`: Aunque este no es un método sino un atributo, fue de utilidad para encontrar el tamaño del arreglo.

9. Haga una lista de los métodos de las listas que le hayan sido de utilidad para este taller. ¿Algún comentario?

- `.contains()`: Al igual que el anterior, su utilidad fue encontrar si existe un elemento o no dentro del arreglo.
- `.get()`: Ya que la lista es una clase, no podemos acceder a los datos de forma directa a través de los corchetes. Por ello se utiliza el método `.get([ITEM])`

- `.add()`: Este método añade un nuevo elemento a la lista.
- `.size()`: Este método se utiliza para encontrar el tamaño de la lista.

Instrucciones iterativas en Java

10. ¿Qué diferencias hay entre las tres estructuras para construir instrucciones iterativas? (`while`, `for` y `for-each`)

La instrucción `while` permite iterar indefinidamente mientras se cumpla una condición. En el caso del taller, fue utilizado para leer el archivo, puesto que no se conoce el tamaño de este y debe recorrerse hasta que termine.
Ejemplo: `while ([CONDICIÓN]){`
`[INSTRUCCIÓN]`
}

La instrucción `for` permite iterar un número definido de veces mientras se cumpla una condición. Para ello, se declara una variable con valor inicial, una condición para que el ciclo siga vivo, y un incremento que se ejecuta al final de cada iteración.
Ejemplo: `for ([TIPO][VARIABLE]; [CONDICIÓN]; [INCREMENTO]){ [INSTRUCCIÓN` }

La instrucción `for-each` permite iterar una estructura de datos y todos los elementos dentro. En este caso, se itera hasta que no existan más elementos en la estructura. Para ello, se declara una variable del tipo que contenga la estructura.
Ejemplo: `for ([TIPO][VARIABLE] : [ESTRUCTURA]{ [INSTRUCCIÓN }`

Cada una de las 3 instrucciones permiten recorrer estructuras y repetir instrucciones de una forma distinta. En algunos casos es mucho más eficiente una u otra según el caso.