IT BOARDING

BOOTCAMP

Programación Java

Introducción al lenguaje Java

// Práctica integradora

Objetivo

El objetivo de esta guía práctica es que podamos afianzar y profundizar los conceptos sobre encapsulamiento, clases y objetos. Para esto vamos a plantear una serie de ejercicios simples que nos permitirán repasar los temas que estudiamos.

¿Are you ready? 

Ejercicio 1

Crea una clase CuentaCorriente, con los métodos: ingreso, egreso, reintegro y transferencia. La clase contendrá un constructor por defecto, uno con parámetros y otro que recibirá una CuentaCorriente de la cual copiará todos sus datos. Además se deben desarrollar los métodos de acceso (para cada variable de instancia de la clase, desarrollar un método set y otro get. Por ejemplo: si la clase tiene una variable double saldo, sus métodos de acceso serán: **double getSaldo() y void setSaldo(double s)**. Los prototipos de los métodos y constructores deberán ser discutidos y diseñados en equipo.

Ejercicio 2

Crea una clase Contador (sí, contador de programación) con métodos que permitan incrementar y decrementar su valor. La clase contendrá un constructor por defecto, un constructor con parámetros, un constructor copia y los *setters* y *getters* (métodos de acceso) que corresponda.

Ejercicio 3

- Crea una clase Libro con los métodos: préstamo, devolución y toString, cuyo prototipo debe ser: public String toString(). Este método debe retornar una cadena que represente al objeto. Por ejemplo: si la clase tiene los atributos: título, isbn y autor, una cadena que represente a un libro podría ser: “Harry Potter, 9780545582889, Rowling, J. K.”. La clase contendrá un constructor por defecto, un constructor con parámetros y los métodos de acceso.
- Agregar la línea @Override justo arriba del encabezado del método toString. Luego, cambiar el nombre del método por: tostring (todo en minúscula). ¿Qué sucede?

Ejercicio 4

Crea una clase Fraccion con métodos necesarios para sumar, restar, multiplicar y dividir fracciones. Todos los métodos deben estar sobrecargados de modo que también puedan usarse para operar entre fracciones y números enteros (por ejemplo: ⅜+2 o ⅝*4).

Ejercicio 5

Crea una clase Fecha. La clase contendrá además de los constructores que consideres adecuados, métodos de acceso y el método toString, tal como lo explicamos en el ejercicio anterior, un método para comprobar si la fecha es correcta y otro para sumarle un día al valor actual de la fecha. Se debe investigar y utilizar la clase GregorianCalendar para implementar los métodos y constructores de Fecha.

Ejercicio 6

Partiendo de la clase StringUtil (que usamos para resolver Radix Sort), agregar los siguientes métodos estáticos:

- public static String rpad(String s,char c,int n); idem lpad, pero agregando caracteres a la derecha.

- public static String ltrim(String s); Retorna una cadena idéntica a s pero sin espacios a la izquierda.
- public static String rtrim(String s); idem ltrim, pero sin espacios a la derecha.
- public static String trim(String s); idem lpad, pero sin espacios a derecha ni izquierda.
- public static int indexOfN(String s,char c,int n); Retorna la posición de la n-ésima ocurrencia del carácter c dentro de s, o -1 si s no contiene a c. Por ejemplo, si s = “John|Paul|George|Ringo”, c = ‘|’ y n=2, la función debe retornar la posicióon de la segunda ocurrencia del carácter ‘|’ (pipe) dentro de la cadena s. Que, en este caso, es: 9.