Tarea Java #8: Clase Administradora de Objetos

* Nombre de Archivo Java: ***ClaseAdministradora.java***

* Ejemplo Nombre Archivo Comprimido:

GutierrezAvilesLuisJ8TSOA0413B.zip

Objetivo

Abstraer la gestión de información relacionada dentro de una clase que oculte la información.

Descripción

Implementar una clase administradora de objetos Materia con sus operaciones ABC.

Requerimientos Generales

1. Investigar sobre la tabla Hash (dispersión) Hashtable de Java el uso de ella, sus métodos get, put y remove;
2. Implementar un programa que utilice una ClaseAdministradora para escribir en una tabla que maneje registros;
3. Los campos de la tabla sean: id_secuencial, clave_materia, nombre_materia
4. Desde main se solicite registrar 4 tuplas que al imprimirse se vean de la forma:

id_secuencial	clave	nombre_materia
1	CC317	Compiladores
2	CC318	Taller de Compiladores
3	CC319	Sistemas Operativos Avanzados
4	CC320	Taller de Sistemas Operativos Avanzados

5. Los métodos sean públicos;
6. Dicha ClaseAdministradora que contenga lo siguiente:
   a) un atributo tipo Hashtable y otro contador
   b) un método que agrega tupla recibiendo 2 parámetros: clave y nombre_materia y regrese el id_secuencial
   c) el método anterior agregue un objeto tipo Materia en la tabla (la llave del método put de Hashtable sea el id secuencial envuelto como objeto)
   d) el objeto tipo Materia no contenga el id_secuencial

e) un método que obtenga una Materia de la tabla según parámetro id_secuencial y devuelva el objeto Materia
f) un método que elimine una tupla según parámetro id_secuencial.
g) implementar un método que imprima en consola todas las tuplas de la tabla, no importa que sea en desorden, pero considerar que es posible que se hayan eliminado tuplas, de modo que la extracción no puede ser directa con un contador, sino usando un método de la Hashtable, el cual devuelve una Enumeration (u otro que devuelva un Iterator) con las llaves existentes
h) el método que muestre todas las tuplas, las muestre una por línea como se muestra en el punto 4.
i) cada que se agregue ó elimine una tupla, imprimir el contenido actual de la tabla, solicitando la impresión desde el mismo método invocado
j) evitar impresiones a consola desde la clase administradora y cualquiera de sus clases componentes; lo anterior a excepción de lo necesario para el método que muestra las tuplas

7. En main elaborar un menú que permita el registro de una materia, eliminar una materia, obtener los datos de una materia, imprimir materias o salir.

Requerimientos y Restricciones Complementarios

A. Todos los identificadores de variables y constantes tengan nombres claros y significativos de modo que sea fácil saber qué dato almacenarán
B. Todo identificador debe ser necesario para la operación del programa
C. Cumplir con las convenciones para nomenclatura de variables y constantes explicados en clase según convenciones Java
D. Todo valor asignado a un identificador debe ser necesario para la operación de alguna parte del programa
E. A excepción de las constantes 0 y 1, los cómputos utilicen únicamente variables y constantes simbólicas
F. A excepción de que un requerimiento lo solicite, no incluir la redacción de los requerimientos en el código fuente
G. Las líneas de código fuente sean máximo de 100 caracteres
H. Código fuente indicando como comentario en la parte superior de cada archivo: nombre del(la) alumn@, sección y no. de tarea
I. Cumplir con las convenciones sobre indentación de código fuente
J. Código fuente libre de instrucciones anuladas mediante comentarios
K. El código fuente libre de errores de compilación y advertencias

L. Código fuente libre de comentarios o impresiones de pantalla que describan el funcionamiento del código
M. Evitar comparar banderas contra constante numérica o simbólica
N. Toda instrucción ejecutada sea necesaria según los datos de entrada; por ejemplo, en un programa que validara una fecha en base a día, mes y año, si se trata del mes de Enero, sería inapropiado comprobar si se trata de un año bisiesto, pues esto solo sería necesario para caso del mes de Febrero.
O. Evitar el uso del operador relacional == de una misma variable contra 2 ó más tipos de datos enteros siempre que en su lugar se pueda utilizar la estructura de control “switch”
P. Programa en ejecución libre de desbordamiento de arreglos
Q. Todos los métodos tengan nombres claros y significativos de modo que sea fácil saber cuál es su función en el programa.
R. Ejercer la programación modular siempre que sea posible la reutilización de código
S. Evitar declarar atributos cuyos datos no formen parte de las propiedades que describan la clase en la que se declaren o bien no representen el estado del programa, y que por tanto sea posible una implementación utilizando variables locales y paso de parámetros.
T. Todas las clases (y sus atributos) cuenten con un nombre (identificador) claro y significativo a modo que a la lectura sea intuitivo saber para qué se utiliza.
U. Toda instrucción del programa en ejecución sea ejecutable sin necesidad de recompilar el programa
V. Cumplir con las convenciones Java (para clases, métodos, etc.).
W. Todo atributo siempre sea privado, a menos que exista una implementación que por eficiencia justifique un menor ocultamiento de información, pero que igualmente el diseño cumpla con la propiedad de encapsulamiento.
X. Si incluye comentarios, estos sean solo para Javadoc; en sus métodos, estos sean breves y referentes solo a los parámetros y resultados a obtener, y de las clases solo su propósito; los anteriores en términos de caja negra.
Y. El Código fuente sea libre de instrucciones para suprimir advertencias
Z. Evitar el uso del modificador static, a menos que sea indispensable el uso de atributos de clase y métodos de clase, justificando su uso con comentarios en código

Criterios de Evaluación

- Cumplir con las “Reglas de Operación y Evaluación del Taller de Sistemas Operativos Avanzados”.

- Fecha de asignación: 11 de Septiembre del 2013
- Fecha planeación de entrega: 18 de Septiembre del 2013
- Fecha límite de entrega (Sólo 55% del total): 09 de Octubre del 2013
- Observación: _Ninguna_____
- Calificación en base a cobertura de requerimientos y fecha de entrega en horas clase