Desarrollo de Aplicaciones Web /

Desarrollo de Aplicaciones Multiplataforma

Estructuras de datos avanzadas y control de excepciones

Programación I

Actividad

Estructuras de datos avanzadas y control de excepciones

**Objetivos**

* Escribir programas que manipulen información seleccionando y utilizando los tipos avanzados de datos facilitados por el lenguaje.
* Utilizar las clases básicas para almacenar y procesar información (listas y tablas de Hash).
* Implementar la gestión de excepciones en la utilización de clases facilitadas por el lenguaje.
* Implementar el lanzamiento de excepciones.

|  |
| --- |
| **¿Cómo lo hago?** |
| 1. Rellena los datos que se piden en la tabla “Antes de empezar”. 2. Haz uso de fuentes comunes como Arial, Calibri, Times New Roman etc. 3. Utiliza el color negro para desarrollar tus respuestas y usa otros colores para destacar contenidos o palabras que creas necesario resaltar. 4. Recuerda entregar la actividad en formato PDF a no ser que el profesor o profesora indique lo contrario. 5. Recuerda nombrar el archivo siguiendo estas indicaciones:  * Ciclo\_Módulo o crédito\_Tema\_ACT\_número actividad\_Nombre y apellido   + Ejemplo: AF\_M01\_T01\_ACT\_01\_Maria Garcia |

|  |  |
| --- | --- |
| **Antes de empezar…** | |
| Nombre |  |
| Apellidos |  |
| Módulo/Crédito |  |
| UF (solo ciclos LOE) |  |
| Título de la actividad |  |

***Se debe entregar un zip que contenga todos los archivos. java que has creado. Para poder aprobar un ejercicio, éste debe poder ejecutarse sin errores. Crea los archivos .java dentro de una carpeta de nombre actividad07***

Crea un programa para gestionar un listado de trabajadores y los integrantes de un grupo de trabajo.

1. Crea una clase **Persona** con los siguientes campos: nombre , edad y dni. Se debe crear el constructor y los métodos get y set necesarios.
2. Crea una excepción propia **EdadIncorrecta** que se lance con el mensaje “La edad no puede ser menor a 1 ni mayor a 110” en el método **setEdad** de Persona cuando se intente establecer una edad menor a 1 o mayor a 110.
3. Crea una excepción propia **NombreIncorrecto** que se lance con el mensaje “El nombre debe tener mínimo 3 letras" o bien "El nombre no puede tener dígitos" en el método **setNombre** de Persona cuando el nombre tenga menos de 3 letras o contenga algún número respectivamente.
4. Crea una excepción propia **DniIncorrecto** que se lance con el mensaje "El DNI ha de ser una string de 9 valores" en el método **setDni** de Persona cuando el dni no sea una string de longitud 9 (no hace falta hacer más comprobaciones para el DNI).
5. Crea una clase **GestionaPersonas** que al ejecutarse:
   1. Defina una **HashMap** de nombre “**trabajadores**” para almacenar un grupo de trabajadores según su DNI (String) y añada dos nuevas personas a “trabajadores” con valores establecidos por el programador.
   2. Defina una **ArrayList** de nombre “**grupoTrabajo**” para almacenar las personas que formarán parte del grupo de trabajo.
   3. Muestre el siguiente menú:

* 1-Crea un trabajador
* 2- Borra un trabajador
* 3- Añade un trabajador al grupo de trabajo
* 4- Borra un trabajador del grupo de trabajo.
* 0- Salir
  1. Después de ejecutar cada opción se debe mostrar el listado de personas dentro de “**trabajadores**” y dentro de “**grupoTrabajo**”.
  2. La opción “1-Crea un trabajador” debe de pedir los datos de un trabajador y añadirlo a “**trabajadores**” según su DNI. Si algún dato es incorrecto se debe capturar la excepción y volver a pedir el dato.
  3. La opción “2- Borra un trabajador” debe de pedir el DNI del trabajador a borrar y borrarlo de “**trabajadores**” si existe (no hace falta borrarlo también de “grupoTrabajo”).
  4. La opción “3- Añade un trabajador al grupo de trabajo” debe de pedir el DNI del trabajador a añadir en “grupoTrabajo”. Como máximo el ArrayList puede contener 2 personas, si se intenta añadir una tercera lanza una excepción propia del tipo **DemasiadoObjetos** y captúrala en el main.
  5. La opción “4- Borra un trabajador del grupo de trabajo.” debe pedir el índice dentro de la ArrayList y quitar ésa persona de “**grupoTrabajo”** pero no de “**trabajadores**”.Si el usuario indica un índice fuera del rango posible, lanza desde la función de borrar una persona, una excepción PosicionIncorrecta y captúrala en el main.

A continuación se muestra un ejemplo de ejecución del programa:





