Unidad 5

DIPLOMATURA EN PROGRAMACION JAVA

Ejercicios

Capítulo 9

Corrientes de Entrada y Salida

Capítulo 9

Ejercicio 1

En este ejercicio se creará un programa para leer texto de la corriente estándar de entrada y escribir el mismo en un archivo, anteponiendo a cada ingreso el número de línea que corresponde a cada entrada del archivo. El nombre del mismo se deberá especificar por línea de comando como argumento

1. Crear un espacio de trabajo en la carpeta Ejercicio 1 y dentro de él un nuevo proyecto Java llamado NumerarLineas.
2. Crear una clase que posea el inicio del programa llamada PruebaNumerarLineas en el paquete comandos que acepte sólo un argumento por línea de comando como nombre de archivo. El código deberá verificar la falta del argumento y pedir su ingreso en caso de ser necesario. Descartar cualquier argumento adicional ingresado.
3. Declarar una variable que almacena una referencia del tipo File que deberá ser inicializada con un objeto del mismo tipo cuyo constructor será invocado con el argumento ingresado por línea de comando
4. Dentro de un bloque try-catch, crear un objeto del tipo "lector" con "buffer" que "envuelva o decore" a la corriente System.in
5. También, en base a la referencia del tipo File, crear un objeto para escribir el archive en disco
6. Dentro de un ciclo, leer cada ingreso por teclado y escribirlo añadiendo el número de línea apropiado para cada ingreso
7. Asegurarse de cerrar todas las corrientes de E / S

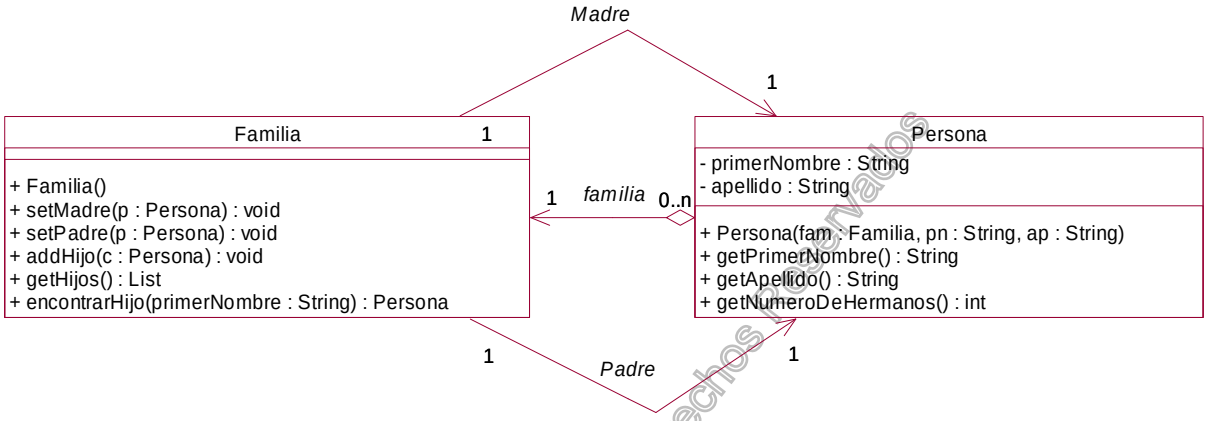
Nota 1: Para ingresar argumentos por línea de comandos en Eclipse, Seleccionar el comando Run Configurations del menú Run. En la ventana que se despliega, seleccionar "Java Application" y el botón a extrema izquierda de la parte superior que asemeja a una hoja con un signo +. Asegurarse que en la ventana a derecha aparece en la pestaña Main, en la caja de texto Project, "NumerarLineas" (sino seleccionarlo con el botón Browse). En la caja de texto Main class debe figurar "comandos.PruebaNumerarLineas". Seleccionar la pestaña Arguments. En el área de texto con título "Program arguments", ingresar el parámetro que se desea pasar al programa por línea de comandos. Posteriormente puede reutilizar la configuración creada para volver a ejecutar el programa.

Nota 2: Para ingresar los datos. Seleccionar el menú Window → Show view → Console en el Eclipse y escribir en la ventana que se abre. Para ver el archivo de salida, presionar F5 sobre el proyecto.

Ejercicio 2

Objetivo

En este ejercicio se van a leer y escribir objetos serializados a un archivo. Para eso deberá crear un espacio de trabajo en Capítulo 9 \Ejercicio 2 y un proyecto llamado Familias.



Escribir la clase Familia

La clase Familia se ha implementado. Lo que debe realizar es modificarla para que soporte serialización

En el archivo Familia.java se encuentran bloques de comentarios que empiezan y terminan con /*** ... ***/. Estos comentarios indican el lugar en el cual deberá proveer el código para el programa. Los archivos fuente para que construya el proyecto se encuentran en el directorio Capítulo 9\Ejercicio 2

1. Declarar que la clase Familia es serializable

Escribir la clase Persona

La clase Persona le es provista. Se debe modificar para que soporte serialización.

En el archivo Persona.java se encuentran bloques de comentarios que empiezan y terminan con /*** ... ***/. Estos comentarios indican el lugar en el cual deberá proveer el código para el programa

2. Declarar que la clase Persona es serializable

3. El aributo numeroDeHermanos se calcula en el procesamiento, de manera que no necesita ser serializado. Crear la declaración de numeroDeHermanos de manera que el mismo no se serialice.

4. El método getNumeroDeHermanos() retorna el número de hermanos para esa persona. De manera que el atributo se calcula cuando se procesa a la persona en particular. Este método deberá revisar si el atributo ya fue inicializado (que no sea cero) para luego realizar los cálculos. El cálculo es: restar uno al número total de hijos de esta familia. Asegurarse de imprimir en la consola el calculo que se esta por realizar y el resultado. Por último, retornar el valor calculado

Completar el programa EscribirFamilias
Este programa crea tres familias y la gente que le pertenece. Inicializa las relaciones entre los familiares y sus padres. Luego consulta el "número de hermanos" para Patricia Folino y Carlos Diaz para demostrar el cálculo en un atributo transient.

En el archivo EscribirFamilias.java se encuentran bloques de comentarios que empiezan y terminan con /*** ... ***/. Estos comentarios indican el lugar en el cual deberá proveer el código para el programa

5. Editar el archivo EscribirFamilias.java para incluir el código para declarar e inicializar la variable de la corriente de objetos. Nombrar al archivo que contiene a los objetos familias: familias.ser

6. Agregar el código para escribir las tres familias a la corriente de salida de objetos: No olvidarse de cerrar las corrientes de alto nivel.

7. Compilar y ejecutar el programa desde la clase EscribirFamilias. Se deberá ver la siguiente salida y se debe corroborar que exista el archivo familias.ser

Calculando el número de hermanos para Patricia Folino
Patricia Folino tiene 4 hermanos.
Calculando el número de hermanos para Carlos Diaz
Carlos Diaz tiene 2 hermanos.

Completar la clase LeerFamilias
Este programa asume que las tres familias han sido escritas al archive familias.ser. Este programa leerá las tres familias y recuperará el objeto Persona para Patricia Folino y Carlos Diaz. Por último, el programa consulta por el número de hermanos para cada uno de ellos para demostrar el uso de atributo transient numeroDeHermanos el cual no se guardó en disco y deberá recalcularse.

En el archivo LeerFamilias.java se encuentran bloques de comentarios que empiezan y terminan con /*** ... ***/. Estos comentarios indican el lugar en el cual deberá proveer el código para el programa

8. Editar el archivo LeerFamilias.java para incluir el código para declarar e inicializar la variable de la corriente de objetos.

9. Agregar el código para leer las tres familias a la corriente de entrada de objetos: No olvidarse de cerrar las corrientes de alto nivel.

10. Compilar y ejecutar el programa desde la clase LeerFamilias. Se deberá ver la siguiente salida

Calculando el número de hermanos para Patricia Folino
Patricia Folino tiene 4 hermanos.
Calculando el número de hermanos para Carlos Diaz
Carlos Diaz tiene 2 hermanos.

Ejercicio 3

En este ejercicio modificará el proyecto Banco para que el reporte generado cree un archivo en disco en lugar de una salida por pantalla para el reporte.

Modificar la clase ReporteCliente

1. Abrir el proyecto que se encuentra en el espacio de trabajo definido en el directorio en Capítulo 9\Ejercicio 3
2. Escribir un bloque try con manejo de recursos para que cree un objeto del tipo FileOutputStream y cree un archivo llamado "Reporte.txt". Finalizar la sentencia con ";"
3. Dentro del mismo paréntesis de manejo de recursos del bloque try generar un objeto del tipo PrintWriter que decore las salidas cuando se escriba en el archivo. Por ejemplo: PrintWriter salida = new PrintWriter(archivo);
4. Colocar todas las salidas que se tenían por pantalla mediante System.out.println dentro del bloque try que se acaba de construir
5. Reemplazar las salidas para que ahora escriban en el disco. Por ejemplo, si el decorador del tipo PrintWriter se llamó salida, como se mostró en el punto 3, remplazar System.out.println por salida.println
6. Verificar que no existan errores de compilación y ejecutar el programa
7. Refrescar el entorno de desarrollo para poder visualizar en el archivo creado
8. Verificar que la salida obtenida sea la misma que se obtenía por pantalla

Ejercicio 4 (desarrollarlo si el tiempo lo permite)

En este ejercicio se creará un programa para imprimir un listado para el directorio especificado por el último argumento de línea de comandos, o, en su defecto, ante la falta de argumento, del directorio actual. También se generará código para permitir como argumento un modificador llamado "–R" que indique si la lectura es recursiva y descendiente en el árbol de directorios.

1. Crear un espacio de trabajo en la carpeta Ejercicio 4 y dentro de él un nuevo proyecto Java llamado Directorios.

2. Crear una clase que posea el inicio del programa llamada ListadoDirectorios en el paquete utilidades, que reciba argumentos por línea de comando para especificar un directorio que se quiera listar. El programa deberá soportar hasta tres argumentos por línea de comandos. Si recibe más de tres argumentos deberá finalizar con el llamado al método `System.exit(0);` que finaliza el programa inmediatamente. Si no se recibe ningún argumento se deberá listar el directorio actual sin recursividad y sin modo "-verbose"
3. Si lo desea, puede crear una o más clases auxiliares para el manejo de las funciones de utilidad de directorio. En caso contrario, puede crear métodos estáticos en la clase ListadoDirectorios. Evalúe su solución y analice la calidad de código generado en base a las decisiones tomadas.
4. Todos los argumentos deberán ser opcionales para el programa, menos el camino que define el directorio. Si este no se encuentra como argumento pero si se pasa al programa un modificador, se tiene que terminar el mismo indicando que se coloque el camino adecuadamente.
5. El modificador "–R" indica si la lectura es recursiva y descendiente en el árbol de directorios para mostrar los subdirectorios.
6. El modificador "–verbose" indica al programa que muestre el tamaño del archivo y la fecha de la última modificación
7. Forma de uso:

java ListadoDirectorios [-R] [-verbose] [directorio]