Práctica: aplicación para gestionar un animalario

Actualización 15/03/2023: La práctica evaluará sobre 10 puntos sin la implementación de swing, aunque se valorará de manera positiva la entrega con swing. Esto solo aplica a la entrega I de la práctica, no a la entrega II.

Una empresa que se dedica a la cría de ratones transgénicos para su posterior venta a laboratorios necesita una aplicación de escritorio que les ayude a recoger información relativa a distintas poblaciones de ratones que cría en sus instalaciones. La aplicación debe permitir llevar un registro de los ratones y las características de los ratones que forman parte de una población. Las poblaciones serán finalmente vendidas como un todo a un laboratorio.

Para cada población de ratones, inicialmente el científico indicará: un nombre para la población, el nombre de la persona de la compañía responsable de dicha población y un número de días durante los cuales la población estará en las instalaciones de la empresa procreando. Ese número de días siempre será inferior a 270 días. Para cada ratón que forma parte de la población deberá poder indicarse un código de referencia, su fecha de nacimiento, su peso (en gramos, un número entero), su sexo (sólo podrá tomar valores "Macho" y "Hembra"), su temperatura corporal en grados centígrados (un número real), y un campo de texto libre que puede contener más de una línea de texto. Además, para cada ratón habrá que describir posibles mutaciones que afectan a sus cromosomas X e Y. El cromosoma X podrá ser normal, o contener una mutación que hace que el portador sea estéril (en el caso de los machos, si su cromosoma X tiene la mutación el macho será estéril; en el caso de las hembras ambos cromosomas X deberán contener la mutación para que sean estériles). El cromosoma Y podrá ser normal, o podrá contener una mutación que hace al ratón macho propenso a la poligamia (emparejarse con múltiples hembras). Todos los datos que se pidan al usuario deberán ser validados. Cada población de ratones se almacenará en un archivo independiente en el disco duro.

La aplicación deberá tener un menú principal con las siguientes opciones:

1. Abrir un archivo de texto plano con formato y extensión csv que contenga una población de ratones.
2. Crear una nueva población de ratones.
3. Añadir un nuevo ratón a una población ya existente.
4. Listar los códigos de referencia de todos los ratones de una población.
5. Eliminar un ratón de una población indicando su código de referencia.

6. Modificar los datos de un ratón, indicando previamente su código de referencia.
7. Ver información detallada de un ratón, habiendo especificado previamente su código de referencia.
8. Guardar (se supone que para usar esta opción previamente hemos abierto un archivo). En caso de no haber abierto un archivo se redirigirá directamente al usuario a la opción guardar como.
9. Guardar como (es decir, crear una copia diferente de la población de ratones).

Cuando el usuario seleccione la opción 7, deberá pedírsele al usuario el código de referencia del ratón del cual quiere ver los detalles, y a continuación deberá mostrarse toda la información detallada del ratón.

Consejos

Habla con el profesor para determinar cuál es la mejor forma para representar y almacenar los datos, y de estructurar la práctica. Organiza la práctica en al menos tres paquetes. Uno se encargará de la interfaz de usuario, otro se encargará de cargar y guardar datos en el disco duro, y el tercero se encargará de la lógica de negocio: añadir ratones, eliminar ratones, etc.… Utiliza funciones aisladas, atómicas y con una responsabilidad completa. Si tienes dudas sobre la utilidad de una función, consulta con el profesor. Visualice especialmente las clases donde se realiza el UML y la separación de paquetes.

Sobre la entrega y evaluación de la práctica

La práctica debe ser entregada en formato electrónico antes del 26 de marzo (inclusive). Para considerar que la práctica está entregada, el alumno deberá haber completado la entrega en formato electrónico a través del campus virtual. La práctica deberá contener así mismo un fichero ejecutable .jar que permita ejecutar el código fuente sin necesidad de compilar el código. Si no se encuentra el fichero .jar la práctica se calificará con un 0.

Por cada semana que se retrase la entrega de la práctica a partir de esta fecha, la nota final de la práctica se le descontará 1 punto (0.15 puntos por día aproximadamente). La práctica puede realizarse a través de una consola (en ese caso la nota máxima a la que podrá aspirar el alumno es un 8) o a través de una aplicación gráfica de escritorio empleando swing. El alumno tendrá que realizar una defensa de la práctica; si el alumno es incapaz de defender la práctica y explicar su funcionamiento, su nota en la práctica será un 0.

Ítems a entregar: Implementación íntegra de los requisitos establecidos en el enunciado
de la práctica junto a la memoria descriptiva del trabajo realizado. Se evaluará de forma más positiva la descripción de requisitos no implementados y conocidos descritos en la memoria que la falta de funcionalidad desconocida y no descrita.
Como resultado de esta entrega final se deberá enviar un fichero zip con la carpeta completa que contenga el proyecto software Netbeans correspondiente a la práctica (asegúrese de que incluye todo el código fuente, es la carpeta anterior al src, nbproject, etc..) y además un fichero ejecutable .jar que permita ejecutar la práctica sin tener que compilar el código fuente.

La entrega de la práctica forma parte de la evaluación de la práctica.

La memoria incluirá una portada con el nombre del alumno y se ajustará a la siguiente estructura:

- Análisis y descripción de la aplicación. Este análisis y descripción dará respuesta a las siguientes preguntas:
  - Cómo se han organizado y estructurado las clases y cuál es la responsabilidad de cada una.
  - Qué decisiones de diseño se han tomado.
  - Qué comprobaciones de integridad (y excepciones) se han implementado.
  - Qué técnicas de ordenación y búsqueda ha utilizado y por qué lo ha hecho.
  - Diagramas de clases UML.
- **Listado de fallos conocidos y funcionalidades definidas en el enunciado que no se han implementado** en el código entregado.
- Conclusiones (que incluirán, obligatoriamente, una valoración del tiempo dedicado a la práctica).

Además de la defensa, la práctica se evaluará en relación a:

- **Organización y estructura del código** (utilización de conceptos y patrones de programación, orientada a objetos: herencia, polimorfismo, encapsulación, reutilización, utilización correcta de estructuras de control, etc.).
- **Uso de Javadoc** para documentar las clases y métodos.
- Funcionamiento ajustado a los requisitos establecidos (incluyendo, además de chequeos de datos, integridad de la información, gestión de excepciones...).
- **Claridad del código y adecuación a las normas de estilo** (correcto nombrado de clases, métodos y variables, comentarios internos, indentación del código...).

Se advierte al alumno de que, como parte de la evaluación, se utilizará una herramienta antiplagio de código. En caso de copia total o parcial del código de la práctica, se aplicarán las medidas correspondientes en el código de conducta de la universidad CEU-San Pablo.

Checklist

Puedes utilizar este documento para organizar las tareas. *Las tareas en cursiva son opcionales y serán tratadas como opcionales.* **Las tareas en negrita con obligatorias y deben estar completadas para superar la práctica y el curso.**

No borres ninguna prueba realizada durante el proyecto. Se recomienda utilizar tests unitarios con Junit o, al menos, un main local en cada archivo.

Documentación

☐ **Todas las clases y las funciones con una visibilidad no privada tienen su javadoc correspondiente.**

☐ *Listado de fallos y funcionalidades no implementadas conocidas.*

☐ *Manual de aplicación.*

Diagramas

☐ **Realización de diagramas de clase UML.**

Implementación (se recomienda seguir el orden aquí descrito)

☐ **Aplicación con una única población sin ficheros funcionando (ver opciones de menú principal 2 a 7).**

☐ **Aplicación guarda datos de una población en un fichero con formato CSV.**

☐ **Aplicación carga datos de fichero CSV y permite luego interaccionar con la población cargada.**

☐ *Se han implementado test unitarios con Junit o en un main local.*

Interfaces

☐ **Creación de interfaz de consola de usuario.**

☐ *Creación de interfaz gráfica de usuario con Java Swing o JavaFX (no cubierto en la asignatura).*

Práctica: aplicación para gestionar un animalario II

La empresa de cría de ratones transgénicos se ha dado cuenta que el número de días durante los cuales la población estará en las instalaciones de la empresa procreando no siempre tiene que ser inferior a 270, por lo que debemos permitir valores más altos para este dato; hasta 630. El número de días siempre será un múltiplo de 45.

La empresa también desea buscar determinados ratones y actualmente no tiene forma de ordenarlos.
Se solicita que al seleccionar la opción 4, deberá preguntársele al usuario si desea ver los ratones ordenados alfabéticamente por referencia, cronológicamente por fecha o de mayor a menor por peso, y mostrarlos ordenados por el campo adecuado.

Nota: hasta aquí el 50% de la evaluación de la práctica.

Además, se ha dado cuenta que debido a las mutaciones inducidas en los ratones la evolución de la población no siempre es la esperada. Algunas poblaciones han estado procreando más o menos de lo esperado. Por ello nos han pedido que realicemos una simulación de Montecarlo en la aplicación para, dada una población inicial de ratones, estudiar cómo va a ir evolucionando a lo largo del tiempo.

La simulación partirá de una población inicial de ratones para la cual los científicos especificarán el número total de ratones en la población (NR), el porcentaje de machos ($0 < M < 100$) y el porcentaje de hembras (obviamente estos dos porcentajes deberán sumar 100%). Se indicará el porcentaje de machos estériles ($0 <= ME < 100$), y el porcentaje de machos proclives a la poligamia ($0 <= MP < 100$). Se indicará el porcentaje de cromosomas X estériles entre la población de hembras ($0 <= HE < 100$). Una hembra es estéril cuando ambos cromosomas X portan la mutación de la esterilidad.

A continuación, se creará una población "virtual" (es decir, simulada, no la introduce el usuario) de ratones para realizar la simulación. Para ello se repetirá NR veces el siguiente proceso:

1. Se genera un número aleatorio entre 0 y 99.
    a. Si el número aleatorio es < M, el ratón será un macho.
        i. Se genera un segundo número aleatorio entre 0 y 99.
            1. Si el número aleatorio es < ME el macho será estéril.
        ii. Se genera un tercer número aleatorio entre 0 y 99.
            1. Si el número aleatorio es < MP el macho tendrá el gen que lo hace propenso a la poligamia.
    b. Si el primer número aleatorio generado es >= M el ratón será una hembra

i. Se genera un segundo número aleatorio entre 0 y 99.
    1. Si el número aleatorio es < HE el primer cromosoma X de la hembra tendrá la mutación de la esterilidad.
ii. Se genera un tercer número aleatorio entre 0 y 99.
    1. Si el número aleatorio es < HE el segundo cromosoma X de la hembra tendrá la mutación de la esterilidad.

Se recomienda almacenar los ratones machos y los ratones hembra en dos listas independientes.
Una vez se ha terminado de generar la población de ratones, se crearán "familias" de ratones. Para ello se seguirá el proceso siguiente:

2. Se selecciona un ratón macho de la población.
    a. Si no es un ratón con el gen de poligamia, se selecciona un ratón hembra de la población y se crea una familia "Normal" de ratones. Ambos ratones son retirados de la población inicial.
    b. Si es un ratón con el gen de poligamia, se selecciona un ratón hembra de la población y se genera un número aleatorio entre 0 y 9.
        i. Si el número es < 5, terminamos la formación de la familia y creamos una familia "Poligámica" con el ratón macho y la ratona/las ratonas hembras.
        ii. Si el número es >= 5, seleccionamos un ratón hembra de la población y la añadimos a la actual familia, y volvemos al punto 2.b.

Nota: hasta aquí el 65% de la práctica.

El proceso de formación de familias termina cuando se agoten los ratones macho o los ratones hembra. Los ratones macho o hembra no emparejados no formarán parte de los ratones que procrean en esta camada. Se recomienda almacenar las familias de ratones creadas en una lista.

Una vez formadas las familias de ratones comenzará la simulación de la reproducción de estas familias. En la naturaleza, la fecundidad de los ratones depende de la disponibilidad de comida. En nuestro caso, los ratones siempre tendrán disponible toda la comida que quieran, y su fertilidad será elevada y constante a lo largo de todo el período de procreación. La reproducción se realiza del siguiente modo:

3. Para cada familia de ratones.
    a. Si la familia está compuesta por un único ratón macho y un único ratón hembra y ninguno es estéril.
        i. Se genera un numero aleatorio N entre 0 y 99; el número de crías de la camada será:

Número aleatorio	Número de crías
N <5	2
5 <= N <15	3
15 <= N <30	4

30 <= N <50	5
50 <= N <70	6
70 <= N <85	7
85 <= N <95	8
95 <= N <=99	9

b. Si la familia está compuesta por un ratón macho no estéril y 2 o más hembras, si bien la fertilidad de las hembras no se ve alterada, la fertilidad del macho relativa al número de hembras decrementa. Por ello, para cada una de las hembras.
    i. Se genera un numero aleatorio N entre 0 y 99; el número de crías de la camada de dicha hembra será:

Número aleatorio	Número de crías
N <10	2
10 <= N <25	3
25 <= N <45	4
45 <= N <60	5
60 <= N <75	6
75 <= N <95	7
95 <= N <=99	8

c. Si el macho de la familia es estéril, hay una cierta probabilidad de que la hembra quede embarazada por otro ratón macho y salga adelante la camada. Por ello, para simular la posible descendencia:
    i. Se genera un numero aleatorio N entre 0 y 99; el número de crías de la camada será:

Número aleatorio	Número de crías
N <15	2
15 <= N <35	3
35 <= N <70	4
70 <= N <90	5
90 <= N <=99	6

d. Si la hembra de la familia es estéril, la familia producirá 0 crías.

Cada una de las crías tiene un 50% de probabilidad de heredar el cromosoma X del padre, un 50% de heredar el cromosoma Y del padre, y un 50% de probabilidad de heredar cada uno de los cromosomas X de la madre. Cuando heredan los cromosomas del progenitor correspondiente, las mutaciones que pudiese tener presentes ese cromosoma estarán presentes en el hijo.

Para determinar la composición genética de un ratón hijo, para cada uno de los ratones hijos,

a. Generamos un número aleatorio P entre 0 y 99. Si el número aleatorio P < 50, el ratón hereda el gen X del padre. En caso contrario, hereda el gen Y.

b. Generamos un número aleatorio M entre 0 y 99. Si el número aleatorio M <50, el ratón hereda el "primer" gen X de la madre. En caso contrario, hereda el "segundo" gen X de la madre.

Cada ciclo de reproducción de los ratones dura 45 días (unos 20 días de gestación, y unos 25 días para que las crías se independicen). Habrá que simular los múltiples ciclos de reproducción hasta llegar al número de días que la población de ratones permanece en las instalaciones de la empresa. Por ello, una vez termine el primer ciclo de reproducción volvemos al punto 2, se vuelven a formar familias de ratones, y se vuelven a reproducir. Las crías del ciclo inmediatamente anterior en el ciclo siguiente todavía no son maduras sexualmente, por lo que no forman parte del proceso de reproducción. Las crías serán sexualmente maduras en el segundo ciclo de reproducción después de que hayan nacido.

Dado que la longevidad de los ratones es aproximadamente 2.5 años, y dado que nunca se crían las poblaciones durante más de dos años en la empresa, no es necesario simular la muerte de algunos de los ratones.

La aplicación deberá tener un menú principal con las siguientes opciones:

1. Abrir un archivo de texto plano con formato y extensión csv que contenga una población de ratones.
2. Crear una nueva población de ratones a partir de datos introducidos por el usuario.
3. Crear una población virtual de ratones a partir de los porcentajes de machos, hembras y de mutaciones.
4. Añadir un nuevo ratón a una población ya existente.
5. Listar los códigos de referencia de todos los ratones de una población.
6. Eliminar un ratón de una población indicando su código de referencia.
7. Modificar los datos de un ratón, indicando previamente su código de referencia.
8. Ver información detallada de un ratón, habiendo especificado previamente su código de referencia.
9. Simular la evolución a lo largo del tiempo de la población; dicha población podría haber sido introducida manualmente por el usuario o creada virtualmente.
10. Guardar (se supone que para usar esta opción previamente hemos abierto un archivo).
11. Guardar como (es decir, crear una copia diferente de la población de ratones).

Los resultados de la simulación de la evolución de la población no es necesario guardarlos en el fichero o en la memoria una vez mostrados, ya que se podrán generar de un modo sencillo desde el programa cuando se quiera.

Al seleccionar la opción 4, deberá preguntársele al usuario si desea ver los ratones ordenados alfabéticamente por referencia, cronológicamente por fecha o de mayor a menor por peso, y mostrarlos ordenados por el campo adecuado.

Al seleccionar la opción 9, deberá mostrarse el número de ratones total de la población, el porcentaje de machos y de hembras, y el porcentaje de machos y hembras con mutación: machos normales, machos polígamos y machos estériles; y hembras normales, hembras que tienen un gen de esterilidad, pero son fértiles, y hembras

estériles. Para calcular los porcentajes, un macho estéril es considerado estéril independientemente de si tiene el gen de poligamia o no. Debemos indicar el valor de estos porcentajes desde la población inicial, hasta la población final pasando por cada uno de los ciclos de procreación de la población. Idealmente, esto se hará mediante un gráfico de barras empleando códigos de colores para los distintos porcentajes. Opcionalmente, se puede hacer mostrando estos porcentajes en la consola o un campo de texto en una aplicación gráfica.

Consejos

Habla con el profesor para determinar cuál es la mejor forma para representar y almacenar los datos, y de estructurar la práctica. Organiza la práctica en al menos tres paquetes. Uno se encargará de la interfaz de usuario, otro se encargará de cargar y guardar datos en el disco duro, y el tercero de la interfaz de usuario. Podría usarse un cuarto paquete para las simulaciones. Utiliza abundantes métodos de pequeño tamaño.

Sobre la entrega y evaluación de la práctica

La práctica debe ser entregada en formato electrónico e impresa en papel antes del 12 de mayo (inclusive). **Todos los estudiantes que quieran entregar la práctica antes, pueden hacerlo y así podrán realizar la defensa con antelación para saber si deben acudir al examen de la convocatoria ordinaria.**
Para considerar que la práctica está entregada, el alumno deberá haber completado la entrega en formato electrónico (a través del campus virtual). La práctica deberá contener así mismo un fichero ejecutable .jar que permita ejecutar el código fuente sin necesidad de compilar el código. Si no se encuentra el fichero .jar la práctica se calificará con un 0.

La práctica puede realizarse a través de una consola (en ese caso la nota máxima a la que podrá aspirar el alumno es un 8) o a través de una aplicación gráfica de escritorio empleando swing. El alumno tendrá que realizar una defensa de la práctica; si el alumno es no supera la defensa de la práctica y explica su funcionamiento, su nota en la práctica será un 0.

Ítems a entregar: Implementación íntegra de los requisitos establecidos en el enunciado de la práctica junto a la memoria descriptiva del trabajo realizado. Se evaluará de forma más positiva la descripción de requisitos no implementados y conocidos descritos en la memoria que la falta de funcionalidad desconocida y no descrita.
Como resultado de esta entrega final se deberá enviar un fichero zip con la carpeta completa que contenga el proyecto software Netbeans correspondiente a la práctica (asegúrese de que incluye todo el código fuente, es la carpeta anterior al src, nbproject, etc..) y además un fichero ejecutable .jar que permita ejecutar la práctica sin tener que compilar el código fuente.

La entrega de la práctica forma parte de la evaluación de la práctica.

La memoria incluirá una portada con el nombre del alumno y se ajustará a la siguiente estructura:

- Análisis y descripción de la aplicación. Este análisis y descripción dará respuesta a las siguientes preguntas:
    - Cómo se han organizado y estructurado las clases y cuál es la responsabilidad de cada una.
    - Qué decisiones de diseño se han tomado.
    - Qué comprobaciones de integridad (y excepciones) se han implementado.
    - Qué estructuras de datos ha utilizado y por qué lo ha hecho.
    - Qué técnicas de ordenación y búsqueda ha utilizado y por qué lo ha hecho.

    - Diagramas de clases UML.
- **Listado de fallos conocidos y funcionalidades definidas en el enunciado que no se han implementado** en el código entregado.
- Conclusiones (que incluirán, obligatoriamente, una valoración del tiempo dedicado a la práctica).

Además de la defensa, la práctica se evaluará en relación a:

- **Organización y estructura del código** (utilización de conceptos y patrones de programación, orientada a objetos: herencia, polimorfismo, encapsulación, reutilización, utilización correcta de estructuras de control, etc.).
- Utilización de **estructuras de datos adecuadas** y utilización de **técnicas de ordenación y búsqueda**. El uso de todas ellas deberá ser analizado y justificado en la memoria descriptiva.
- **Uso de Javadoc** para documentar las clases y métodos.
- Funcionamiento ajustado a los requisitos establecidos (incluyendo, además de chequeos de datos, integridad de la información, gestión de excepciones...).
- **Claridad del código y adecuación a las normas de estilo** (correcto nombrado de clases, métodos y variables, comentarios internos, indentación del código...).
- Se advierte al alumno de que, como parte de la evaluación, se utilizará una herramienta antiplagio de código. En caso de copia total o parcial del código de la práctica, se aplicarán las medidas correspondientes en el código de conducta de la universidad CEU-San Pablo.

Checklist

Puedes utilizar este documento para organizar las tareas. *Las tareas en cursiva son opcionales y serán tratadas como opcionales.* **Las tareas en negrita con obligatorias y deben estar completadas para superar la práctica y el curso.**

No borres ninguna prueba realizada durante el proyecto. Se recomienda utilizar tests unitarios con Junit o, al menos, un main local en cada archivo.

Documentación

☐ **Todas las clases y las funciones con una visibilidad no privada tienen su javadoc correspondiente.**

☐ *Listado de fallos y funcionalidades no implementadas conocidas.*

☐ *Manual de aplicación.*

Diagramas

☐ **Realización de diagramas de clase UML actualizados respecto a la entrega I.**

Implementación (se recomienda seguir el orden aquí descrito)

☐ **Aplicación con una única población sin ficheros funcionando (ver opciones de menú principal 2, y 4 a 8).**

☐ **Aplicación guarda datos de una población en un fichero con formato CSV.**

☐ **Aplicación carga datos de fichero CSV y permite luego interaccionar con la población cargada.**

☐ **Aplicación ordena los ratones al listarlos según la opción escogida por el usuario correctamente.**

☐ **Aplicación permite crear poblaciones simuladas a partir de porcentajes (opción 3).**

☐ Aplicación genera familias de acuerdo a la simulación de Montecarlo.

☐ *Se han implementado test unitarios con Junit o en un main local.*

Interfaces

☐ **Creación de interfaz de consola de usuario.**

☐ *Creación de interfaz gráfica de usuario con Java Swing o JavaFX (no cubierto en la asignatura).*