
Primer proyecto: Técnicas algorítmicas
v. 1.0

"An algorithm must be seen to be believed."
-- D. E. Knuth

Descripción general

Con el fin de abarcar las diversas técnicas algorítmicas estudiadas hasta ahora, el presente proyecto está dividido en 2 partes: una sobre algoritmos de ordenamiento y otra sobre una modificación al problema del "recorrido del caballo" en un tablero de ajedrez.

Parte 1

El ordenamiento de estructuras de datos lineales ha sido un problema que se ha resuelto utilizando diferentes técnicas algorítmicas, algunas de las cuales han sido, desde luego, mejores que otras.
En este proyecto se van a implementar una serie de algoritmos de ordenamiento con la particularidad de que se asignarán frecuencias de sonido para cada uno de los valores por ordenar.
De este modo, cada vez que un elemento cambie su ubicación en la estructura de datos, se emitirá su correspondiente sonido.

Descripción específica

Algoritmos

Los algoritmos de ordenamiento por implementar son:

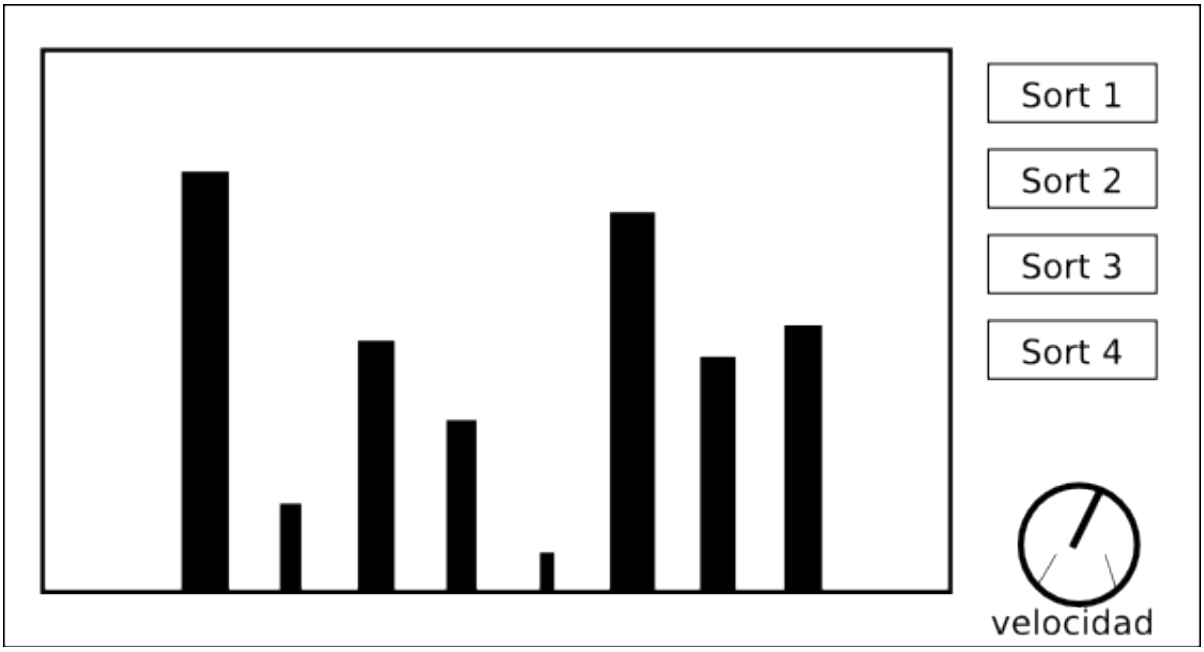
- Bubble sort
- Insertion sort
- Merge sort
- Quick sort

Estructura de datos

La estructura de datos que almacenará los elementos a ordenar, será una **lista doblemente enlazada**.

GUI

Se visualizará en todo momento, una representación del arreglo a modo de **gráfico de barras**, de forma que el elemento más pequeño del arreglo se representará con la barra más pequeña y el elemento más grande del arreglo con la barra más grande.
Una vez que el arreglo sea llenado con número aleatorios, se reflejarán estos números con la representación gráfica correspondiente.
Junto a la representación gráfica del arreglo, se deberá mostrar un botón por cada algoritmo de ordenamiento implementado, de forma que al darle click, se ejecutará ése ordenamiento sobre el arreglo.
Se debe visualizar la animación de las barras cambiando de posición de acuerdo al algoritmo utilizado.
Adicionalmente, se debe poder aumentar o disminuir la velocidad de la animación.



Input

La entrada de la aplicación será la cantidad de números aleatorios que se almacenarán en la estructura de datos.

Output

La salida será:

1. La animación de las barras posicionándose en el lugar apropiado de acuerdo a cada algoritmo de ordenamiento.
2. El sonido de cada una de las barras al moverse dentro del arreglo. Cada barra tendrá asociada una frecuencia de sonido que se escuchará al moverse la barra.

Ver referencia [0].

Se debe realizar la ejecución de cada uno de los algoritmos con 1.000, 5.000 y 10.000 elementos en 3 computadoras distintas, llevando registro del hardware de cada máquina (memoria y CPU) y los resultados (tiempo de ejecución) obtenidos en cada una de ellas. En todos los casos, las ejecuciones se deben hacer con la misma velocidad en la aplicación.

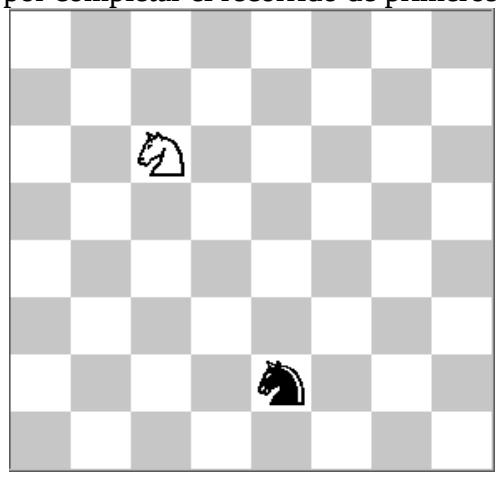
Parte 2

Se realizará una versión modificada del problema del “recorrido del caballo” en un tablero de ajedrez. Dicho problema consiste en encontrar una secuencia de movidas del caballo, de forma que pueda recorrer todos los cuadros del tablero, sin repetir ninguna de las casillas que ya ha sido visitada.

Descripción específica

Este proyecto se basa en el problema del “recorrido del caballo”.

En esta versión modificada del problema, se tendrán 2 caballos en el mismo tablero que competirán por completar el recorrido de primeros.



Siempre deberán estar visibles ambos caballos y podrán hacer sólo un movimiento por segundo siguiendo una **estrategia de backtracking**.

La colocación inicial de cada caballo será aleatoria y diferente (i.e. no pueden haber 2 caballos en una misma casilla en ningún momento del juego).

Una vez iniciado el juego, cada casilla que visita un caballo será “asegurada” con una contraseña (generada aleatoriamente), de modo que cuando el caballo rival necesite pasar por esa casilla, tiene primero que romper por **fuerza bruta** la contraseña que su rival colocó.

Las contraseñas serán de longitud configurable utilizando solamente las **letras** del alfabeto inglés (no tildes, ni eñes, etc).

La contraseña sólo será conocida por el caballo “dueño” de la casilla y el caballo que trata de adivinar la contraseña deberá “consultar” a su rival cada uno de los intentos. El caballo “dueño” de la casilla le contestará de forma negativa cada vez que reciba una contraseña incorrecta y una respuesta positiva cuando finalmente el caballo “invasor” logre adivinar la contraseña.

Se sugiere la utilización de hilos para implementar este mecanismo de “preguntas” y “respuestas”.

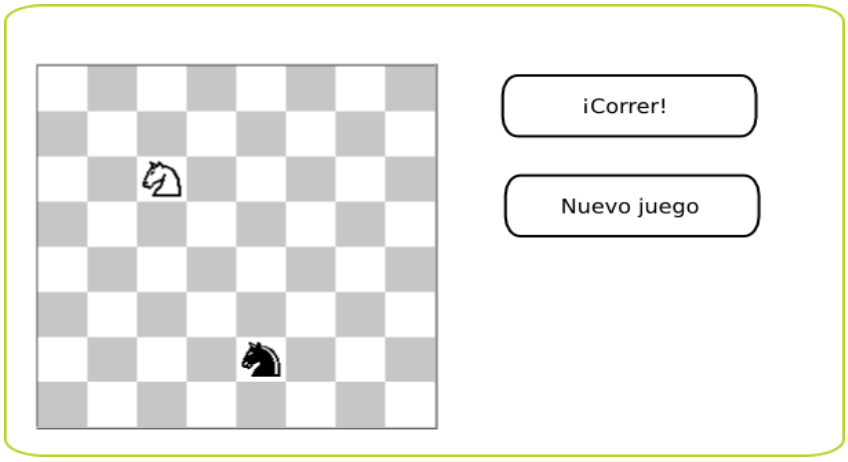
Se debe respetar los siguientes lineamientos:

- Debe haber “fair-play” (sin colocar delays antes de responder al rival, aceptar de inmediato cuando fue rota una contraseña, etc).

Input

- Longitud de las contraseñas
- Dimensiones del tablero: 8x8, 12x12, 16x16

GUI



Para efectos de concordancia y uniformidad entre proyectos la primer columna de la izquierda siempre se denotará con la letra “a” y la primera fila (abajo) del tablero se denotará con el número “1”. Por ejemplo en un tablero de 8x8 las coordenadas serían:

8	a8	b8	c8	d8	e8	f8	g8	h8
7	a7	b7	c7	d7	e7	f7	g7	h7
6	a6	b6	c6	d6	e6	f6	g6	h6
5	a5	b5	c5	d5	e5	f5	g5	h5
4	a4	b4	c4	d4	e4	f4	g4	h4
3	a3	b3	c3	d3	e3	f3	g3	h3
2	a2	b2	c2	d2	e2	f2	g2	h2
1	a1	b1	c1	d1	e1	f1	g1	h1
	a	b	c	d	e	f	g	h

Las coordenadas con este formato servirán para establecer tanto la posición inicial de los caballos como los movimientos posteriores que cada caballo realizará.

Una vez definidos los detalles de inicio, se dará click sobre el botón de “¡correr!” y comenzará el juego.

Aspectos administrativos

Puntos extra

Antes de la entrega del proyecto el profesor podrá solicitar por medio del foro, funcionalidad adicional con algún valor sobre la nota base.
Sólo podrán optar por los puntos extra aquellos proyectos que hayan completado la totalidad de la funcionalidad descrita en este documento.

Entregables

- Documentación impresa con:
  - Diagrama de clases en UML
  - Explicación de las decisiones de diseño
  - Una tabla con las diferentes subtareas de este proyecto, que se deberá llenar con el tiempo estimado para cada subtarea y tiempo real que tomó realizar dicha tarea.
  - Conclusiones
  - Recomendaciones
- Un archivo comprimido con el código fuente completo al correo hesquivel@ic-itcr.ac.cr El título del correo debe ser **I-2014-IC-3002-P1-[nombres]**

Fecha de entrega

- Martes 29 de abril, 2014.

Aspectos varios

- El proyecto se desarrollará en parejas a lo largo de todo el proyecto o de manera individual.
  - Si el/la estudiante decide hacerlo individual, deberá terminarlo de la misma forma.
  - No se permite el cambio de parejas.
  - Si la pareja no continúa trabajando junta, ambos miembros deben continuar de manera individual el proyecto hasta el final.
  - A más tardar el domingo 13 de abril me deberán enviar un correo con el título **I-2014-IC-3002-P1-[nombres]** indicando los nombres de los equipos de trabajo. Si no se recibe correo, se asumen que trabajarán de forma individual.

- Cualquier intento de fraude se calificará con nota de 0, además de la aplicación del reglamento vigente.
- **La tarea debe ejecutarse correctamente en GNU/Linux.**
- Las revisiones serán en los laboratorios a una hora acordada mediante citas.
- La calificación será individual.
- Durante la defensa de proyecto se le solicitará a los/las estudiantes, realizar modificaciones al código fuente.

Referencias

[0] https://www.youtube.com/watch?v=t8g-iYGHpEA