**Departamento de TIC**

**Algoritmos y Estructuras de Datos**

**Laboratorio Unidad 6**

**Concurrencia y Dibujo Básico en 2D**

**Unidad 6: Concurrencia y Dibujo Básico en 2D**

Al finalizar esta unidad, el estudiante estará en capacidad de:

OE6.1 Desarrollar un programa que maneje concurrencia, de manera que sea posible que ejecute más de una parte del programa de manera simultánea, utilizando hilos de ejecución (threads).

OE6.2 Construir interfaces de usuario que incluyan gráficas en 2 dimensiones como una alternativa en la presentación de información al usuario.

**Preparación**

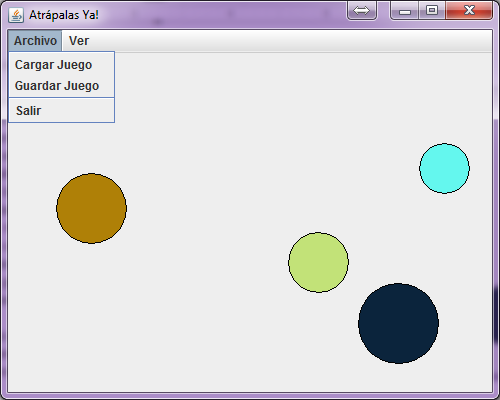
* Lea cuidadosamente el enunciado, la documentación suministrada y cada uno de los puntos que debe desarrollar antes de empezar su desarrollo. Pregunte a su profesor cualquier duda respecto al enunciado o a los requerimientos funcionales que debe desarrollar.
* Lea cuidadosamente la rúbrica del laboratorio de la unidad 3 ([ver rúbrica](https://docs.google.com/spreadsheets/d/1W9Ne3AZXnRRuVPlJ0fgZnPwx0q9N3PQRtaXP4GH83Wg/edit?usp=sharing)).
* El trabajo debe ser realizado **individualmente**.
* El trabajo será entregado en la fecha y hora establecida en Moodle.

**Enunciado**

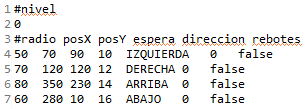
**JUEGO ATRÁPALAS YA!**

Con el auge de las aplicaciones para móviles y sobre todo los juegos para estos dispositivos, usted ha notado que para que un juego tenga éxito no tiene que ser necesariamente muy elaborado. Usted ha visto juegos bastante sencillos que han tenido gran acogida precisamente gracias a su sencillez.

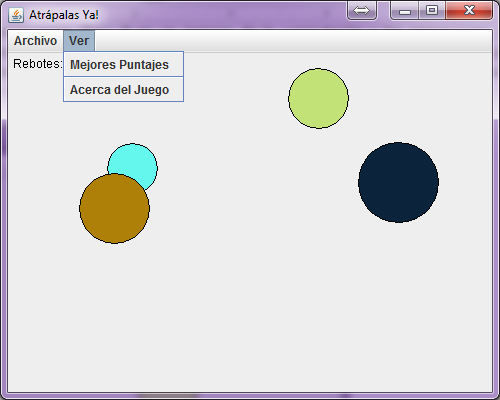
Usted se está aprendiendo cómo desarrollar aplicaciones móviles pero aún no tiene tanta destreza como en Java. Tiene una idea y quiere desarrollar rápidamente un prototipo para hacer pruebas sobre el juego para ver si es prometedor, por lo tanto ha decidido desarrollar el prototipo utilizando Java y las herramientas conceptuales y técnicas adquiridas en su actual curso de algoritmos.

La idea, como se indicó antes, es bastante sencilla y se explica a continuación: aparecen unas esferas en la pantalla moviéndose, algunas horizontal y otras verticalmente. Durante su movimiento, si la esfera alcanza un extremo de la ventana de juego, ésta rebotará y se moverá ahora en sentido contrario. El jugador debe detenerlas haciendo clic sobre cada una de las esferas que aparecen en la pantalla, lo más rápido posible y antes de que reboten. Por cada rebote, el contador de rebotes aumentará. El mejor jugador es aquel que detenga todas las esferas con la menor cantidad de rebotes.

La configuración de un nuevo juego se puede (debe) cargar desde un archivo de texto a través de la opción del menú que se muestra en la imagen de la izquierda. El archivo de la configuración de un (1) posible juego se muestra a continuación:



Las líneas en blanco deben ser ignoradas. Las que inicien con el símbolo # son líneas de comentarios, por tanto también deben ser ignoradas. La primera línea válida es un número entero que indica el nivel de dificultad del juego que tiene ese archivo. Son tres niveles posibles 0, 1 y 2. En el resto del archivo, cada línea válida representa una esfera y tiene los valores de las características de la esfera en el siguiente orden y separados por tabulador (“\t”): diámetro de la esfera, posición en X, posición en Y, tiempo de espera (para su movimiento), dirección inicial del movimiento de la esfera, cantidad de rebotes y si está detenida. Su programa debe tener la capacidad de leer un archivo como el descrito, el cual debe ser leído y cargado en los objetos del modelo.

Una vez el jugador ha detenido todas las esferas, si su puntaje (la cantidad de rebotes) se encuentra dentro de las mejores 10 del nivel de dificultad, entonces se le pide su nombre y éste junto con el puntaje queda guardado en el Hall de la Fama. Pero para que esto ocurra de forma permanente se requiere que los datos sean persistentes. Sin embargo, un archivo de texto sería demasiado fácil de modificar fuera del programa, por lo que un usuario podría agregarse fácilmente a través de un editor de texto y ponerse un excelente (o imposible) puntaje, que realmente no es cierto. Por lo anterior, tanto la escritura y la lectura de los puntajes se llevará a cabo a través de serialización. Cuando el programa se cierra debe guardarse la información el puntaje y al abrirse de nuevo, si se consulta esta información a través de la opción del menú debe visualizarse pues el objeto ha sido deserializado con la información que tenía almacenada.

**Requerimientos funcionales**

|  |  |
| --- | --- |
| **Nombre:** | R. #1 Si la esfera alcanza el borde, hacer que rebote |
| **Resumen:** | El programa debe hacer que las esferas al llegar al borde de la pantalla rebote |
| Entradas: |  |
|  |  |
| Resultados: | La esfera rebota con éxito |
|  |  |

|  |  |
| --- | --- |
| **Nombre:** | R. #2 Las esferas deben detenerse con un click |
| **Resumen:** | Cuando el usuario de click encima de la esfera, esta deberá detenerse |
| Entradas: |  |
|  |  |
| Resultados: | La esfera se detiene con éxito |
|  |  |

|  |  |
| --- | --- |
| **Nombre:** | R. #3. Cargar desde un archivo plano la configuración del juego |
| **Resumen:** | El programa debe cargar la configuración del juego de un archivo plano que contiene información como el nivel y las características de las determinadas esferas |
| Entradas: |  |
|  |  |
| Resultados: | El programa carga con éxito la configuración del juego desde un archivo plano |
|  |  |

|  |  |
| --- | --- |
| **Nombre:** | R. #4. Registrar los 10 mejores puntajes |
| **Resumen:** | El programa debe guardar el puntaje del usuario entre los mejores 10, si este puntaje se encuentra entre los mejores |
| Entradas: |  |
|  |  |
| Resultados: | El puntaje se guarda con exito |
|  |  |

|  |  |
| --- | --- |
| **Nombre:** | R. #5. Serializar los puntajes |
| **Resumen:** | La escritura de los puntajes se hará por medio de serialización |
| Entradas: |  |
|  |  |
| Resultados: | Se serializan correctamente los puntajes |
|  |  |

|  |  |
| --- | --- |
| **Nombre:** | R. #6. Deserialización puntajes |
| **Resumen:** | El programa lee la serialización de los puntajes |
| Entradas: |  |
|  |  |
| Resultados: | Deserialización realizada con éxito |
|  |  |

|  |  |
| --- | --- |
| **Nombre:** | R. #7. El jugador debe ganar cuando haya detenido todas las esferas |
| **Resumen:** | El programa debe saber cuando el jugador ha detenido todas las esferas para así declarar ganador al jugador |
| Entradas: |  |
|  |  |
| Resultados: | Se declara ganador al jugador exitosamente |
|  |  |

|  |  |
| --- | --- |
| **Nombre:** | R. #8. Pedir el nombre del jugador para registrar el puntaje |
| **Resumen:** | El programa debe registrar el puntaje de un jugador si este se encuentra entre los mejores, para esto se requiere el nombre del jugador |
| Entradas: |  |
|  |  |
| Resultados: | Nombre pedido con éxito |
|  |  |

|  |  |
| --- | --- |
| **Nombre:** | R. #8. Llevar la cuenta del puntaje |
| **Resumen:** | El programa debe por cada esfera que rebote aumentar el puntaje del jugador |
| Entradas: |  |
|  |  |
| Resultados: | Cuenta del puntaje llevada con éxito |
|  |  |