Instituto Tecnológico de Costa Rica
Escuela Ingeniería en Computadores
Paradigmas de Programación (CE1106)
Segundo Semestre 2024
Tarea# 4

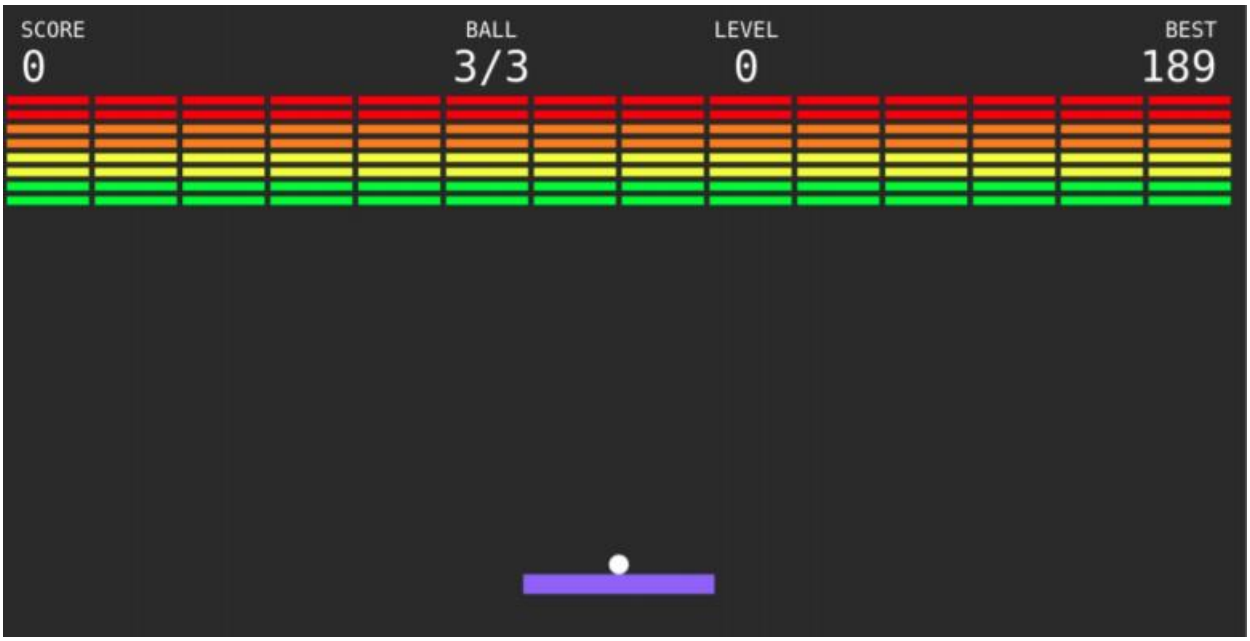
1. **Objetivo General**
   - ➔ Desarrollar una aplicación que permita reafirmar el conocimiento de los **paradigmas de programación imperativo y orientado a objetos**.

2. **Objetivos Específicos**
   - ➔ Desarrollar una aplicación en el lenguaje de programación C y java.
   - ➔ Aplicar los conceptos de programación imperativa y orientada a objetos.
   - ➔ Crear y manipular listas como estructuras de datos.

3. **Datos Generales**
   - ➔ El valor del proyecto: 20% (7.5% C, 7.5% Java, 5 Anexo)
   - ➔ Nombre código: breakOutTec.
   - ➔ La tarea debe ser implementada en grupos de no más de 4 personas.
   - ➔ La **fecha de entrega** es: 5 de Noviembre 2024.
   - ➔ Cualquier indicio de copia será calificado con una nota de 0 y será procesado de acuerdo al reglamento.

4. **Descripción del juego.**

"Breakout es un videojuego arcade desarrollado por Atari, Inc. y lanzado al mercado el 13 de mayo de 1976. El juego consiste que en la parte inferior de la pantalla una rayita simulaba una raqueta de front-tenis que el jugador podía desplazar de izquierda a derecha. En la parte superior se situaba una banda conformada por rectángulos que simulaban ser ladrillos. Una pelotita descendía de la nada y el jugador debía golpearla con la raqueta, entonces la pelota ascendía hasta pegar en el muro, los ladrillos tocados por la pelota desaparecían. La pelota volvía a descender y así sucesivamente. El objetivo del juego era terminar con la pared de ladrillos" (Wikipedia, 2024).



Reglas del juego:

El jugador contará con tendrá tres vidas (oportunidades).
Se tendrán 4 niveles de ladrillos 2 filas verdes, 2 filas amarillas, 2 filas naranjas y 2 filas rojas.
Por cada uno de los ladrillos que se destruya el computador asignará el valor que el **administrador** del juego desde el servidor asigno a esa fila de ladrillos.

Si la bola destruye un ladrillo que tiene una vida la cantidad de oportunidades aumentará en uno.
Si la bola destruye un ladrillo que tiene una bola un nuevo balón se pondrá en juego (si tenía una ahora tendrá 2, si tenía 2 ahora tendrá 3 así sucesivamente).
Si la bola destruye un ladrillo raqueta doble, el tamaño de la raqueta se duplicará.
Si la bola destruye un ladrillo raqueta a la mitad, el tamaño de la raqueta se disminuye en un 50% de su tamaño.
Si la bola destruye un ladrillo Velocidad Mas la velocidad de la bola aumenta.
Si la bola destruye un ladrillo Velocidad Menos la velocidad de la bola disminuye.
Si un balón no es impactado por la raqueta se pierde una vida.
Si todos los ladrillos se destruyen se avanza un nivel, donde la velocidad de la(s) bola(s) aumenta.
Si la cantidad de vidas/oportunidades llega a cero el juego termina.

4.1. **El servidor:** deberá estar programado en Java y mantendrá la estructura del juego. Será el encargado de crear y mantener la estructura del juego.
   4.1.1. Será el encargado de:
      4.1.1.1. Asignar/cambiar puntuación a cada nivel de ladrillos.
      4.1.1.2. Asignar vidas a un ladrillo específico.
      4.1.1.3. Asignar una bola a un ladrillo.

4.1.1.4. Asignar Raqueta Doble a un ladrillo.
4.1.1.5. Asignar Raqueta Mitad a un ladrillo.
4.1.1.6. Asignar Velocidad Mas a un ladrillo.
4.1.1.7. Asignar Velocidad Menos a un ladrillo.

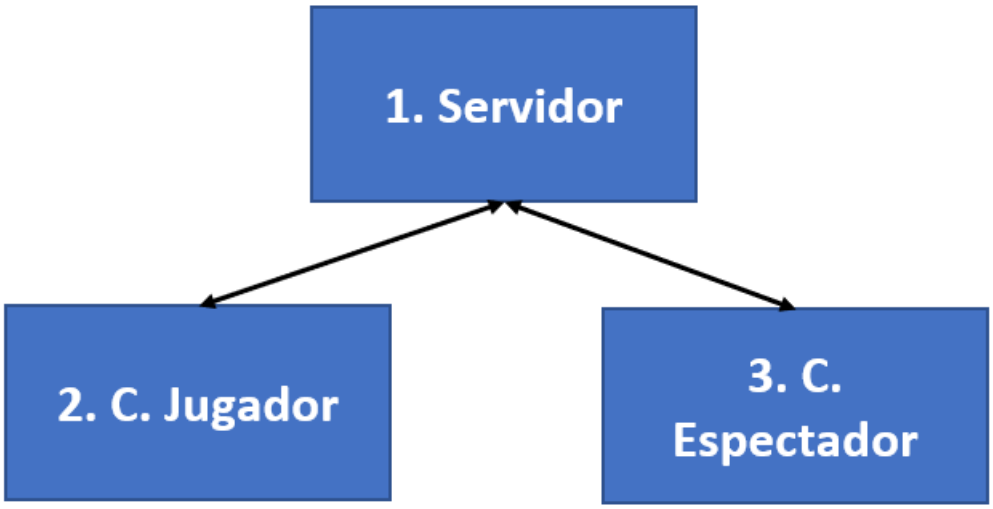
Nota: *Estas asignaciones y cambios se realizan en tiempo de ejecución del juego.*

4.2. **Cliente Jugador:**

4.2.1. Deberá estar programado en C, se refiere a la interfaz gráfica del juego controla la raqueta con las direccionales: izquierda, derecha.

4.2.2. Es el encargado de mostrar la cantidad de vidas disponibles, la puntuación del juego y el nivel en el que se encuentra.

4.3. **Cliente espectador:** Es aquel cliente que puede observar una partida en desarrollo. Debe leer la estructura enviada por el servidor, pero no podrá mover la raqueta.



4.4. **Conexión cliente servidor:** La conexión entre C y Java debe realizarse utilizando Sockets (chuidiang.org, 2024)

4.5. **Aspectos de implementación en C:** Recuerden se evaluará que todas las constantes estén en un archivo aparte. Se evaluará el uso de struts.

4.6. **Aspectos de implementación en Java:** Recuerden se evaluará que todo esté programado según los conceptos de OO (Clases, paquetes, patrones de diseño -incluir al menos 3 excluyendo el singleton-).

5. **Entregables**

5.1. Código fuente comentado.

5.2. Manual de usuario.

5.3. Anexo de Atributos.

6. **Documentación**
    1. Se deberá entregar un documento que contenga:
        1.1. Manual de usuario: cómo ejecutar el programa.
        1.2. Descripción de la utilización de las estructuras de datos desarrolladas (por ejemplo descripción del nodo de una lista).
        1.3. Descripción detallada de los algoritmos desarrollados.
        1.4. Problemas sin solución: En esta sección se detalla cualquier problema que no se ha podido solucionar en el trabajo.
        1.5. Plan de Actividades realizadas por estudiante: Este es un planeamiento de las actividades que se realizaran para completar la tarea, este debe incluir descripción de la tarea, tiempo estimado de completitud, responsable a cargo y fecha de entrega.
        1.6. Problemas encontrados: descripción detallada, intentos de solución sin éxito, soluciones encontradas con su descripción detallada, recomendaciones, conclusiones y bibliografía consultada para este problema específico.
        1.7. Conclusiones del proyecto.
        1.8. Recomendaciones del proyecto.
        1.9. Bibliografía consultada en todo el proyecto
    2. Bitácora en digital, donde se describen las actividades realizadas, desde reuniones con el compañero de trabajo, investigaciones, consultas, etc. Está se puede encontrar hecha a mano, se debe describir todo por más insignificante que sea, esto demostrará si ustedes están trabajando en realidad. Este es su diario de trabajo, llevan seguimiento de todo en el tiempo, imaginen que si un compañero los releva en su trabajo, le bastaría con leer sus bitácoras para seguir el trabajo.

7. **Evaluación**
    1. El proyecto tendrá un valor de un 70% de la nota final, debe estar funcional.
    2. La documentación tendrá un valor de un 20% de la nota final, cumplir con los requerimientos especificados en la documentación no significa que se tienen todos los puntos, se evaluará que la documentación sea coherente, acorde al tamaño del proyecto y el trabajo realizado, no escatimen en documentación.
    3. La defensa tendrá un valor de 10%, todos los integrantes del grupo deben participar.
    4. Cada grupo recibirá una nota en cada uno de los siguientes apartados Código, Documentación y defensa.
    5. El profesor no sólo evaluará la funcionalidad del proyecto, esto quiere decir que aunque el proyecto este 100% funcional esto no implica una nota de un 100, ya que se evaluarán aspectos de calidad de código, aplicación del **paradigma imperativo y orientado a objetos**, calidad de documentación interna y externa y trabajo en equipo.
    6. No se revisarán funcionalidades parciales, ni funcionalidades no integradas.
    7. Es responsabilidad de cada miembro del grupo conocer su código, el profesor puede preguntar a cualquier miembro del grupo que le explique alguna funcionalidad/porción de código.
    8. De las notas mencionadas en los puntos 1, 2 y 3 se calculará la Nota Final del Proyecto.

9. Las citas de revisión oficiales serán determinadas por el profesor durante las lecciones o mediante algún medio electrónico.
10. Aun cuando el código, la documentación y la defensa tienen sus notas por separado, se aplican las siguientes restricciones
    10.1. Si no se entrega documentación, automáticamente se obtiene una nota de 0.
    10.2. Si no se entrega el punto 1.3 de la documentación se obtiene una nota de 0.
    10.3. Si el código y la documentación no se entregan en la fecha indicada se obtiene una nota de 0.
    10.4. Si el código no compila se obtendrá una nota de 0, por lo cual se recomienda realizar la defensa con un código funcional.
    10.5. Si el grupo no cuenta con los equipos necesarios para realizar la revisión y no avisó al profesor de esta situación obtendrá una nota de 0.
    10.6. El código debe ser desarrollado en **el lenguaje de programación C** utilizando el **paradigma de programación imperativo** y java utilizando el **paradigma de programación orientado a objetos**, en caso contrario se obtendrá una nota de 0.
    10.7. **NO** presentarse a la defensa se obtendrá una nota de 0.
11. Cada grupo tendrá como máximo 30 minutos para exponer su trabajo al profesor y realizar la defensa de éste, es responsabilidad de los estudiantes mostrar todo el trabajo realizado, por lo cual se recomienda tener todo listo antes de ingresar a la defensa.
12. Cada excepción o error que salga durante la ejecución del proyecto y que se considere debió haber sido contemplada durante el desarrollo del proyecto, se castigará con 2 puntos de la nota final del proyecto.
13. Cada grupo es responsable de llevar los equipos requeridos para la revisión.
14. Durante la revisión únicamente podrán participar los miembros del grupo, asistentes, otros profesores y el coordinador del área.
15. Las revisiones se realizan con los estudiantes matriculados en el curso, cualquier persona fuera de estos y los mencionados en el punto 14, no pueden participar en la revisión.
16. Después de enviada la nota final del proyecto el estudiante tendrá un máximo de 3 días hábiles para presentar un reclamo siempre y cuando la funcionalidad esté completa.

8. **Referencias**

chuidiang.org. (2024, 5 21). *Socket entre C y java*. Retrieved from Socket entre C y java: http://www.chuidiang.org/java/sockets/cpp_java/cpp_java.php

Wikipedia. (2024, 5 21). *BreakOut (videojuego)*. Retrieved from https://es.wikipedia.org/wiki/Breakout_(videojuego)