Instituto Tecnológico de Costa Rica
Escuela Ingeniería en Computadores
Paradigmas de programación (CE3104)
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 y 5 Anexo)
   - ➔ Nombre código: **DonCEy Kong Jr**.
   - ➔ La tarea debe ser implementada en grupos de no más de 3 personas.
   - ➔ La **fecha de entrega** es: 21 de Noviembre de 2025.
   - ➔ Cualquier indicio de copia será calificado con una nota de 0 y será procesado de acuerdo al reglamento.

4. **Descripción del juego.**

Donkey Kong Jr es una secuela del video juego arcade, Donkey Kong, fue lanzado en 1982 y esta vez Mario tiene prisionero a Donkey Kong y su hijo un pequeño gorila llamado Donkey Kong Jr tiene que rescatarlo. En la primera escena Jr, tiene que ir trepando por lianas y caminando por plataformas evitando trampas animadas que le arroja Mario (parecidos a los kremlings, enemigos de los juegos de Donkey Kong Country), hasta colocarse en una plataforma al lado de Donkey Kong.

4.1. **El servidor:** mantendrá la "lógica del juego" y debe ser implementado en **Java**.
La creación de los "cocodrilos" será desde [consola/app] por un usuario administrador del juego y el decidirá en qué liana o plataforma se creará y qué tipo de cocodrilo es:
4.1.1. Rojos: Son aquellos que suben y bajan en una única liana y se mantienen en ella, no se caen.
4.1.2. Azules: Estos son los que eligen una liana y descienden de forma vertical y caen.

Si Donkey Kong Jr es alcanzado por algún Cocodrilo muere o si cae al abismo.

4.1.3. Creación de frutas: Las frutas en lugar de hacer perder vidas a Donkey Kong Jr estas le otorgan puntos, para crear una fruta se requiere se especifique la liana y la altura en la liana donde estará y la cantidad de puntos que otorga esa fruta en caso de que Donkey Kong Jr la tome.
4.1.4. Eliminación de Frutas: Se pueden eliminar frutas ya creadas, para lo que se requiere se indique la liana y altura en la liana de la fruta.
4.1.5. En nuestro Juego cuando Donkey Kong Jr llegue a salvar a Donkey Kong el juego le otorgara una vida y el juego inicia desde el principio, pero la velocidad con la que los cocodrilos rojos y azules suben y bajan es mayor.

4.2. **El cliente Jugador:** se refiere a la interfaz gráfica del juego debe realizarse **en C**, controla a Donkey Kong Jr, debe proveer las opciones de desplazarse (con direccionales del teclado) y brincar para colgarse

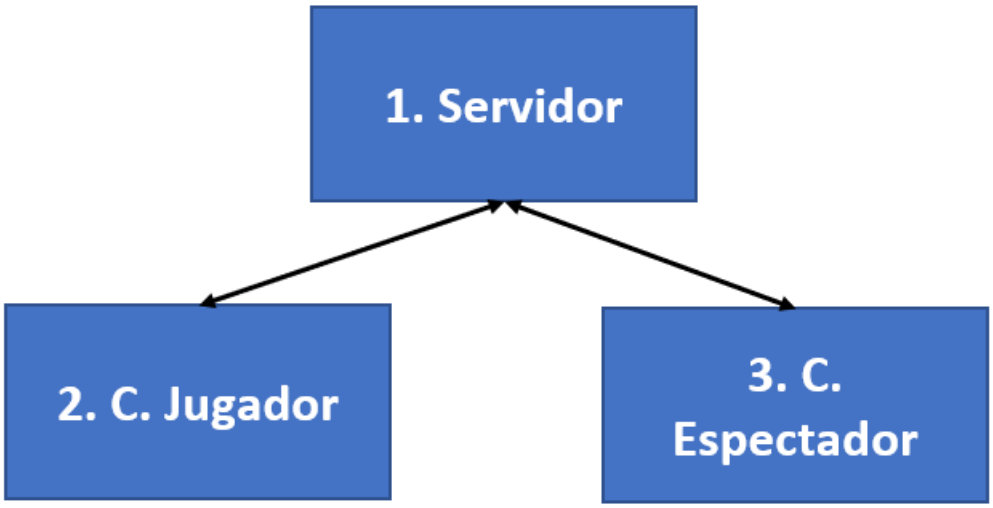
de las lianas. Máximo pueden existir 2 jugadores activos en diferentes consolas/apps NO en la misma pantalla.

4.6. **El cliente Espectador:** Se podrá unir a una partida existente pero sólo podrá observar lo que sucede. Debe realizarse en C. Máximo pueden existir 2 observadores por jugador.

4.7 **Conexión cliente servidor:** La conexión entre C y Java debe realizarse utilizando Sockets (chuidiang.org, 2022).

4.8 **Aspectos de implementación en C:** Recuerden se evaluará que todas las constantes estén en un archivo aparte. Se evaluará el uso de strutcs. **Se requiere un ejecutable para la revisión.**

4.9 **Aspectos de implementación en Java:** Recuerden se evaluará que todo esté programado según los conceptos de OO (Clases, paquetes, patrones al menos deben implementar 2 patrones excluyendo el singleton), No se permiten tipos de datos simples. **Se requiere un ejecutable para la revisión.**



5 Entregables

5.8 Código fuente comentado.

5.9 Manual de usuario.

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 realizarán 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 o digital, 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 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. (2025, 10 23). *Socket entre C y java*. Retrieved from Socket entre C y java: https://old.chuidiang.org/java/sockets/cpp_java/cpp_java.php

retrogames.cz. (2025, 10 23). *retrogames*. Retrieved from retrogames: https://www.retrogames.cz/play_002-NES.php