finis
Universidad Finis Terrae
Facultad de Ingeniería

PROYECTO PROGRAMACIÓN 2025-01

INGENIERÍA CIVIL INFORMÁTICA Y TELECOMUNICACIONES

[1] Contexto

Las carreras de caballos han sido, a lo largo de la historia, un evento que une tradición, emoción y competitividad. En muchas culturas, estos encuentros han sido motivo de celebración, donde la destreza del jinete y la capacidad del caballo se combinan en una experiencia llena de adrenalina y estrategia. Con la irrupción de la tecnología en todos los ámbitos de la vida, se abre una ventana para trasladar esta pasión al mundo digital, permitiendo que las emociones y la dinámica de las carreras se reproduzcan en un entorno virtual.

En este proyecto, se invita a los estudiantes a diseñar y desarrollar un software que simule una carrera de caballos en un hipódromo virtual. La idea es recrear la esencia de una competencia ecuestre, donde cada caballo, representado por sus características propias, compita en una pista definida con precisión. La simulación deberá reflejar aspectos reales, como la longitud de la pista, el número de competidores, y eventos aleatorios que puedan afectar el rendimiento de cada participante.

A través de esta iniciativa, se pretende que los estudiantes apliquen los conceptos básicos de programación —como el manejo de variables, estructuras de datos, condicionales y bucles— en un proyecto práctico y motivador. El simulador no solo permitirá experimentar con la lógica y la implementación de algoritmos, sino que también fomentará el trabajo en equipo, la resolución de problemas y la creatividad en el diseño de soluciones.

[2] Requerimientos

Para el desarrollo de este proyecto, deberán trabajar en equipos para entender los requerimientos, diseñar una solución, crear un software funcional en Python, probar que se ejecute correctamente, documentarlo y finalmente defender su implementación.

Toda la interacción del usuario deberá ser mediante la consola (no es necesario crear una interfaz gráfica), mediante ella, el usuario deberá poder configurar los parámetros de la carrera, crear los caballos participantes, recibir información de eventos relevantes que ocurran en la carrera y finalmente informar al caballo ganador.

[2.0] Respecto a la interfaz del usuario.

[2.0.1] El usuario debe poder navegar por las diferentes opciones del programa, sin que el programa termine de forma inesperada.

[2.0.2] Cada vez que el usuario ingrese una opción inválida, debe tener la opción de reintentar el ingreso de datos, sin que el programa termine de forma inesperada.

[2.0.3] El usuario debe tener la opción de simular una carrera las veces que estime necesario, sin que el programa termine una vez concluida la carrera.

[2.1] Respecto a la configuración de la carrera.

[2.1.1] El usuario debe tener la posibilidad de seleccionar el largo de la pista entre las siguientes opciones válidas : 1000m, 1100m, 1200m, 1300m, 1600m, 1700m, 1800, 2000m y 2400m.

[2.1.2] El usuario debe tener la posibilidad de seleccionar la cantidad de caballos participantes considerando un mínimo de dos caballos y un máximo de 16 caballos.

[2.1.3] El usuario debe tener la posibilidad de asignar valores a cada uno de los atributos de los caballos participantes. Estos atributos están detallados en la sección **[2.2]**.

[2.2] Respecto a la configuración de atributos de caballos.

[2.2.1] El usuario debe tener la posibilidad de configurar el nombre del caballo.

[2.2.2] El usuario debe tener la posibilidad de configurar la pista o carril por la cual correrá el caballo.

* *En caso de que esa pista ya haya sido asignada a otro caballo, se le deberá informar al usuario y solicitar que ingrese otra pista*.

[2.2.3] El usuario debe tener la posibilidad de configurar la velocidad base del caballo, siendo el rango válido entre 50 y 65 km/h.

* *A esta velocidad base, se le deberá sumar de manera "aleatoria" un mínimo de 0km/h hasta un máximo de 10 km/h antes de iniciar la carrera*.

* *Para esta tarea pude utilizar el módulo random de Python (generación de números pseudoaleatorios)*.

[2.2.4] El usuario debe tener la posibilidad de configurar la probabilidad de lesión del caballo. El rango válido de probabilidad de lesión puede estar entre un 5% y un 30%.

* *Considere que un caballo que sufre una lesión abandona la carrera de inmediato*.

* *Para esta tarea pude utilizar el módulo random de Python (generación de números pseudoaleatorios)*.

[2.3] Respecto al desarrollo de la carrera.

[2.3.1] Los caballos solo pueden avanzar en dirección a la meta, sin posibilidad de retroceder.

[2.3.2] Se deberá implementar un mecanismo para que periódicamente se informe al usuario las posiciones de cada caballo con la finalidad de que pueda realizar un seguimiento a la carrera.

[2.3.3] Se deberá implementar un mecanismo para que periódicamente los caballos puedan llegar a sufrir una lesión (en base a su atributo de probabilidad de lesión) y abandonar la carrera. En caso de ocurrir una lesión se deberá informar al usuario.

[2.4] Respecto al término de la carrera.

[2.4.1] La carrera finaliza en cuanto alguno de los caballos alcance la meta.

[2.4.2] Una vez terminada la carrera, deberá informar al usuario el caballo ganador.

[3] Entregables

El proyecto contempla cuatro componentes que serán evaluados: informe de avance, informe final, código funcional y presentación de defensa del proyecto.

[3.1] La primera entrega corresponde al 10% de la calificación final del proyecto. Considera el informe de avance del proyecto de la asignatura.

[3.2] La segunda entrega corresponde al 90% de la calificación final del proyecto. Considera el informe final del proyecto de la asignatura, el código funcional y la presentación de defensa del proyecto.

[3.3] Resumen de ponderaciones

[3.3.1] El informe de avance	10% de la calificación final del proyecto (grupal).
[3.3.2] El informe final	15% de la calificación final del proyecto (grupal).
[3.3.3] El código funcional	40% de la calificación final del proyecto (grupal).
[3.3.4] La presentación de defensa	35% de la calificación final del proyecto (individual y reprobatoria).

[3.4] Cada entrega posee una rúbrica de evaluación. Los archivos de cada una de las rúbricas se encuentran publicados en plataforma oficial (eFinis).

[3.5] El proyecto debe ser trabajado considerando los grupos informados por el docente de la sección. Los cambios en la conformación de grupos no están permitidos.

[3.6] Todas las entregas de un grupo, deberán ser gestionadas por un único integrante. En caso de que existan múltiples entregas por parte de un grupo, sólo se considerará la primera.

[4] Fechas de Entregas

[4.1] La primera entrega debe ser enviada al buzón “Proyecto – Entrega 1” en la plataforma oficial (eFinis) a más tardar el día lunes 28 de abril del 2025 a las 08:30.

[4.2] La segunda entrega debe ser enviada al buzón “Proyecto – Entrega 2” en la plataforma oficial (eFinis) a más tardar el día lunes 09 de junio del 2025 a las 08:30.

[4.3] Las presentaciones de defensa de proyecto, se realizarán en bloques de laboratorio o cátedra, definición que será informada con anticipación por el docente de la sección. La fecha de esta evaluación (en base al syllabus de la asignatura) se encuentra planificada inicialmente para la semana del 09 al 13 de junio del 2025.

[4.4] Cualquier entrega que se reciba fuera del plazo establecido o a través de un canal distinto al oficial (eFinis) no será considerada y se asignará la calificación mínima 1.0.

[5] Reglamento

[5.1] En caso de determinar copia en el desarrollo del proyecto entre dos grupos, ambos grupos serán evaluados con la calificación mínima 1.0.

[5.2] El uso de herramientas de IA generativa está estrictamente prohibido, así como utilizar aspectos no tratados en la asignatura y/o cualquier acción que represente una falta al “Reglamento de Responsabilidad Académica y Disciplinaria” de la Universidad Finis Terrae (ver título II).