Trabajo Integrador 2022

Seminario de Lenguajes Opción Python (versión corregida)

El trabajo integrador de este año consta del desarrollo de un juego educativo basado en tarjetas. El desarrollo de este trabajo involucra el análisis de los datos que alimentarán al juego, el desarrollo de una interfaz gráfica simple para jugar y el análisis de los datos generados por el juego.
El desarrollo del trabajo se divide en **2 etapas** que se entregarán de acuerdo al cronograma publicado.
Este documento describe el trabajo a realizar y luego hacia el final detalla los requerimientos para cada entrega.

FiguRace: carrera de datos

El juego consiste principalmente en una pantalla donde se le presenta una tarjeta con una serie de datos al jugador. Estos datos representan características de la tarjeta, todos los datos identifican a un determinado objeto o persona. El objetivo del jugador es identificar el objeto o persona a la que hace referencia la tarjeta en un tiempo configurable. El jugador acumula puntos por cada acierto y pierde puntos si no responde en el tiempo dado o si responde de forma errónea. Luego de cada ronda se muestra al jugador una nueva tarjeta y los datos asociados a la misma.
El juego termina luego de un número determinado de tarjetas o cuando el jugador haga clic en el botón salir. Cuando esto sucede se le mostrará al jugador la pantalla de puntos en un popups y luego la pantalla “Menú de inicio del juego”.

Pantallas

Ver Anexo I para más detalles

- (a) Menú de inicio del juego
- (b) Pantalla de juego
- (c) Pantalla de creación/edición de perfil
- (d) Pantalla de puntajes
- (e) Pantalla de configuración

Carga de datos para el juego

Las tarjetas de FiguRace se generan a partir de un conjunto de datos que deberán ser preparados previamente. Como parte del proyecto se deberá desarrollar un cuaderno de Jupyter Notebook que extraiga los datos necesarios para el juego de alguno de los datasets provistos y muestre en forma detallada el proceso realizado con los mismos . En el Anexo II se detallan los datasets que deberán utilizar para la aplicación y las modificaciones necesarias para luego guardarlo en un nuevo archivo en formato CSV. El juego deberá cargar esos archivos CSV para funcionar.

Consideraciones generales de implementación

El programa principal se debe llamar **figurace.py**.
El software debe correr correctamente tanto en Windows como en Linux.
Se deberá armar una estructura de directorios para organizar los archivos en carpetas y subcarpetas de manera tal que sea clara su corrección. Las mismas se subirán a un repositorio de GitLab determinado por la cátedra en donde se deberá incluir además un archivo denominado **README.md,** que incluya el nombre de los integrantes, la forma de ejecutar cada aplicación, el sitio donde obtuvo los dataset, un archivo con los requerimientos de librerías que utiliza y cualquier consideración especial para su ejecución.
En todos los casos se deberán utilizar imágenes o sonidos, libres o propios, que puedan ser publicados con licencias libres y especificar el sitio donde las obtuvo[1].
El juego tendrá la posibilidad de configurar al menos 3 niveles de dificultad que deberán programarse en la "(e) Pantalla de configuración", las opciones de cada nivel están detalladas en el Anexo, los posibles valores de estas opciones deberán ser definidos por el grupo. Las columnas (características) de cada dataset deberán organizarse en función de su importancia. Es decir, se podrá modificar el orden de ubicación de las mismas de acuerdo a la prioridad de su uso para el armado de las tarjetas en el juego.

Para el desarrollo de la aplicación se deberá:

- Utilizar **PySimpleGUI** para las interfaces gráficas.
- Generar los archivos CSV que se requiera.
- Documentar el código usando docstrings en las funciones y clases.
- Incluir un archivo LICENSE con los nombres de los autores y la licencia.
- Tener en cuenta las guías de estilo de Python para la escritura del código.
- Definir una estructura de carpetas que permita estructurar el código de forma prolija.

Consideraciones de la entrega y defensa

Si bien el trabajo es grupal, **la defensa es INDIVIDUAL.**
La defensa se realiza dentro del horario de consulta inmediatamente posterior a la fecha de entrega. La misma consiste en tener un encuentro virtual o presencial con el ayudante asignado donde el ayudante va a realizar una serie de preguntas sobre la entrega. Estas preguntas serán destinadas a los distintos miembros del grupo. La idea es que se distribuyan las preguntas entre todos los miembros teniendo cada uno su espacio para responder. El desempeño en las respuestas de las mismas es el que define finalmente la nota de la entrega para cada integrante.
Esto quiere decir que por más que sea una entrega grupal, las notas entre los miembros de los grupos pueden ser diferentes.
Es importante también tener participación activa en las consultas prácticas y en el repositorio de GitLab para que el ayudante tenga un panorama conceptual del conocimiento de cada participante del grupo.

[1] Para buscar este tipo de material se pueden usar sitios como openclipart.org, Freepik o Google con filtros para buscar imágenes libres

Primer Entrega

Funcionalidades pedidas:

- Implementación de un Jupyter Notebook que permita convertir los datasets originales con datos del juego (por ejemplo características de formaciones geológicas) en archivos CSV que alimentarán al juego con las opciones para las tarjetas. Ver Anexo II donde se detallan las modificaciones necesarias para cada uno.
- Implementación de la interfaz de todas las pantallas. A modo prototipo todas las ventanas deberán tener los widgets (botones, campos de texto, gráficos, etc…) necesarios para la realización del trabajo (pantallas descritas en el Anexo I).
- Se deberán implementar las funcionalidades de las “Pantalla de configuración” y creación/edición de perfil.
- La “Pantalla de configuración” deberá almacenar/cargar los datos desde un archivo JSON. Se requiere la configuración de al menos 3 niveles (detalles en Anexo I - (e) Pantalla de configuración).
- La pantalla de creación/edición de perfil deberá almacenar/cargar los datos en/desde un archivo JSON (detalles en Anexo I - (c) Pantalla de creación/edición de perfil).
- El resto de las pantallas deberán tener la funcionalidad mínima para permitir navegar entre ellas:
    - Los botones del “Menú de inicio del juego” deben permitir mostrar las otras pantallas.
    - Todas las pantallas deberán tener un botón “Volver/ Salir/ Abandonar” que regresa al “Menú de inicio del juego”.
- La pantalla “Menú de inicio del juego” deberá cargar el listado de jugadores en base a los jugadores que se vayan creando en “Pantalla de creación/edición de perfil” y permitir elegir con qué perfil se quiere jugar.
- La “Pantalla de juego” deberá mostrar (detalles en Anexo I - (b) Pantalla de juego):
    - Una tarjeta con la información para una de las filas de uno de los datasets.
    - Dicha información corresponde a una serie de características asociadas al elemento a adivinar.
    - y una selección aleatoria de 4 opciones de respuesta incorrectas tomadas del archivo y mezclada con la respuesta correcta.

No es necesario implementar ninguna otra funcionalidad en esta etapa (como pasar a otra tarjeta, completar las opciones, la cuenta regresiva, etc) excepto volver a “Menú de inicio del juego”.

Fechas de entrega

La fecha de entrega es la semana del **1 de Junio y la defensa individual semana del 6 de Junio en horario de la consulta práctica.**

Criterios de evaluación

- Funcionalidad implementada de acuerdo al enunciado.

- Cumplimiento de las consideraciones planteadas.
- Código subido al repositorio de GitLab indicado.
- Participación durante el desarrollo del trabajo. Esto incluye tanto la asistencia a las consultas (virtual o presencial) y las contribuciones en el repositorio de GitLab
- En la defensa deberán demostrar los conocimientos utilizados para la realización del trabajo.

Puntos de la primera entrega (en la defensa individual)

- **Puntos de cursada:** 30 puntos.

Segunda Entrega

Si bien para esta primera entrega no es necesario desarrollar todo el juego, en esta sección les describiremos en forma general las funcionalidades que se solicitarán para la siguiente.

Se deberán completar todas las funcionalidades del juego, se pedirá:

- Implementación de la funcionalidad restante del juego tal cual se pide en el enunciado y los anexos.
- Usar librerías avanzadas de Python para procesar los distintos datos del juego.
- Se registrarán y guardarán los datos estadísticos del juego como se indica en el enunciado general.
- Sumar complejidad a la lógica o interfaz del juego.

Esto es sólo una descripción general. La segunda entrega tendrá su propio enunciado con el detalle de lo qué se debe implementar.

Anexo I

Pantallas

(a) Menú de inicio del juego

En esta pantalla se presenta el menú principal de FiguRace que permite a los jugadores acceder al resto de las pantallas o salir del juego.
Para acceder al juego cada jugador debe elegir su **nick** (nombre) desde un menú desplegable, el nivel de dificultad con el que jugará y luego hacer clic en el botón correspondiente para iniciar el juego.

La Figura 1 muestra, a modo de ejemplo, una posible distribución de estos elementos.

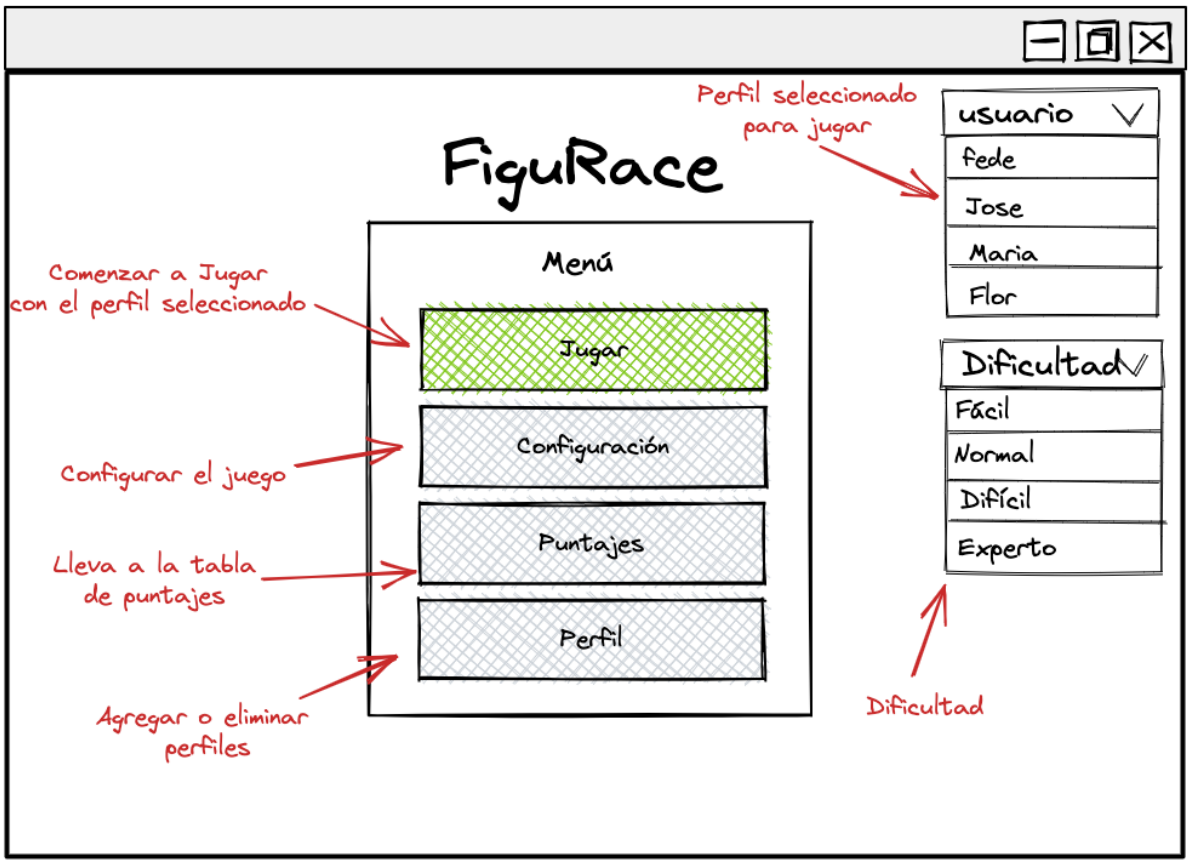

FIgura 1: Menú de inicio del juego

(b) Pantalla de juego

Esta es la pantalla donde se desarrolla el juego. El juego tiene un determinado número de rondas, en cada ronda se muestra al jugador:

- Una tarjeta con datos de 5 características que permiten identificar al objeto o persona al que pertenecen.

- 5 opciones donde sólo una de las opciones es correcta y el resto son opciones aleatorias. En cada opción se puede hacer clic para que el juego determine si la respuesta es correcta o no y se pase a la siguiente ronda.
- Una cuenta regresiva donde se muestran los segundos restantes para responder. Al finalizar la cuenta regresiva se da por perdida la ronda y se pasa a la siguiente.
- El puntaje actual hasta el momento.
- Un botón “Pasar” para saltear una ronda. Al “pasar” una ronda se da por perdida y se pasa a la siguiente.
- Un botón para volver al menú de inicio. Al volver al menú inicio se dan por perdidas la ronda actual y las rondas restantes.
- Al terminar se deberá mostrar un popup o ventana con el puntaje final calculado.

La Figura 2 muestra una posible pantalla con esta funcionalidad.

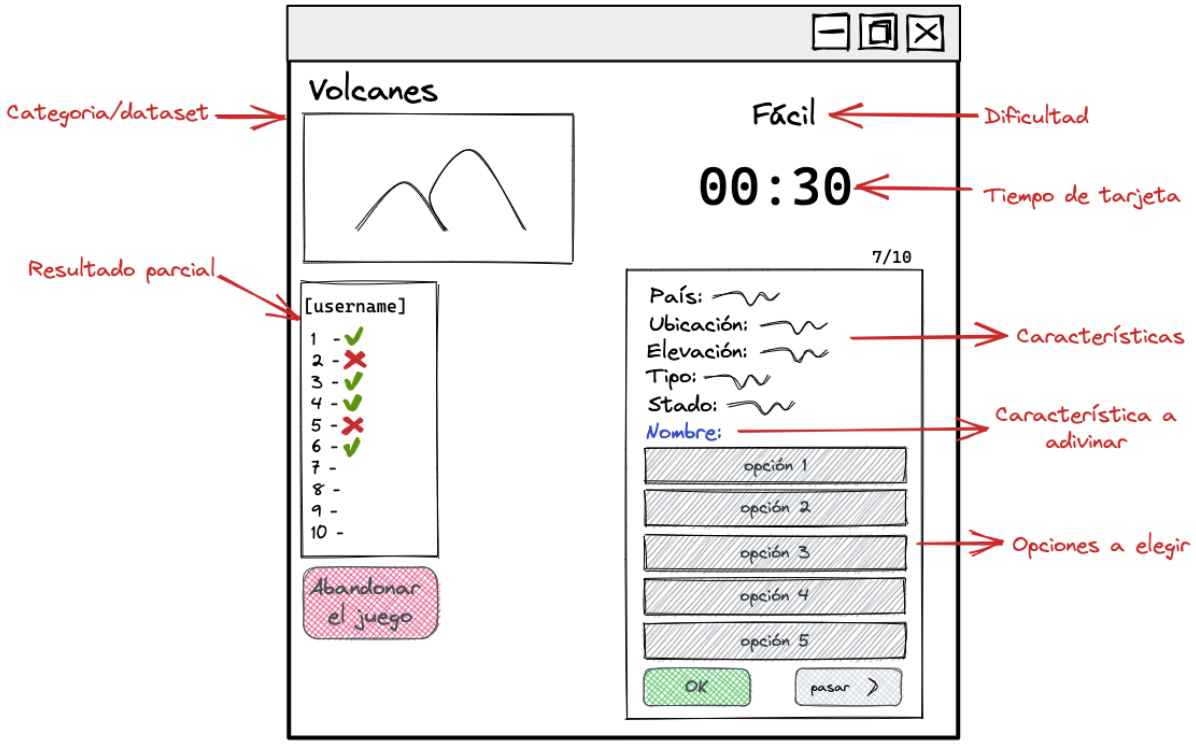

Figura 2: Pantalla del juego

(c) Pantalla de creación/edición de perfil

Esta pantalla cumple una doble función permitiendo crear un nuevo perfil de jugador o editar uno existente. Se solicitará el **nick** (nombre) de cada jugador (que debe ser único), **edad**, **género autopercibido.** Un jugador ya creado no puede modificar su nick. Estos datos se solicitan con el objetivo de generar estadísticas.

(d) Pantalla de puntajes

Mostrará los datos de los 20 mejores puntajes por cada nivel junto al nombre del jugador que lo obtuvo.

(e) Pantalla de configuración

El juego cuenta con una configuración modificable desde esta pantalla. La configuración es global para los jugadores y se almacenará en un archivo de formato JSON. Las opciones disponibles serán:

- Tiempo límite por ronda: un número de segundos.
- Cantidad de rondas por juego.
- Puntaje sumado por cada respuesta correcta.
- Puntaje restado por cada respuesta incorrecta.
- Cantidad de características a mostrar según el nivel. Si el número es 3 toma las primeras 3 columnas o características del dataset. Por este motivo, se debe considerar el orden de las columnas en que se guardaron los datos. Esto significa que las columnas/características del dataset deben estar ordenadas por orden de importancia.

Anexo II

Datasets

Existen muchas páginas en Internet que proveen bases de datos (descargables como archivos de texto fáciles de procesar) con datos sobre distintos temas. Estas bases de datos generalmente se publican para hacer análisis sobre esos datos o para utilizarlos con algún algoritmo de Machine Learning, por lo cuál suelen estar en formato CSV o JSON para ser leídos con facilidad.

En este trabajo cada grupo deberá elegir al menos **dos datasets del grupo A y** al menos **un dataset del grupo B** para generar las tarjetas del juego.

El grupo deberá desarrollar un cuaderno Jupyter con el código necesario para leer el dataset elegido y:

- Eliminar las columnas que no son pedidas.
- Aplicar las transformaciones y limpieza de datos pedida para el dataset elegido.
- Generar las columnas en el orden pedido en el enunciado.

Datasets elegibles

Grupo A

1) Top 100 de temas musicales de Spotify 2010 a 2019

https://www.kaggle.com/datasets/muhmores/spotify-top-100-songs-of-20152019

Deberá adaptar los datos de la siguiente forma:

- Poner en “title case” los géneros musicales excepto las siglas EDM, DFW, UK, R&B y LGBTQ+ que deben ir en mayúsculas. Por ejemplo “dfw rap” debe ser transformado a “DFW Rap”.
- Considerar también la excepción “k-pop” que debe ser transformada a “K-Pop”.
- Se utilizarán como datos de las tarjetas “Top Genre”, “Year Released”, “BPM”, “Top Year” y “Artist Type”. Como dato a adivinar se utilizará “Artist”. Descartar el resto de las columnas.
- El archivo resultante deberá tener las siguientes columnas (en este orden específico): “Top Genre”, “Artist Type”, “Year Released”, “Top Year”, “BPM” y “Artist”

2) Erupciones volcánicas

https://public.opendatasoft.com/explore/dataset/significant-volcanic-eruption-database/table/

Deberá adaptar los datos de la siguiente forma:

- Traducir el tipo de volcán al español.
- Convertir “Flag Tsunami” y “Flag Earthquake” a booleanos, considerando que las celdas vacías corresponden al valor ***False***.
- Transformar la columna “Name” en una nueva columna “Name and country” para que tenga una combinación del nombre del volcán con el país entre paréntesis. Por ejemplo el volcán “Caldera” que se encuentra ubicado en Ecuador tendrá el valor “Caldera (Ecuador)” en la columna “Name and country”.
- Se utilizarán como datos de las tarjetas “Year”, “Flag Tsunami”, “Flag Earthquake”, “Volcanic Explosivity Index” y “Volcano Type”. Como dato a adivinar se utilizará “Name and country”. Descartar el resto de las columnas.
- El archivo resultante deberá tener las siguientes columnas (en este orden específico): “Year”, “Volcanic Explosivity Index”, “Volcano Type”, “Flag Tsunami”, “Flag Earthquake” y “Name and country”.

3) Lagos

https://drive.google.com/file/d/1PzfCgiAhPAq8Cztx9psk3puxDPHsjnqJ/view?usp=sharing
(extraído de https://www.ign.gob.ar/NuestrasActividades/Geografia/DatosArgentina/Lagos)

Deberá adaptar los datos de la siguiente forma:

- Transformar las coordenadas en la columna “Coordenadas” a grados decimales.
- Se utilizarán como datos de las tarjetas: “Ubicación”, “Superficie (km²)”, “Profundidad máxima (m)”, “Profundidad media (m)”, “Coordenadas”. Como dato a adivinar se utilizará “Nombre”. Descartar el resto de las columnas.
- El archivo resultante deberá tener las siguientes columnas (en este orden específico): “Ubicación”, “Superficie (km²)”, “Profundidad máxima (m)”, “Profundidad media (m)”, “Coordenadas” y “Nombre”.

Grupo B

1) Películas

https://www.kaggle.com/datasets/disham993/9000-movies-dataset

Deberá adaptar los datos de la siguiente forma:

- Descartar las películas que no tienen “overview”.
- Descartar las películas cuyo idioma original tenga más de 2 caracteres.
- Tomar las 100 palabras más comunes de todos los overviews combinados.
- Para generar el ”Overview” de cada película para el archivo de salida se deberán descartar las palabras que se encuentren dentro de las 100 más comunes de todos los overviews y se deberá generar un string con 3 palabras tomadas al azar y ordenadas al azar (ver random.sample()) de ese resultado. Nota: luego de filtrar las palabras comunes algunos overviews pueden tener menos de 3 palabras, en esos casos se tomarán las palabras que se pueda.

- Se utilizarán como datos de las tarjetas: “Genre”, “Original_Language”, “Vote_Average”, “Release_Date” y 3 palabras aleatorias del overview separadas por “;”. Como dato a adivinar se utilizará “Title”. Descartar el resto de las columnas.
- El archivo resultante deberá tener las siguientes columnas (en este orden específico): “Genre”, “Original_Language”, “Release_Date”,“Vote_Average”, “Overview” y “Title”.

2) FIFA 2021 Complete Player Dataset

https://www.kaggle.com/datasets/aayushmishra1512/fifa-2021-complete-player-data?resource=download

Deberá adaptar los datos de la siguiente forma:

- Reemplazar “Potential” por la siguiente escala conceptual:
  - Regular: Menos de 60
  - Bueno: Entre 60 y 79 (inclusive)
  - Muy bueno: Entre 80 y 89 (inclusive)
  - Sobresaliente: Desde 90 en adelante.
- Reemplazar el valor de “Position” por las posiciones en español. Por ejemplo “LB|CB” debe ser reemplazado por “Defensor izquierdo|Defensor central”
- Se utilizarán como datos de las tarjetas: “Age”, “Nationality”, “Position”, “Team” y “Potential”. Como dato a adivinar se utilizará “Name”. Descartar el resto de las columnas.
- El archivo resultante deberá tener las siguientes columnas (en este orden específico): “Team”, “Nationality”, “Position”, “Age”, “Potential” y “Name”.

Las posiciones son siglas que se pueden encontrar en https://en.wikipedia.org/wiki/Association_football_positions