TUIA - ENTORNO DE PROGRAMACIÓN
PRÁCTICA INTEGRADORA

- Generar un usuario y contraseña en un proveedor de hosting de para control de versiones, GitHub o GitLab a elección de cada estudiante.

- Crear un repositorio edp-practica-integradora-*apellido* (reemplazando *apellido* por el tuyo) y clonarlo localmente.

- Comenzar a desarrollar los siguientes puntos, utilizando git para versionar nuestros cambios, siguiendo las mejores prácticas:
    - Utilizar un branch distinto para cada ejercicio. Fusionar las ramas con la rama principal a medida que los ejercicios estén completos.

      * En los mensajes de commit utilizar siempre *ejercicio-x* como prefijo para identificar rápidamente con que ejercicio está relacionado cada commit. Al lado un pequeño título describiendo lo principal que agregaron o cambiaron, luego una línea en blanco y debajo toda la información que consideren útil para las personas que exploren la historia del proyecto (en este caso, los profes).

    - Realiza un push al repositorio por cada commit que hagas.

- Programar un script **forma_antonimos.sh** que acepte por entrada estándar (STDIN) un adjetivo y devuelva su antónimo agregando in- o im- según corresponda (Asumir que el antónimo siempre se forma así).
  Ejemplos:
  Audito -> Inaudito
  Posible -> Imposible
  CoMPRenSiBLe -> Incomprensible

- Programar un script **agrega_prefijo.sh** que tome una cadena como argumento y luego espere palabras por STDIN y para cada palabra la muestre prefijada con la cadena que tomamos argumento. Debe seguir esperando palabras hasta obtener el EOF (Ctrl+D)
  Ejemplos suponiendo que pasamos como argumento “pre”:
  histórico -> prehistórico
  viaje -> previaje

- Programar un script **ordena_numeros.sh** que reciba números enteros en la entrada estándar (validando que son números enteros con una expresión regular) y lo escriba, en orden, en el archivo salida-c.txt

- Escribir un pequeño Dockerfile, basándose en la imagen `ubuntu:latest` y mediante la inclusión en el contenedor del script **navegador.sh** y un comando ENTRYPOINT apropiado. La imagen resultante debe llamarse **tuia:develop**. Commitear el Dockerfile al repositorio.

- Trabaja sobre el repositorio de un compañero. Para ello, deberás poder clonarlo. Luego haz un branch `menu` y realiza lo siguiente:
    - Programar un script, utilizando la instrucción SELECT para ofrecer un menú de opciones que permite elegir entre los scripts anteriores, en caso de necesitar argumentos los pide y luego llama al script correspondiente.
    - Pushear la branch al repositorio de nuestro compañero y que nuestro compañero revise el script y fusione la rama con la principal

- Modificar el Dockerfile para incluir todos los scripts y que el ENTRYPOINT sea el menú. ¿Cuántas capas conviene usar? Committear las modificaciones.

- Agregar un archivo README.md[1] que incluya instrucciones para construir la imagen (docker build…) y para correr el contenedor (docker run …)

- Al elegir la opción **ordena_numeros.sh** el archivo resultante, ¿queda dentro o fuera del contenedor? ¿Cómo podemos resolver este problema? “Montar” el archivo salida-c.txt a un archivo en el host para solucionar este problema. Modificar Dockerfile y README.md si fuera necesario y committearlos.

- Al finalizar estos ejercicios, realizar un tag en el repositorio llamada v1.0.0 y pushear el tag. Así mismo, crear una imagen de docker taggeada como v1.0.0

[1] Un archivo README.md contiene información sobre los otros archivos del directorio, y es una forma rudimentaria de documentación de proyectos. Incluir este archivo, aunque sea con la información básica del proyecto se convirtió en un estándar de facto y de hecho, herramientas como GitHub o GitLab presentan la información contenida en este archivo en la página principal del proyecto.