TRABAJO FINAL

Comentarios iniciales

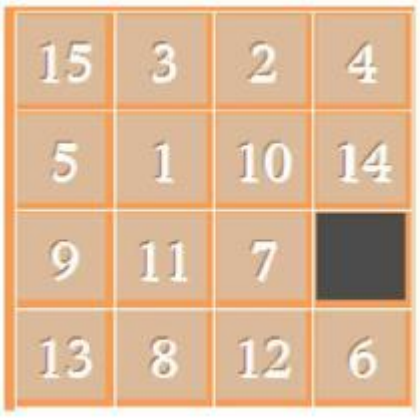
El trabajo debe ser entregado **al menos 2 días (hábiles) previos a la mesa examinadora**.

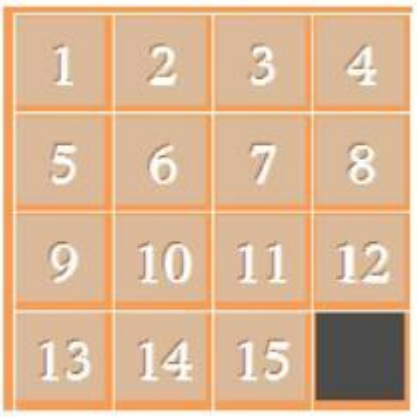
El trabajo puede realizarse de manera individual o bien en equipos **de hasta máximo 2(dos) programadores**, sin excepción. En este caso, ambos participantes del equipo deben conocer claramente la totalidad del código y su propósito, poder explicarlo y saber cómo modificar el mismo con fines correctivos o para el agregado de nuevas funcionalidades.

Cabe mencionar que el trabajo práctico requiere más esfuerzo dependiendo de la cantidad de programadores que trabajen en equipo, será requerido resolver un mayor número de consignas.

Se ***exige*** que el código sea acompañado con la **documentación solicitada**, sólo de esa manera será evaluado el trabajo práctico en su totalidad.

PUZZLE 15





El **Puzzle 15**, o **Solitario del 15**, fue creado en el siglo XIX y es un juego de deslizamiento, donde en una grilla de dimensiones 4x4, se disponen aleatoriamente 15 piezas numeradas, sin repetir, del 1 al 15, quedando un espacio vacío.

Se parte de una configuración aleatoria (arriba a la izquierda) y el **objetivo del juego (solución)** es ordenar dichas piezas logrando una disposición final (arriba a la derecha).

El **espacio vacío** sólo puede ocuparse por una de las fichas vecinas o adyacentes, que comparten una coordenada horizontal o vertical (no una vecina en una posición diagonal), es decir, si observamos la figura de la izquierda, el espacio vacío en la posición (fila3, columna4), sólo puede ocuparse por las piezas 14, 7 o 6 (generando un nuevo y único espacio vacío al desplazarse), pero no puede ocuparse por las piezas 10 o 12 (diagonales) ni cualquier otra.

Partida

En el programa, deberá presentarse al participante una disposición de las piezas y permitir que *de manera sencilla* pueda realizar los movimientos de desplazamiento. Se advierte que no toda configuración inicial para las piezas es “solucionable”. Casi la mitad de las configuraciones iniciales posibles no admiten solución, es por ello que deberán generar un set de “inicios de partidas” y tomar del mismo una para presentar al participante, considerando no repetir la configuración durante una ronda de juego.

Pueden tomar configuraciones de aquí: https://15puzzle.netlify.app/ presionando “new game”.

A medida que el participante juegue debe mostrarse en la pantalla la cantidad de movimientos que ejecutó hasta el momento.

En cada movimiento se debe constatar si la configuración que consigue en la grilla es solución o no del juego, en caso de ser solución notificarlo y finalizar la partida.

Apuesta

Debe agregarse al inicio de cada partida una instancia donde el participante pueda apostar que va a resolver el puzzle en, por ejemplo, M movimientos (M debe ser un número entre 40 y 200, inclusive)

Al finalizar la partida debe mostrarse el puntaje obtenido:

- 500 puntos si lo hace en una cantidad de movimientos dentro del rango [M-10, M
- 100 puntos si lo logra en (M, M+10] movimientos
- 1000 puntos si lo hace en exactamente M movimientos
- 0 puntos en otro caso

Continuar jugando

Al finalizar la partida debe ofrecerse al participante si quiere realizar una nueva partida dentro de la ronda antes de cerrar el programa.

Entre cada partida deberá mostrarse su historial de apuestas y resultados, de la última y las partidas anteriores, considere que en una ronda puede jugar un máximo de 10 partidas.

Partida 1: puntos PPP
Partida 2: puntos PPP
. . .

También debe presentarse la opción de abandonar la partida cuando lo desee, en ese caso obtendrá para esa partida 0 puntos.

En caso de realizar el TP sólo (un programador)

- Crear un programa que cumpla con todas las consignas indicadas anteriormente

- Entregar la documentación sin excepciones

En caso de realizar el TP en equipo de dos programadores

- Mismas tareas que el caso anterior de único programador.

- Contabilizar el tiempo transcurrido en cada partida.
  Mostrar junto con las estadísticas de las apuestas el tiempo transcurrido para cada partida.

- Investigar y proponer (no es obligatorio programarlo) cómo haría para que en el juego, superando una cantidad preestablecida de minutos, la partida termine, es decir incluir el modo de juego *contrarreloj*.

Recomendaciones finales

- Analizar el enunciado atentamente y prestar atención a los detalles, requerimientos y valores
- Definir el conjunto de variables y funciones necesarias
- Diseñar el algoritmo antes de programar, verificando que los requerimientos pueden satisfacerse
- Comenzar con la documentación en etapas tempranas de la implementación, no dejarlo para último momento, recordar que es un requerimiento la entrega de la misma.
- Escribir código claro y comentar pertinentemente
- Considerar el uso de macros y funciones, discernir cuándo es conveniente cada herramienta.
- Realizar pruebas parciales del programa para depurarlo
- Exponer a diferentes pruebas al programa final y revisar que no se reciban errores ni warnings durante la compilación.
- Asegurarse que al momento de enviar el programa, se trate de su versión final
- Al trabajar en equipo, comunicarse activamente y no desentenderse de los avances del resto de los integrantes del equipo, considerar el uso de un sistema de control de versiones.