TAREA Nº2

Evaluación de todas las Unidades del Curso: Fundamentos de Programación en Python, Programación Modular, Uso de Objetos

Fecha de envío: Domingo 12 de julio de 2020, 23:59 hrs.
Modalidad: Trabajo en grupos de **dos** personas.
Horario de Consultas: El que tu profesor te indique.

I. Objetivo.

El objetivo de la presente tarea es evaluar tu capacidad para:

1. Llevar a cabo un programa completo en el lenguaje Python, utilizando todos los elementos básicos que provee el lenguaje, y sus tipos de datos básicos y compuestos (*strings* y listas).
2. Manejar la clase archivo, en particular de texto, para leer información desde ellos.
3. Resolver un problema, usando un diseño funcional, considerando que se cuenta con el procedimiento en lenguaje natural, que se debe implementar para resolverlo.
4. Seguir en forma precisa la interfaz solicitada.

II. Contexto.

La siguiente tarea es una **generalización** del trabajo que hiciste en la Tarea 1 resolviendo el juego del ***BuscaMinas***. Esta vez, deberás generar el tablero a partir de la información establecida en un archivo de texto y hacer un diseño funcional para la solución de esta tarea. Es decir, deberás construir un programa que, a través de funciones con una labor precisa y clara, y la coordinación entre ellas, la resuelvan.

III. Enunciado.

Al ejecutar tu programa, deberá presentar al usuario el menú mostrado en la figura 1. A continuación, se explican cada una de estas opciones.

```
Escoge una opción: (1) Generar tablero (2) Cargar tablero (3) Salir:
```

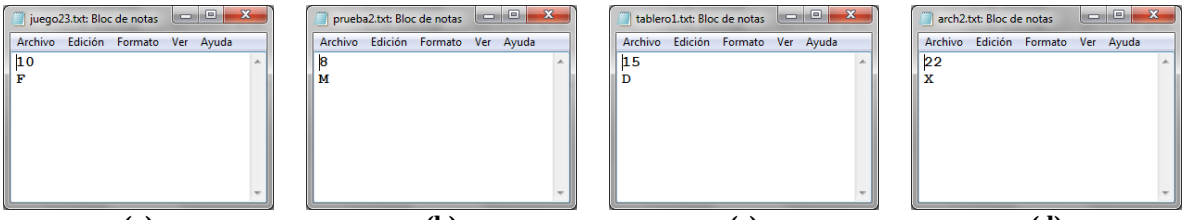
Figura 1. Mensaje inicial que debe tener el programa solicitado.

III.1 Opción 1 “Generar tablero”:

Tu programa deberá pedir el nombre del archivo (figura 2) donde está especificado: la dimensión del tablero (en la primera línea) y la dificultad (en la segunda línea) (ver ejemplos en la figura 3). La **dimensión** establece el número de filas y de columnas que deberá tener el tablero (siempre será cuadrado).

```
Escoge una opción: (1) Generar tablero (2) Cargar tablero (3) Salir: 1
Ingresa el nombre del archivo: juego23.txt
```

Figura 2. Mensaje solicitando nombre del archivo de entrada, asociado a la opción 1 "Genera tablero".



(a) **(b)** **(c)** **(d)**

Figura 3. Ejemplos de archivos de entrada usados para la opción **"Generar Tablero"**. **(a)** Tablero de 10×10 casillas con 10% de minas (10 minas en total). **(b)** Tablero de 8×8 casillas con 15% de minas (9 minas en total). **(c)** Tablero de 15×15 casillas con 20% de minas (45 minas en total). **(d)** Tablero de 25×25 casillas con 30% de minas (187 minas en total).

La **dificultad** será una de estas letras: "**F**", "**M**", "**D**" o "**X**", y corresponde al número de minas que deberá contener el tablero. Las posibles dificultades y el porcentaje de casillas que deberán ser minas son:

- Si la dificultad es **F**ácil (**F**), y debe tener un **10%** de minas.
- Si la dificultad es **M**edio (**M**), y debe tener un **15%** de minas.
- Si la dificultad es **D**ifícil (**D**), y debe tener un **20%** de minas.
- Si la dificultad es e**X**perto (**X**), y debe tener un **30%** de minas.

La forma de calcular el número de casillas que deberán ser ocupadas por minas dentro del tablero es:

*Número de Minas = Entero Inferior(Número Total de Celdas * Porcentaje de Casillas)*

En de tu programa deberás establecer en forma aleatoria la ubicación de las minas dentro del tablero.

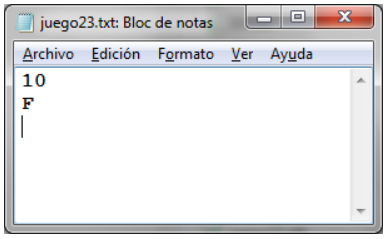
Guardado del tablero generado:

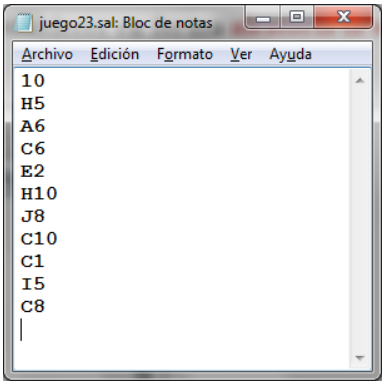
Luego de establecer la ubicación de las minas, deberás almacenar la información en un archivo de texto, que **se debe llamar igual al archivo de entrada**, pero deberás **cambiar la extensión** de "**.txt**" a "**.sal**". Este cambio permitirá distinguir que se trata de un "archivo de **sal**ida".

Lo que debes almacenar en este archivo y la forma de hacerlo es (figura 4):

- En la primera línea: dimensión del tablero.
- En cada una de las siguientes líneas: coordenada de cada mina (letra de la fila seguida por el número de la columna donde se ubica)

Luego de generar el archivo de "salida", tu programa deberá terminar su ejecución.





(a) **(b)**

Figura 4. (a) Archivo de entrada **juego23.txt**, que indica que se debe generar un tablero de 10x10, y con 10% de casillas con minas, es decir 10 minas en total (dificultad **F**ácil). **(b)** Archivo de salida **juego23.sal** generado: indica que corresponde a un tablero de dimensión 10 (primera línea), y enumera las coordenadas de las 10 minas que se deben ubicar aleatoriamente en él.

III.2 Opción 2 "Cargar tablero":

En este caso, tu programa deberá pedir el nombre del archivo donde está almacenada la información del tablero concreto que se jugará. Este archivo **debe tener el formato del archivo de salida** explicado en el punto anterior. Luego, deberá generar internamente el tablero con las minas especificadas, **e iniciar el juego del *BuscaMinas*, siguiendo las mismas reglas explicadas en la Tarea 1** (ver un ejemplo de inicio de juego en la figura 6).

Impresión del tablero:

La impresión del tablero es **casi** la misma que la usada para la Tarea 1: En esta oportunidad, deberán existir 2 espacios entre cada columna que se imprime. Esto se debe a que la dimensión del tablero podría ser mayor a 9 (y entonces usar 2 dígitos), por lo que se requiere más espacio para ellas (ver **figura 5 y figura 6**).

```
  1  2  3  4  5
A .  .  .  .  .
B .  .  .  .  .
C .  .  .  .  .
D .  .  .  .  .
E .  .  .  .  .
```

(a)

			1			2			3			4			5
A			.			.			.			.			.
B			.			.			.			.			.
C			.			.			.			.			.
D			.			.			.			.			.
E			.			.			.			.			.

(b)

Figura 5. Interfaz del tablero que debes construir: **(a)** cómo se debe mostrar al usuario. **(b)** Cómo debes organizar la impresión por pantalla (cada casilla contiene un caracter). Cada celda vacía es un espacio en blanco.

```
Escoge una opción: (1) Generar tablero (2) Cargar tablero (3) Salir: 2
Ingresa el nombre del archivo: juego23.sal
   1  2  3  4  5  6  7  8  9  10
A  .  .  .  .  .  .  .  .  .  .
B  .  .  .  .  .  .  .  .  .  .
C  .  .  .  .  .  .  .  .  .  .
D  .  .  .  .  .  .  .  .  .  .
E  .  .  .  .  .  .  .  .  .  .
F  .  .  .  .  .  .  .  .  .  .
G  .  .  .  .  .  .  .  .  .  .
H  .  .  .  .  .  .  .  .  .  .
I  .  .  .  .  .  .  .  .  .  .
J  .  .  .  .  .  .  .  .  .  .
Ingresa la casilla del tablero que quieres abrir: c2
   1  2  3  4  5  6  7  8  9  10
A  .  .  .  .  .  .  .  .  .  .
B  .  .  .  .  .  .  .  .  .  .
C  .  1  .  .  .  .  .  .  .  .
D  .  .  .  .  .  .  .  .  .  .
E  .  .  .  .  .  .  .  .  .  .
F  .  .  .  .  .  .  .  .  .  .
G  .  .  .  .  .  .  .  .  .  .
H  .  .  .  .  .  .  .  .  .  .
I  .  .  .  .  .  .  .  .  .  .
J  .  .  .  .  .  .  .  .  .  .
Ingresa la casilla del tablero que quieres abrir:
```

Figura 6. Interfaz del juego ***BuscaMinas*** que debes construir para la opción 2 “Cargar tablero”.

IV. Consideraciones en la programación.

En la solución que implementes debes considerar que:

1. El archivo de entrada y el de salida deberán estar en la **misma ubicación** donde está tu código fuente.
2. El archivo de entrada para generar un tablero **estará bien escrito**, según el formato explicado en el punto III.1.
3. El **mayor tamaño** que tendrá el tablero será de 26×26.
4. Tu programa debe soportar (no *caerse*) el ingreso de **mayúsculas o minúsculas** para referenciar las filas de las celdas, como también el ingreso de números enteros fuera de rango.
5. Si la referencia a la casilla que ingresa el usuario es incorrecta (porque está fuera de rango o porque ya abrió esa casilla), tu programa debe **volver a mostrar el tablero y solicitar la casilla** para abrir (nada más!)
6. Los únicos caracteres que el tablero de juego debe mostrar son: “`.`”, “`*`”, y los números entre el `0` y el `8`.
7. Tú programa **solo puede contener**:
   7.1. Importación de módulos nativos de Python. No puedes utilizar módulos que impliquen una instalación en el computador.
   7.2. Definición de constantes.
   7.3. Funciones definidas por ti.
8. Las funciones que definas **deben tener una responsabilidad clara** en el contexto de la solución. **Se aplicarán descuentos en la nota si no se cumple con este punto.**

9. Algunas funciones/métodos a considerar para el manejo de **archivos** son:

   9.1. `open(nombreArchivo, modo)`: Permite abrir un archivo de texto llamado `nombreArchivo` (un *string*) y el `modo` corresponde a: "r" (lectura), "w" (escritura), "a" (escribe al final) y "r+" (lectura y escritura). Para el caso de "w", si el archivo no existe, se crea. Esta función devuelve un objeto del tipo *file* ("archivo").

   9.2. `file.write(string)`: permite escribir el `string` dentro del archivo que se ha abierto y se está referenciando.

   9.3. `file.read(1)`: permite leer un caracter dentro del archivo que se ha abierto y se está referenciando.

   9.4. `file.readline()`: permite leer una línea dentro del archivo que se ha abierto y se está referenciando.

   9.5. `file.readlines()`: permite leer el archivo completo que se ha abierto y se está referenciando.

   9.6. `file.readlines()`: permite leer el archivo completo que se ha abierto y se está referenciando.

   9.7. `file.close()`: permite cerrar el archivo.

V. Sobre la entrega, atrasos y faltas a la ética.

1. El profesor del curso indicará los integrantes de cada grupo.
2. No se aceptarán tareas atrasadas.
3. Debes subir su trabajo a la plataforma BlackBoard, en el aula virtual de tu curso, en el lugar que tu profesor informe para aquello.
4. El trabajo debe ser subido por **SOLO UNO de los integrantes del grupo**.
5. El nombre de su archivo debe ser: los RUTs de los integrantes del grupo (sin puntos, ni guiones, ni los dígitos verificadores), separados por un guion (no importa en que orden los escribas).

   Ejemplo:

   Si los integrantes tienen los RUTs **12.345.678-9** y **98.654.321-0**, el archivo se debe llamar: **12345678-987654321.py**

6. **Si tu programa tiene una interfaz diferente en algún detalle a la solicitada, se aplicarán descuentos en la nota.**
7. **Ante un alto porcentaje de similitud entre 2 o más tareas, los trabajos involucrados serán evaluado con nota 1.0**.
8. Las consultas las debe realizar directamente a los profesores, de lunes a viernes en los horarios y formas que ellos establezcan.