

## **TRABAJO GRUPAL N°1**

### **Evaluación Unidad 1: Fundamentos de Programación en Python**

**Fecha de entrega:** Domingo 31 de mayo de 2020, 23:59 hrs.

**Modalidad:** Trabajo en grupos de **dos** personas.

**Horario de Consultas:** Coordinar reunión para consultas por correo o **wsp (+56951987988)**.

#### **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. Resolver un problema, considerando que se cuenta con el procedimiento en lenguaje natural, que se debe implementar para resolverlo.
3. Seguir **en forma precisa la interfaz solicitada**.

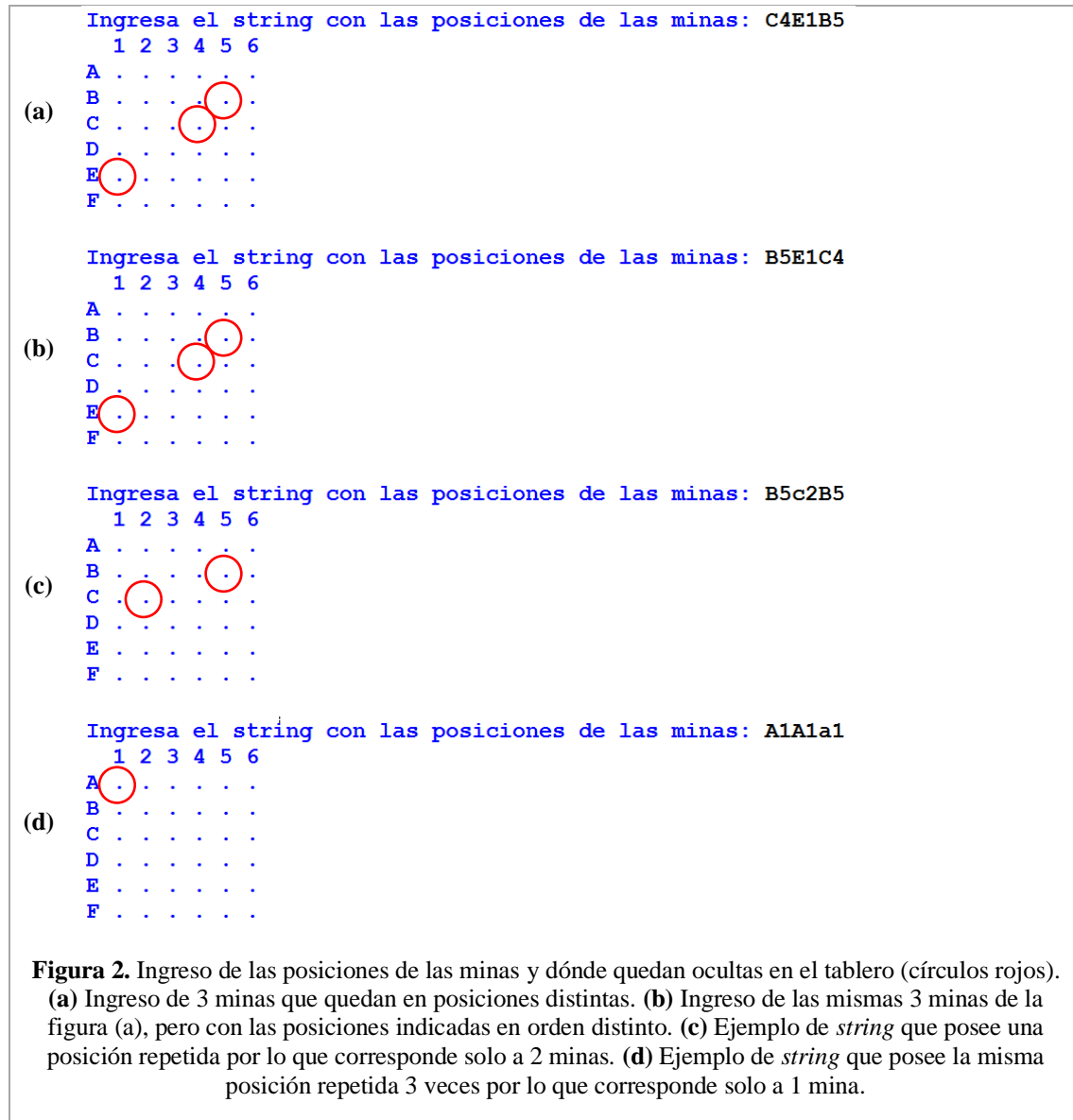
#### **II. Enunciado.**

Debes crear un programa en Python que le permita al usuario jugar al juego *BuscaMinas*, considerando que posee 6x6 casillas y 3 minas. En la **figura 1(a)** se muestra la interfaz que debes lograr en el programa que construyas. Observa que las filas se referencian con las letras mayúsculas entre la **A** y la **F**, y las columnas con números entre el **1** y el **6**. Para simular casillas “cerradas” debes usar un punto “.”. Para referenciar una casilla en particular debes escribir primero la **fila (letra)** y luego la **columna (número)**. En el ejemplo la casilla que está destacada en el círculo rojo es la **E2**. En la **figura 1(b)**, puedes observar el detalle de la impresión por pantalla. Considera que en cada casilla cabe un carácter. Las casillas que se muestran vacías poseen un espacio en blanco.



### Las reglas del juego son:

1. Al iniciar tu programa debe solicitar las posiciones donde deben ocultarse las 3 minas. Estas posiciones se deben ingresar en un solo *string*. En él se deben indicar las posiciones de las 3 minas (escribiendo primero la **fila** – **letra** - y luego la **columna** - **número**). Debes tener cuidado con las posiciones que podrían estar repetidas: en este caso podrían haber menos de 3 minas (ver ejemplos en **figura 2**).



2. Luego, tu programa debe pedir al usuario la casilla que desea abrir. La posición de la casilla debe indicar primero la **fila** (**letra**) y luego la **columna** (**número**). En este punto, se pueden dar 3 posibles escenarios:

2.1. La casilla no tiene una mina: En este caso tu programa deberá **contar en las casillas en torno** a la escogida, el número de minas que hay. De esta manera, el programa le debe mostrar el tablero con ese número en la posición que escogió (ver **figura 3**). Además, en las siguientes jugadas, el programa debe mostrar el tablero con el valor calculado para todas las casillas ya abiertas (ver **figura 4**).

```

Ingresa el string con las posiciones de las minas: B5C4B3
 1 2 3 4 5 6
A . . . . .
B . . . . .
C . . . . .
D . . . . .
E . . . . .
F . . . . .
Ingresa la casilla del tablero que quieres abrir: A2
 1 2 3 4 5 6
A . 1 . . . .
B . . . . .
C . . . . .
D . . . . .
E . . . . .
F . . . . .
Ingresa la casilla del tablero que quieres abrir:

```

**Figura 3.** Ejemplo de apertura de una casilla: el usuario ingresa la ubicación **A2**, y revisando en torno a esa celda se cuenta 1 sola mina, por lo que se muestra un 1 en esa posición (cuadrado amarillo).

```

Ingresa el string con las posiciones de las minas: B5C4B3
 1 2 3 4 5 6
A . . . . .
B . . . . .
C . . . . .
D . . . . .
E . . . . .
F . . . . .
Ingresa la casilla del tablero que quieres abrir: A2
 1 2 3 4 5 6
A . 1 . . . .
B . . . . .
C . . . . .
D . . . . .
E . . . . .
F . . . . .
Ingresa la casilla del tablero que quieres abrir: B4
 1 2 3 4 5 6
A . 1 . . . .
B . . 3 . . .
C . . . . .
D . . . . .
E . . . . .
F . . . . .
Ingresa la casilla del tablero que quieres abrir:

```

**Figura 4.** Ejemplo de una segunda apertura de casilla: el usuario ingresa esta vez la ubicación **B4**, e imprime un 3 en esa posición por la cantidad de minas que se encuentran alrededor (cuadrado amarillo). Observa que el 1 de la jugada anterior se sigue mostrando.

- 2.2. La casilla no tiene una mina, y es la última que falta abrir: en este caso tu programa debe mostrar al usuario el tablero con un asterisco “\*” en las posiciones donde estaban las minas con los valores de las minas entorno de cada casilla. Abajo del tablero tu programa debe mostrar el mensaje “**GANASTE**” (ver **figura 5**).

```
...
  1 2 3 4 5 6
A . 2 1 0 0 0
B 2 . 2 1 0 0
C 1 2 . 1 0 0
D 0 1 1 . 0 0
E 0 0 0 0 0 0
F 0 0 0 0 0 0
Ingresa la casilla del tablero que quieres abrir: d4
  1 2 3 4 5 6
A * 2 1 0 0 0
B 2 * 2 1 0 0
C 1 2 * 1 0 0
D 0 1 1 1 0 0
E 0 0 0 0 0 0
F 0 0 0 0 0 0
GANASTE
>>> |
```

**Figura 5.** Ejemplo del final de un juego en el que al usuario sólo le falta abrir la casilla **D4**. Al ingresar esa posición el programa muestra el tablero, y esta vez muestra con asteriscos “\*” las posiciones donde estaban ocultas las minas. Luego de eso, imprime el mensaje “**GANASTE**”.

- 2.3. La casilla tiene una mina: En este caso tu programa debe mostrar al usuario el tablero con un asterisco “\*” en las posiciones donde estaban las minas, los valores de las casillas que se alcanzaron a abrir, y un punto para las casillas que no se abrieron. Abajo del tablero tu programa debe mostrar el mensaje “**PERDISTE**” (ver **figura 6**).

```
...
  1 2 3 4 5 6
A . 2 . 0 . .
B 2 . 2 . . .
C . . . . .
D . . . 1 . .
E . . . . .
F . . . . . 0
Ingresa la casilla del tablero que quieres abrir: C3
  1 2 3 4 5 6
A * 2 . 0 . .
B 2 * 2 . . .
C . . * . . .
D . . . 1 . .
E . . . . .
F . . . . . 0
PERDISTE
>>> |
```

**Figura 6.** Ejemplo del final de un juego en el que al usuario abre la casilla **C3** donde se encuentra una mina. Al ingresar esa posición el programa muestra el tablero, y esta vez muestra con asteriscos “\*” las posiciones donde estaban ocultas las minas. Luego de eso, imprime el mensaje “**PERDISTE**”.

### III. Consideraciones en la programación.

En la solución que implementes debes considerar que:

1. Tu programa debe soportar el ingreso de mayúsculas o minúsculas para referenciar las filas de las celdas (ver **figuras 2(c), 2(d) y 5**).
2. 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!)
3. Los únicos caracteres que el tablero de juego debe mostrar son: “.”, “\*”, y los números entre el **0** y el **3**.
4. Algunas operaciones a considerar para el manejo de **listas** y de **strings** son:

#### 4.1. Para las **listas**:

- 4.1.1. Las posiciones de los elementos dentro de una lista se enumeran desde 0.
- 4.1.2. En particular, para acceder a una posición de tu tablero deberás colocar entre 2 pares de corchetes la referencia a una casilla (luego del nombre de la variable): [fila][columna]  
  
Tanto “fila” como “columna” deben ser valores entre 0 y 5. El uso de 2 pares de corchetes se debe a que el tablero deberá ser una **lista de listas**.
- 4.1.3. **variableLista.append(elem)**: Permite agregar elementos **elem** a la lista **variableLista**. En particular es útil para agregar las sublistas que llevará el tablero.

#### 4.2. Para los **strings**:

- 4.2.1. Las posiciones de los caracteres de un *string* se enumeran desde 0.
- 4.2.2. Para acceder a un caracter en particular, debes colocar entre corchetes un entero (luego del nombre de la variable).
- 4.2.3. **variableString.upper()**: Genera un nuevo *string* donde aparecen en mayúsculas las letras del *string* almacenado en **variableString**.
- 4.2.4. **variableString.lower()**: Genera un nuevo *string* donde aparecen en minúsculas las letras del *string* almacenado en **variableString**.
- 4.2.5. **len(variableString)**: Indica la cantidad de caracteres que posee el *string* almacenado en **variableString**.

#### IV. 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. Debe subir su trabajo a la plataforma: <https://edx.icfunab.cl/>, en el lugar que se informe para aquello.
4. Su 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, será calificado con nota **1.0**.
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 sábado en los horarios y formas que ellos establezcan.