TP

TP 1
Introducción 1
Primera parte 2
pedir_dato 2
pedir_datos_tablero 2
generar_tablero 2
escribir_juego 3
Segunda parte **Error! Bookmark not defined.**
Armado de tableros **Error! Bookmark not defined.**
Sopa de letras **Error! Bookmark not defined.**

Introducción

El trabajo práctico consiste de dos partes. La primera parte se entrega la segunda clase de la semana 5 y la segunda parte se entrega la última semana (a convenir con cada docente). El docente hará correcciones a la parte 1 para que sean aplicadas para la entrega final.

Q C C O V A R D E C R A T O B H Í V
C H I B O V Z I N R E V Á V R W T I
V U L T O O G C O N V O I B O R G O
R V M A R A B I L L A R R B A G A Z
E A R A V A R G Q N G Ü E S U S A V
B S V K O Z N A V A R A G V Ú C T L
A Q C O O L L E M R E V V C E G A O
Z U V A G O L E B O N N A B A L X L
R E O O R M O B I L E J A H A B A Q
E I B B T A O O R V I C A A O S V R
V R B A O N V T V F O R Ñ E B I E P
O O O E G R E E L O V C A V R D L A
G B C B T O T B L E O O A T X M I Q
S V A L A V A N E L V O I L O V N J
E O L R E B E L A R O S B T L V A S
V A S T O R E T A V A L E A R T Y W
A D O V N G R A V A T A M J R E Ó Í

Ejemplo de una sopa de letras clásica

El TP está conformado por dos programas:

- Armado de tableros: El objetivo de este programa es armar los tableros de sopas de letras para que luego puedan ser jugados.

- Jugar a la sopa de letras: El objetivo de este programa es el de permitir al usuario jugar a la sopa de letras

Se espera que las dos partes del TP utilicen todos los conceptos vistos en las clases, como por ejemplo **las buenas prácticas**.

Primera parte

La primera parte consta de varias funciones aisladas, cada una deberá ser entregada en un archivo independiente. Las funciones a implementar son las siguientes

pedir_dato

Tiene como objetivo pedir un dato al usuario, validarlo y retornarlo
Esta función debería recibir como parámetro un texto y una **función** de validación (que retorna *True* si el dato es válido y *False* en caso contrario) y retorna el dato ingresado por el usuario. Debe pedir al usuario que ingrese el dato mostrando el texto, en caso de que el dato ingresado no pase la validación, deberá pedir el dato nuevamente.

Ejemplo:

```
def mayor_a_10(numero):
 return int(numero) > 10

pedir_dato("Ingrese un número mayor a 10: ", mayor_a_10)

"Ingrese un número mayor a 10: " 2
"ERROR. Ingrese un número mayor a 10:" 23
```

pedir_datos_tablero

Tiene como objetivo pedir los datos al usuario para generar un tablero. No recibe parámetros y retorna en un *tupla* los datos ingresados por el usuario (validados). Los datos a pedir son:

- N: Número entero. Va a representar la cantidad de columnas y filas de la sopa de letras. Tiene que ser mayor o igual a 15.
- Lista de palabras: Se ingresan de a una. La cantidad de palabras tiene que ser menor a N / 3. La longitud de la palabra tiene que ser menor a N / 3. Se termina el ingreso de palabras cuando se haya llegado al límite de palabras o se ingrese la palabra “fin”.
- nombre de archivo: Texto. Va a representar el nombre del archivo donde se guardará el tablero. El nombre del archivo no puede ser mayor a 30 caracteres.

generar_tablero

Tiene como objetivo el de generar el tablero de una sola de letras. Recibe un número N y una lista de palabras. Retorna una matriz (lista de listas) de dimensiones NxN donde en cada posición hay una letra. La matriz representa el tablero de sopa de letras y en cada posición tiene que haber una letra. El tablero devuelto tiene que contener las palabras entre

todas las letras de forma horizontal o de forma vertical. Los casilleros que no contengan una letra de las palabras a ubicar, deben tener una letra generada de forma aleatoria. Todas las letras tienen que estar en minúscula

Consejo: Intentar de insertar las palabras de a una en lugares aleatorios. En el caso de no ser posible re intentarlo hasta que se logre.

Ejemplo:

```
N = 4
palabras = ['pez', 'casa']
tablero = generar_tablero(N, palabras)
tablero: [
 ['c', 'p', 'e', 'z'],
 ['a', 'x', 'p', 'l'],
 ['s', 'c', 'q', 'u'],
 ['a', 'b', 'e', 'x'],
]
```

Visualización del tablero devuelto:

```
c | p | e | z
a | x | p | l
s | c | q | u
a | b | e | x
```

escribir_juego

El objetivo de esta función es la de escribir en archivos el juego (el tablero y la solución). Esta función recibe una matriz de NxN, un diccionario de palabras con sus ubicaciones y el nombre del archivo. Tiene que escribir dos archivos:

- nombre_del_archivo.csv: Escribe el tablero (la matriz) en un formato que luego pueda leerse y convertirse nuevamente en matriz de forma sencilla. Se recomienda guardarlo como CSV (sin columnas), donde cada línea es una fila de la matriz y las letras están separadas con coma. **Recomendación**: Revisar la liberia *csv,* en especial el método *writerow*.
- nombre_del_archivo_solucion.csv: Escribe las palabras a partir del diccionario de solución. Se espera que el diccionario de palabras tenga la siguiente forma:

```
{
 "casa": {
   "x_inicial": 0,
   "y_inicial": 0,
   "x_final": 3,
   "y_final": 0
 }
 ...
}
```

Se entiende que la fila 0 de la matriz es la primera contando desde arriba y la columna 0 es la primera contando de izquierda a derecha. El formato del archivo debe ser un CSV con las siguientes columnas: *palabra, x_inicial, y_inicial, y_final, x_final.*