debe ser un CSV con las siguientes columnas: *palabra, x_inicial, y_inicial, y_inicial, x_final.*

Segunda parte

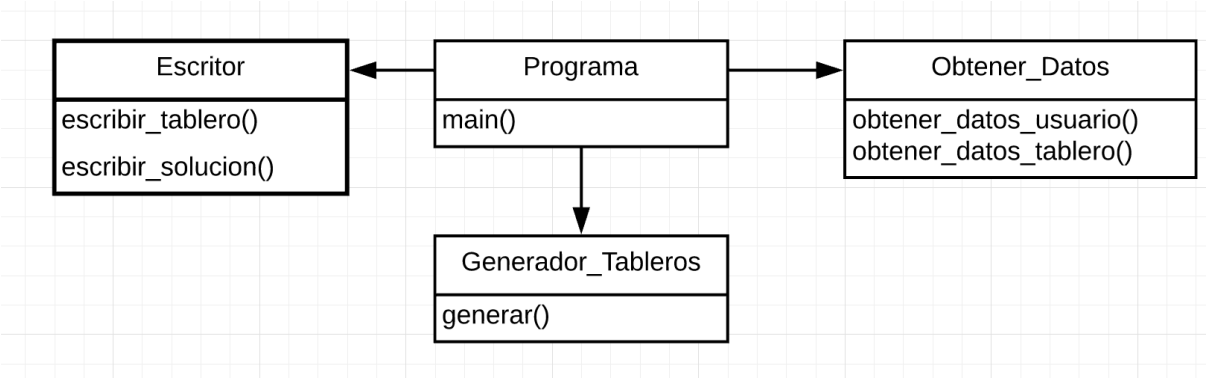
En la segunda parte del TP se espera que se integren todas las funciones implementadas en la parte 1 (corrigiendo las correcciones pertinentes) resultando en los dos programas completos. Se espera que para ambos programas utilicen todos los conceptos vistos en las clases, como por ejemplo las **la POO (objetos).** Se espera que las funciones implementadas en la primera parte sean utilizadas como métodos en las clases correspondientes.

Armado de tableros

Este programa debe integrar la mayoría de las funciones implementadas en la primera parte y generar una experiencia completa para que el usuario pueda crear tableros de sopa letras.

El usuario deberá ingresar un número N (cantidad de filas y columnas del tablero), una lista de palabras y el nombre del tablero y el resultado final deberá resultar en dos archivos: el tablero y la solución.

Se recomiendan utilizar las siguientes clases:



Sopa de letras

Este es el programa principal. Permitirá al usuario jugar a los tableros previamente generados. Tiene que tener el siguiente funcionamiento:

- Pedir al usuario su nombre y el tablero a utilizar. Validar:
    - Longitud del nombre no puede ser mayor a 40 caracteres
    - El archivo tiene que existir.

  En caso de que alguna validación falle, tiene que pedir nuevamente el dato.
- Mostrar al usuario el tablero con el formato mostrado anteriormente

```
c | p | e | z
a | x | p | l
```

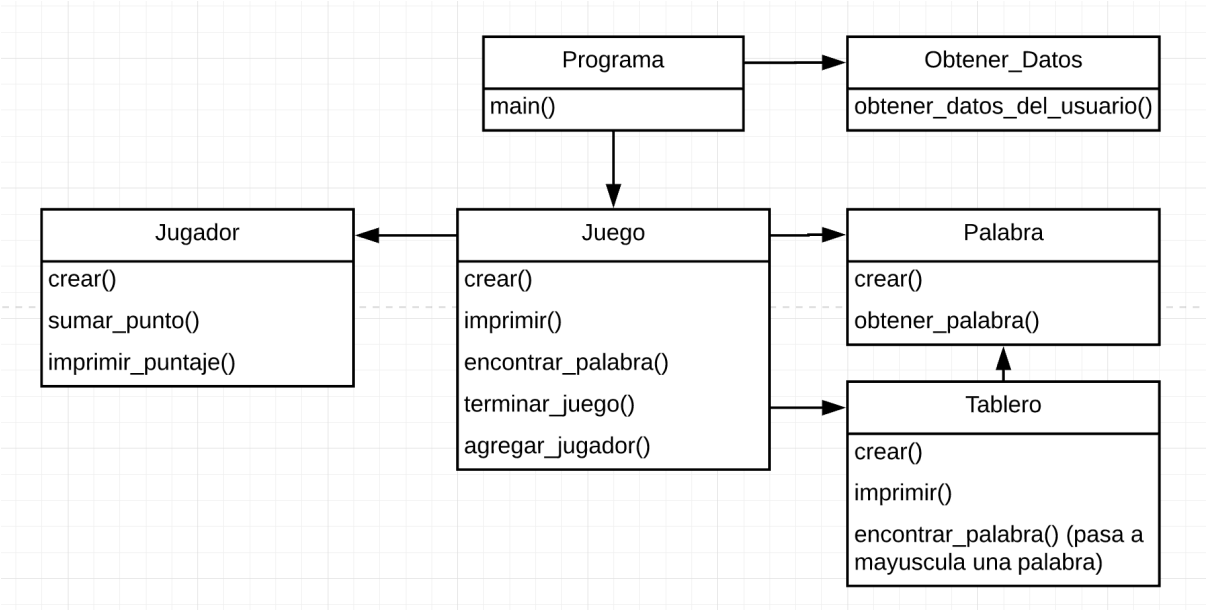
```
s | c | q | u
a | b | e | x
```

- Pedir al usuario que ingrese una palabra que encuentre. Si la palabra es parte de la sopa de letras, deberá re imprimir el tablero pero con la palabra encontrada con letras mayúsculas. Por ejemplo si hasta el momento las palabras encontradas son: “casa”, debería imprimir algo asi:

```
C | p | e | z
A | x | p | l
S | c | q | u
A | b | e | x
```

- En caso de que la palabra no esté presente en el tablero, deberá imprimir un mensaje indicandolo. El chequeo para ver si la palabra está presente o no, deberá tener la complejidad computacional de **O(1).**
- Al encontrar todas las palabras o cuando el usuario escribe la palabra “fin” deberá hacer las siguientes acciones:
    - Mostrar el nombre del usuario junto a su puntaje (cantidad de palabras encontradas).
    - Mostrar las palabras encontradas en el orden que fueron encontradas. Por ejemplo: “Palabras encontradas: Casa, Pez”. Recomendación: Utilizar alguna de las estructuras vistas en las últimas clases.
    - Mostras las palabras no encontradas por orden alfabético.

Se recomiendan utilizar las siguientes clases:



IMPORTANTE: Las recomendaciones de clases para ambas partes no están necesariamente completas, se espera que el estudiante agregue/modifique lo que sea necesario para que el diseño resultante sea eficiente.

Criterios de aprobación

El trabajo debe cumplir con los siguientes requerimientos para ser aprobado:

- Toda la funcionalidad debe funcionar como se espera, incluyendo casos borde.
- El código debe ser subido al sistema antes de la fecha límite.
- Las correcciones de la primera parte deben verse aplicadas en la entrega final.
- El código debe cumplir un estándar de calidad aceptable siguiendo los lineamientos establecidos en las clases.
- Cualquier otro criterio establecido durante la clase puede ser evaluado para la aprobación (o no) de este trabajo práctico.