Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Estructuras de Datos
Proyecto #1
Ingenieros

- Ing. Luis Espino
- Ing. Jesús Guzmán
- Ing. Álvaro Hernández

Auxiliares

- Kevin Mejía
- Ricardo Cutz
- Dennis Masaya

SCRABBLE++

Objetivos

1. Familiarizarse con el lenguaje de programación **C++**.
2. Familiarizarse con la lectura y escritura de archivos de **Texto**.
3. Comprender y desarrollar distintas estructuras de datos lineales como lo son listas, pilas, colas
4. Definir algoritmos de búsqueda y recorrido.
5. Familiarizarse con la herramienta **Graphviz** para la generación de reportes de estructuras gráficos.

Definición del problema

Debido a la popularidad reciente de juegos de mesa en cafés y restaurantes de la ciudad de Guatemala, y debido al riesgo que se corre al utilizar plataformas físicas en cuanto a daños y desgaste a la propiedad del restaurante se busca una nueva forma de disfrutar dichos juegos de una manera digital.

Descripción

Según los requerimientos antes mencionados se deberá de desarrollar el juego denominado como Scrabble, la interfaz de interacciones del juego será mediante la consola de comandos, mientras que la visualización del tablero y datos del juego será por medio de una pagina HTML sencilla.

Implementacion de la Solución

La aplicación contará con los siguientes módulos:

- Lectura de Archivo: Este archivo permitirá definir las configuraciones del juego.
- Juego: Este módulo permite la interacción con el juego
- Reportes: Este módulo permite visualizar los reportes de las estructuras utilizadas para el juego.

Lectura de Archivo

Este módulo de la aplicación permitirá leer un archivo en formato JSON que define los siguientes aspectos del juego:

- Las dimensiones del tablero de juego, el cual será cuadrado
- Las casillas que cuentan con doble o triple puntuación
- El diccionario de palabras válidas del juego

Los atributos que contiene el archivo de entrada son los siguientes:

- dimensión: es la definición de ancho y longitud del tablero de juego
- casillas: es un atributo que contiene dos atributos:
    - dobles: es un arreglo de casillas en formato “x” y “y” para definir las casillas que contarán con doble puntaje
    - triples: es un arreglo de casillas en formato “x” y “y” para definir las casillas que contarán con triple puntaje
- diccionario: es un arreglo de palabras que se ingresarán al programa para poder validar las palabras que se coloquen en el tablero. Las palabras del diccionario deben ingresarse en una **lista doblemente enlazada circular.**

Ejemplo de archivo:

```
{
  "dimension":20,
  "casillas":{
    "dobles":[
      {
        "x":2,
        "y":5
      },
      {
        "x":4,
        "y":10
      },
      {
        "x":10,
        "y":15
      }
    ],
    "triples":[
      {
        "x":0,
        "y":1
      },
      {
        "x":6,
        "y":10
      },
      {
        "x":0,
        "y":15
      }
    ]
```

```
  },
  "diccionario":[
    {
      "palabra": "hola"
    },
    {
      "palabra": "mundo"
    },
    {
      "palabra": "proyecto"
    },
    {
      "palabra": "diccionario"
    }
  ]
}
```

Nota: Los archivos de entrada deben de venir perfectamente definidos, es decir que no se colocará errores de forma intencional.

Juego

Las reglas del juego de scrabble se basan en el juego de mesa común en donde el jugador intenta ganar más puntos mediante la construcción de palabras sobre el tablero de cuántas casillas se indique en el archivo de entrada. Las palabras pueden formarse horizontalmente o verticalmente siempre y cuando aparezcan en el diccionario ingresado en el archivo de entrada.

Fichas del Juego

El juego cuenta con diferentes fichas representando letras del alfabeto para formar palabras. Cada una de estas fichas tienen una puntuación diferente. A continuación se presenta una tabla con la puntuación que tendrá cada una de estas fichas:

Puntaje	Letra	Cantidad de Fichas
1 Punto	A	12
	E	12
	O	9
	I	6
	S	6
	N	5

	L	4
	R	5
	U	5
	T	4
2 Puntos	D	5
	G	2
3 Puntos	C	4
	B	2
	M	2
	P	2
4 Puntos	H	2
	F	1
	V	1
	Y	1
5 Puntos	Q	1
8 Puntos	J	1
	Ñ	1
	X	1
10 Puntos	Z	1

Las fichas serán ingresadas a una **cola** de manera al azar.

Jugadores

Dentro del menú del juego debe de existir una opción para poder agregar nuevos jugadores, los cuáles únicamente contarán con un nombre de usuario **único**, es decir que no debe de repetirse. En caso de ingresar un nombre de usuario ya existente, se debe de notificar que no es válido el ingreso. Los usuarios deben de almacenarse en un **árbol binario de búsqueda.**

Score Board

Cada jugador va a tener un historial de puntajes que será almacenado utilizando una **lista simplemente enlazada ordenada**, es decir que al momento de terminar una partida, su puntaje final será agregado a esta lista (puntajes ordenados del mayor al menor) de puntajes. El menú del juego tendrá una opción para poder mostrar los mejores puntajes, es decir que de cada uno de los jugadores que existen registrados se debe de buscar su mejor puntaje y luego crear una **lista simplemente enlazada** (otra lista) que muestre los mejores puntajes ordenados de mayor puntaje al menor. Este scoreboard debe mostrar el nombre del jugador y su puntaje.

Tablero de Juego

Ya que la aplicación es desarrollada en consola, se debe de tener una forma gŕafica de visualizar el tablero, por lo que se requiere que la aplicación pueda genere el reporte de la estructura utilizando la herramienta de **graphviz**. Este reporte debe de actualizarse justo después de ejecutar un turno para ver el progreso del juego. El tablero de juego debe de manejarse utilizando una **matriz dispersa,** con la idea de que solamente las casillas que están siendo utilizadas sean las que existan en memoria. .

Reglas del Juego

- Al inicio se deben de escoger los jugadores que participarán en la partida, estos deben de escogerse de la lista circular doblemente enlazada donde están almacenados. El juego será únicamente para dos jugadores.
- A cada jugador se le darán siete fichas que se almacenarán en una **lista doblemente enlazada**, estas fichas serán tomadas de la cola de fichas disponibles del juego. Cada jugador tendrá su propia **lista doblemente enlazada** con las siete fichas que se dieron.
- Se inicia el juego tomando a uno de los dos jugadores al azar.
- Los turnos se irán rotando por cada jugador.
- Cada jugador solo puede formar una palabra por turno.
- Para terminar el turno el jugador debe de indicar que su turno ha terminado, y validar el turno. Validar, quiere decir que se va a verificar que las letras que se colocaron en el tablero forman palabras válidas, de no ser así las letras son removidas del tablero y regresan a la lista de fichas del jugador que realizó el turno. En caso de que el turno sea válido, se hace la correspondiente suma de puntos.
- Si el jugador no puede formar ninguna palabra, puede intercambiar algunas de sus fichas por otras, esto también acabará su turno y las fichas que tenía anteriormente regresarán a la cola de fichas.
- El objetivo del juego es crear palabras para obtener puntos. El valor de una palabra es igual a la suma de los valores individuales de las letras que la componen. Si se utiliza una casilla que tenga doble o triple puntuación, el multiplicador será aplicado al valor individual de la letra que esté en la casilla, no al valor de la palabra.
- Al momento de finalizar un turno se debe de validar que las letras ingresadas por el jugador formen una palabra válida leída de izquierda a derecha o de forma vertical de arriba hacia abajo.
- El juego debe de finalizar cuando se acaben las fichas de la cola o se escoja la opción de finalizar el juego (queda a discreción del estudiante donde colocar esta opción). Al finalizar el juego se deberá de mostrar el ganador y su puntaje total.

Estructuras a Utilizar en la Lógica del Juego

- El diccionario de palabras que se ingresarán en el archivo de entrada, debe estar almacenadas en una **LISTA DOBLEMENTE ENLAZADA CIRCULAR.**
- Las fichas disponibles en el juego deben estar almacenadas en una **COLA** estas deben de ingresar a la cola de forma aleatoria. De esta cola se tomarán las fichas para cada jugador en la partida.
- Los usuarios del juego que se registren deben de ser almacenados en un **ÁRBOL BINARIO DE BÚSQUEDA** tomando como criterio de ordenamiento el nombre de usuario. Recordar que no deben existir usuarios repetidos.

- Para el historial de puntaje de cada jugador se debe de utilizar una **LISTA SIMPLEMENTE ENLAZADA ORDENADA** del mayor puntaje al menor.
- Para el scoreboard se debe de utilizar una **LISTA SIMPLEMENTE ENLAZADA ORDENADA** del mayor puntaje al menor. Debe almacenar el jugador y su puntaje.
- Para el tablero de juego debe de utilizarse una **MATRIZ DISPERSA**, es decir que solamente las casillas que estén ocupadas por letras deben de mostrarse en los reportes de la estructura. Tomar en cuenta que las casillas a ingresar serán válidas mientras estén dentro de la dimensión definida por el archivo de entrada.
- Para las fichas que los jugadores tienen disponibles durante la partida debe de utilizarse una **LISTA DOBLEMENTE ENLAZADA** para cada uno de los jugadores.

NOTA: Todas las estructuras anteriormente mencionadas deben estar implementadas por el estudiante en el lenguaje C++.

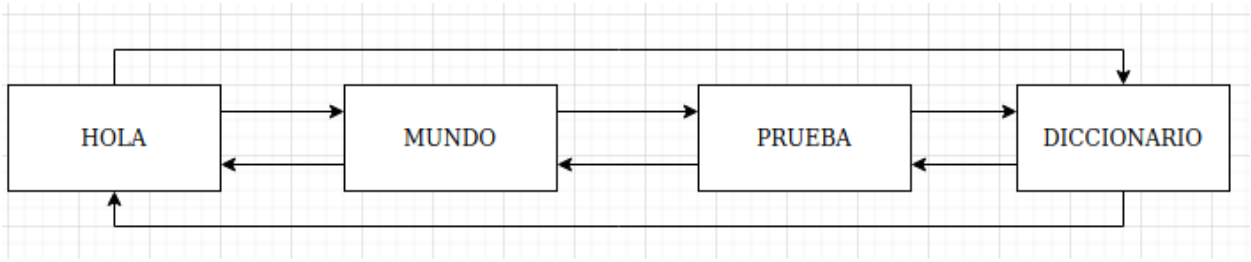
Reportes

El módulo de reportes debe de permitir ver los estados de las estructuras en cualquier momento durante la ejecución de la aplicación. Los reportes tendrán validez siempre y cuando estos sean elaborados utilizando la herramienta de Graphviz. Los reportes que deben de mostrarse son los siguientes:

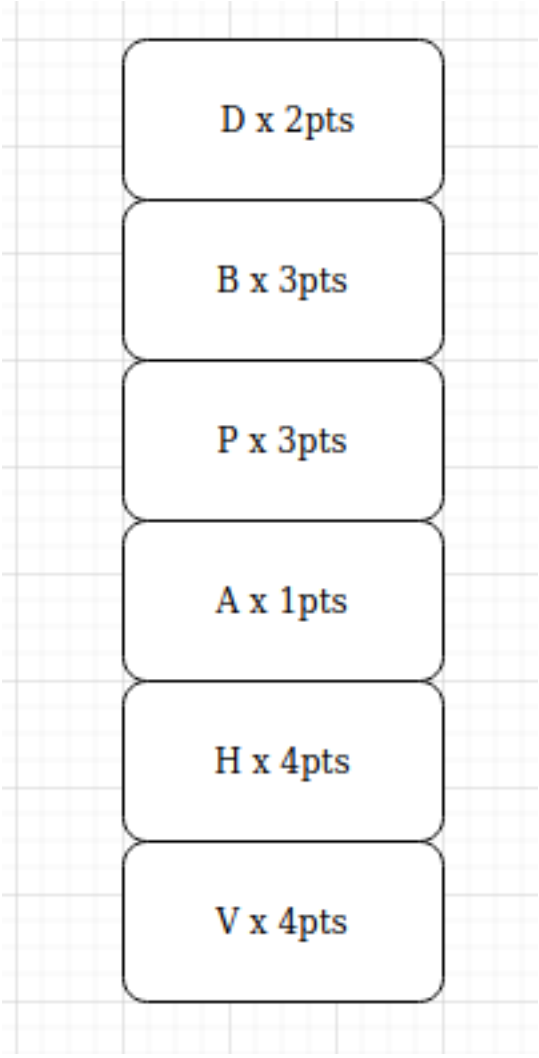
- La lista doblemente enlazada circular en donde se almacenan las palabras del diccionario
- La cola de las fichas que están disponibles en el juego, cada nodo de la lista debe de mostrar la letra que representa y el puntaje asociada a ella.
- El árbol binario de búsqueda donde se encuentran almacenados los usuarios del juego, debe de mostrarse en cada nodo el nombre del usuario.
- Recorrido preorden del árbol binario de búsqueda
- Recorrido inorden del árbol binario de búsqueda
- Recorrido postorden del árbol binario de búsqueda
- La lista simplemente enlazada ordenada que representa el historial de puntajes por jugador, este reporte debe de recibir como parámetro el usuario del jugador del cuál se desea ver su historial de puntajes.
- La lista simplemente enlazada ordenada que representa el scoreboard general del juego.
- La matriz dispersa que representa el estado del tablero en memoria.
- Las dos listas que contienen las fichas de los jugadores durante la partida.

Ejemplos de Reportes

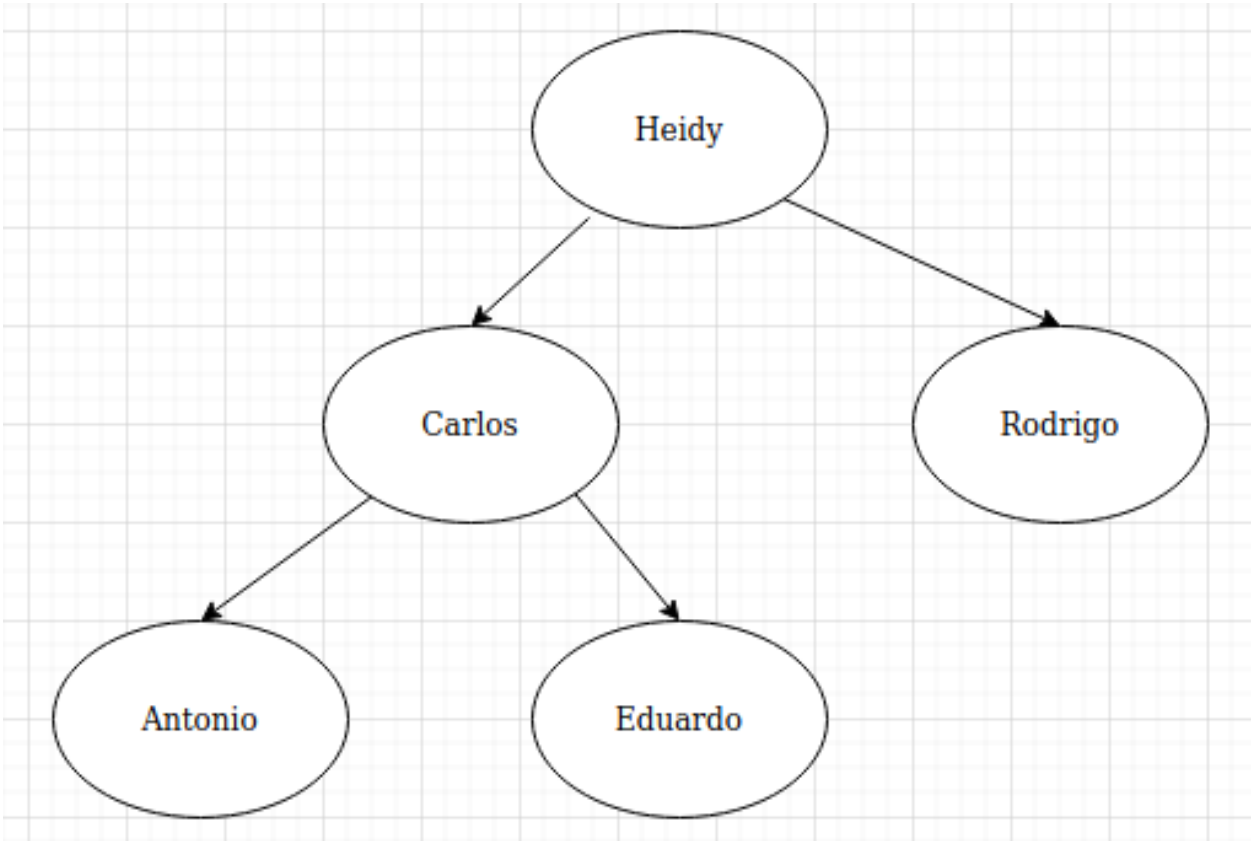
Lista Doblemente Enlazada Circular: Diccionario



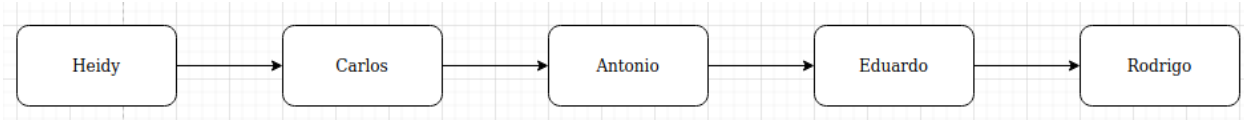
Cola: Fichas disponibles del Juego



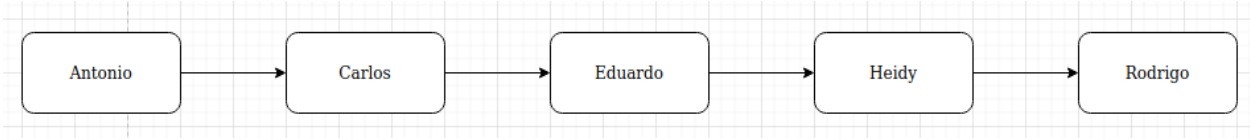
Jugadores: Árbol Binario de Búsqueda



Recorrido Preorden del ABB

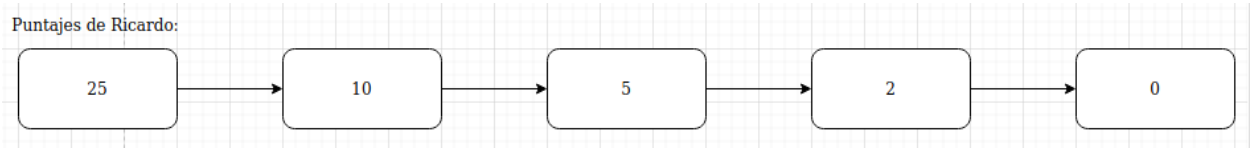


Recorrido Inorden del ABB

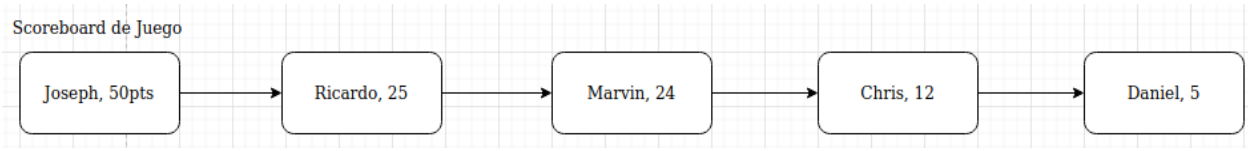


Recorrido Postorden del ABB

Historial de Puntaje Por Jugador: Lista Simple Ordenada

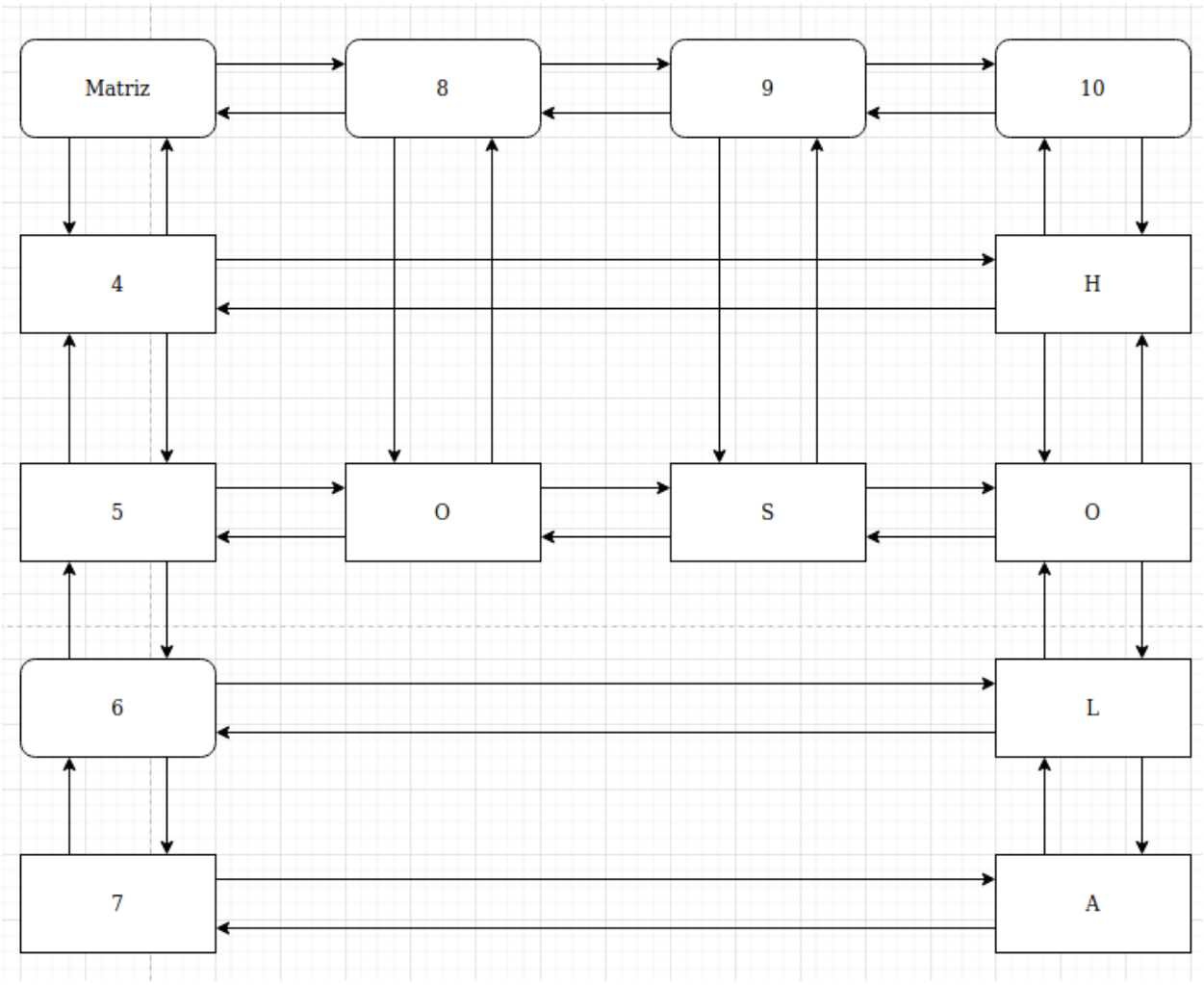


Scoreboard: Lista Simple Ordenada

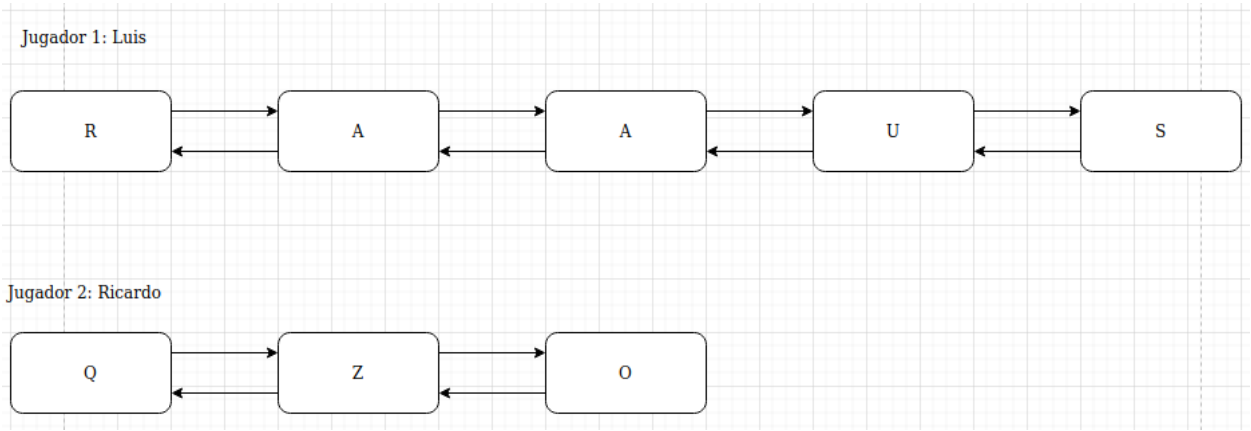


Tablero: Matriz Dispersa

El siguiente ejemplo muestra el estado del tablero con dos palabras: OSO y HOLA



Fichas Por Jugador: Listas Doblemente Enlazadas



Restricciones

Las estructuras deben de ser desarrolladas por los estudiantes sin el uso de ninguna librería o estructura predefinida en el lenguaje a utilizar.

Observaciones

1. Lenguaje a utilizar: **C++**
2. Herramienta para desarrollo de reportes gráficos: **Graphviz.**
3. La aplicación debe de ser capaz de generar y abrir con un visor de imágenes predeterminado las imágenes generadas con Graphviz.
4. La entrega se realizará por medio de: **Github**, cada estudiante deberá crear un repositorio con el nombre: **EDD_1S2020_PY1_#carnet**, y agregar a su auxiliar correspondiente como colaborador del mismo, para poder analizar su progreso y finalmente a partir del mismo repositorio realizar la calificación correspondiente.
   Dennis Masaya: https://github.com/Dennis201503413
   Ricardo Cutz: https://github.com/ricardcutzh
   Kevin Mejía: https://github.com/kevin140414
5. Además de tener a su auxiliar como colaborador del repositorio para tener un control y orden de las personas que entreguen deberán de colocar el Link de su repositorio en la Tarea que cada auxiliar asignara en su classroom correspondiente.
6. Fecha y hora de entrega: **Domingo 29 de Marzo, a las 23:59 horas**.
7. **Copias serán penalizadas con una nota de 0 y castigadas según lo indique el reglamento.**