Universidad San Carlos de Guatemala
Centro Universitario de Occidente
División Ciencias de la Ingeniería
Ingeniería en Ciencias y Sistemas
Laboratorio de Estructura de Datos
Primer Semestre 2025

PRIMERA PRACTICA

Objetivo General

- Elaborar algoritmos eficaces y eficientes para la resolución de problemas.

Objetivos Específicos

- Utilizar la notación BigO.
- Manejar de manera competente el uso de arreglos y listas enlazadas para la gestión dentro del sistema.
- Aplicar conceptos recibidos durante la clase y laboratorios.

Descripción

En base a su experiencia como programador se le solicita desarrollar un programa que simulará una versión simplificada del juego de Scrabble, donde los jugadores formarán palabras con fichas de letras y obtendrán puntos según el valor de la palabra que formaron.

El programa deberá ejecutarse en consola y deber contar con las siguientes características:

1. **Tablero**
    - El tablero deberá de generarse con dimensiones de 15 x 15.
    - Diez casillas del tablero no permitirán a los jugadores colocar letras en ellas. (Estas casillas deberán posicionarse aleatoriamente cuando se inicia el juego)
2. **Gestión de jugadores y turnos**
    - Se permitirá la participación de al menos 2 jugadores.
    - Los turnos serán manejados por una cola en la que los jugadores se rotarán. (El orden de los jugadores se creará de forma aleatoria)
3. **Manejo de fichas**
    - Cada jugador tendrá un conjunto de fichas representadas por una lista enlazada.
    - Las fichas deberán ser ordenadas en base a su valor de puntuación.
4. **Formación de palabras**
    - Los jugadores formarán palabras utilizando sus fichas.
    - Una pila almacenará las palabras jugadas para permitir la funcionalidad de "deshacer" el último movimiento.
5. **Sistema de puntuación**
    - Cada letra tendrá un valor asociado.
    - Se calculará la puntuación en base a la palabra formada y se registrará en la lista de puntuaciones.
    - Se deberá de implementar un algoritmo para ordenar esto con el objetivo de visualizar la lista de jugadores según su puntuación.

6. **Finalización del juego**
    - El juego terminará cuando se acaben las fichas o cuando los jugadores no puedan formar más palabras.
    - Se mostrarán los resultados ordenados por puntaje o alfabéticamente.
7. **Carga de datos (Palabras para el juego)**
    - Al iniciar una ronda se pedirá que se cargue un archivo CSV que contendrá las palabras para utilizar durante el juego.
    - Las palabras deberán de almacenarse en una estructura de datos vista en clase y se deberán ordenar alfabéticamente.
    - En base a las palabras cargadas, se deberán obtener el total de letras y se les asignará un valor aleatorio, estas se deberán de repartir aleatoriamente entre el número de jugadores para esa ronda.

Servicios Críticos

Scrabble			
No.	**Servicio Crítico**	**Complejidad**	**Descripción**
1	Ingreso de jugadores y fichas	O(1)	Se añaden los jugadores y sus fichas a la lista enlazada al inicio del juego.
2	Gestión de turnos con cola	O(1)	Se utiliza una cola para administrar los turnos de los jugadores, facilitando su rotación eficiente.

3	Inserción y eliminación de fichas en lista enlazada	O(n)	Se gestionan las fichas disponibles del jugador mediante una lista enlazada para agregar y quitar letras.
4	Ordenación de fichas por puntuación	O(n log n)	Se utiliza un algoritmo de ordenación eficiente para mostrar las fichas en orden de mayor a menor puntuación.
5	Registro de palabras jugadas con pila	O(1)	Se almacenan las palabras jugadas en una pila, lo que permite deshacer la última palabra fácilmente. (Se revierte la colocación de la última ficha)
6	Cálculo y ordenación de puntuaciones	O(n^2)	Se calcula la puntuación de cada jugador y se ordena la lista de puntuaciones al final del juego.
7	Ordenación de palabras iniciales	O(n^2)	Antes de iniciar la partida deberá el programa ordenar estas palabras alfabéticamente.

Reportes

El programa debe mostrar reportes en base a la última ronda jugada y deberá mostrar lo siguiente:

- Historial de palabras jugadas (que es extraído de la pila de palabras).
- Historial de palabras no encontradas.
- Lista de jugadores ordenada por puntaje y por nombre.
- Resumen del tiempo promedio de cada turno.
- Cantidad de movimientos realizados por cada jugador.

Consideraciones

- Documentación técnica
    - Debe de ir calculada la complejidad de cada uno de los servicios críticos, así como otros métodos que implemente.
    - Debe argumentar la implementación del método o métodos de ordenamiento que utilice.
- Solo se pueden utilizar arreglos, listas enlazadas o librerías estándar. (Cálculos matemáticos, lectura de archivos)
- No sé puede utilizar ningún tipo de librería o API que ayude al procesamiento de los datos. (Listas, Pilas, Colas, Matrices, Vectores)
- Para el ingreso de las palabras se hará por medio de un archivo donde cada línea contendrá una lista de palabras separadas por coma, ejemplo:

<palabra1>, <palabra2>, <palabra3>, <palabra4>, <palabra5>, <palabra6>

Importante

- Lenguaje a utilizar C/C++.
- El programa debe ejecutarse en consola.
- Las malas prácticas de desarrollo serán penalizadas.
- Las copias obtendrán nota de cero y se notificará a coordinación.
- Se requiere la creación de un archivo Makefile, para obtener el compilado del programa.
- Esta práctica es obligatoria para tener derecho a realizar el siguiente proyecto.

Entrega

La fecha de entrega es el día 27 de Febrero. Los componentes se entregarán en un repositorio de Git a través Classroom, siendo estos:

- Código fuente.
- Documentación técnica.
- Manual de Usuario.

Nota

El repositorio debe ser privado, sin embargo debe de incluir los siguientes usuarios como colaboradores: **LuisCif-20, David15Barrera, daaaniel6** para la calificación correspondiente, estos deben ser agregados antes de la fecha y hora de entrega.