Laboratorio 5 – El TAD Lista

GUIÓN DEL LABORATORIO

1.- Objetivos del laboratorio

- Diseño de clases en C++
- Comprensión y uso del TAD Lista para resolver un problema.
- Razonar sobre el comportamiento y limitación de los algoritmos presentados

2.- Antes de asistir al laboratorio

Antes de asistir al laboratorio debes realizar las siguientes tareas:

1. Leer los apuntes de clase sobre el TAD Lista.
2. Revisar cuidadosamente el guión del laboratorio (este documento).
3. Contestar a la cuestiones 1 a 3 de la Actividad 0 del Prerrequisito.

Recuerda que es preciso entregar las respuestas a esta actividad antes de la fecha prevista. En caso contrario, no podrás acceder al laboratorio ni entregar de ninguna manera la solución a esta práctica.

3.- Actividades a realizar

Actividad 1: Implementación del TAD Lista usando estructuras dinámicas

1. Crear un proyecto y añadir los ficheros `Lista.h` y `Lista.cpp`. Este último fichero debe contener la implementación del TAD Lista utilizando estructuras dinámicas. La información que se almacena en cada nodo debe ser del tipo `String`.

Actividad 2: Uso del TAD Lista para resolver un problema.

Se propone la realización de un juego consistente en el que una aplicación informática debe adivinar cuál es la palabra de cinco letras en la que está pensado el jugador. La aplicación debe utilizar una Lista para trabajar con las palabras.

La descripción detallada del juego es la siguiente:

A. La aplicación carga en una lista el contenido de un fichero con palabras.
B. Se le pide al jugador que piense en una palabra de 5 letras.
C. Se selecciona aleatoriamente una palabra de la lista y se le muestra al jugador.
D. Se le pide al jugador que indique cuantas letras coinciden en la palabra mostrada respecto a la que ha pensado (no importa la posición en la que aparezcan). Supongamos que el usuario ha pensado en la palabra “rampa”
    a. Si la palabra seleccionada ha sido “estos” - cero coincidencias
    b. Si la palabra seleccionada ha sido “costa” – una coincidencia (una ‘a’)
    c. Si la palabra seleccionada ha sido “talan” – dos coincidencias (dos ‘a’)
    d. Si la palabra seleccionada ha sido “pared” - tres coincidencias (una ‘p’, una ‘a’ y una ‘r’).
E. Se eliminan de la lista todas las palabras que tengan un número diferente de coincidencias.

F. El proceso se repite (paso C) hasta que se acierte la palabra o la lista esté vacía indicando que no se ha podido adivinar la palabra (o el jugador ha mentido o la palabra no está en el fichero).

Las tareas a realizar se muestran a continuación:

1. En el proyecto creado para la actividad 1 añade los ficheros `AdivinarPalabra.h` y `AdivinarPalabra.cpp`.
2. En los ficheros `AdivinarPalabra.h` y `AdivinarPalabra.cpp` se debe declarar e implementar la clase `AdivinarPalabra` con el siguiente constructor:

   `AdivinarPalabra(string nomfich)`
   Constructor de la clase, recibe el nombre del fichero en el que se encuentran las palabras y añade las palabras a la lista.

   y los siguientes métodos públicos:

   `int contarNodos()`
   cuenta los nodos (=palabras) que hay en la lista.

   `void irAPosicionAleatoria(string &s)`
   asigna al string s el contenido de un nodo seleccionado de forma aleatoria entre los nodos que contiene la lista.

   `void eliminaDiferentes(const string s, const int numCoindicencias)`
   compara la palabra s con las palabras de la lista eliminando aquellas en las que el número de letras no coincida con el entero `numCoincidencias`.

   `void eliminaPalabra(const string s)`
   elimina de la lista la palabra que coincide con la que se pasa como argumento, s.

   `int cuentaCoincidencias(string s, string info)`
   cuenta el número de letras que coinciden entre la palabra s y la palabra info. Este método es utilizado en el método `eliminaDiferentes(…)`.

   Además, tiene los siguientes elementos privados:

   `Lista listapal`
   La lista de palabras.

3. Añade el fichero `main.cpp`. En la función `main` se debe crear un objeto de la clase `AdivinarPalabra` y se deben utilizar los métodos definidos en esta clase para jugar, además se debe mostrar en cada iteración el número de palabras que contiene la lista.

4. Contesta a las preguntas 4 a 7 de la hoja de trabajo.

Actividad 3: Mejoras sobre el programa:

1. Añade la posibilidad de guardar las palabras que se le muestran al jugador y las coincidencias que proporciona el mismo. El objetivo de es que al finalizar el juego se pueda detectar e informar al usuario si se ha equivocado (o ha mentido) a la hora de proporcionar el número de coincidencias.

2. Contesta a la pregunta 8 de la hoja de trabajo.

3. Ofrecer la posibilidad de añadir la palabra al fichero si no está.

Los archivos con los programas correspondientes a las actividades se deberán entregar, a través del aula virtual, al finalizar la sesión de prácticas, junto con la Hoja de Trabajo del estudiante. Los archivos C++ que se deben entregar son:

- `Lista.h y Lista.cpp`
- `AdivinarPalabra.h y AdivinarPalabra.cpp`
- `main.cpp`

4.- Después de asistir al laboratorio

Una vez finalizada la práctica, contesta a las cuestiones correspondientes a la Actividad 4 que aparecen en el Cuestionario Posterior al laboratorio.