# **Resultado de imagen para tec de monterrey logo png**

**Programación de Estructuras de Datos y Algoritmos Fundamentales**

# **Actividad 2.3 | *Actividad Integral Estructura de Datos Lineales (Evidencia Competencia)***

**Alumno:**

Carlos Antonio Buendia López A01379471

Luis Ángel Terrazas García A01377440

**Profesor(a):**

### **Víctor Adrián Sosa Hernández**

06/11/2020

Investigación:

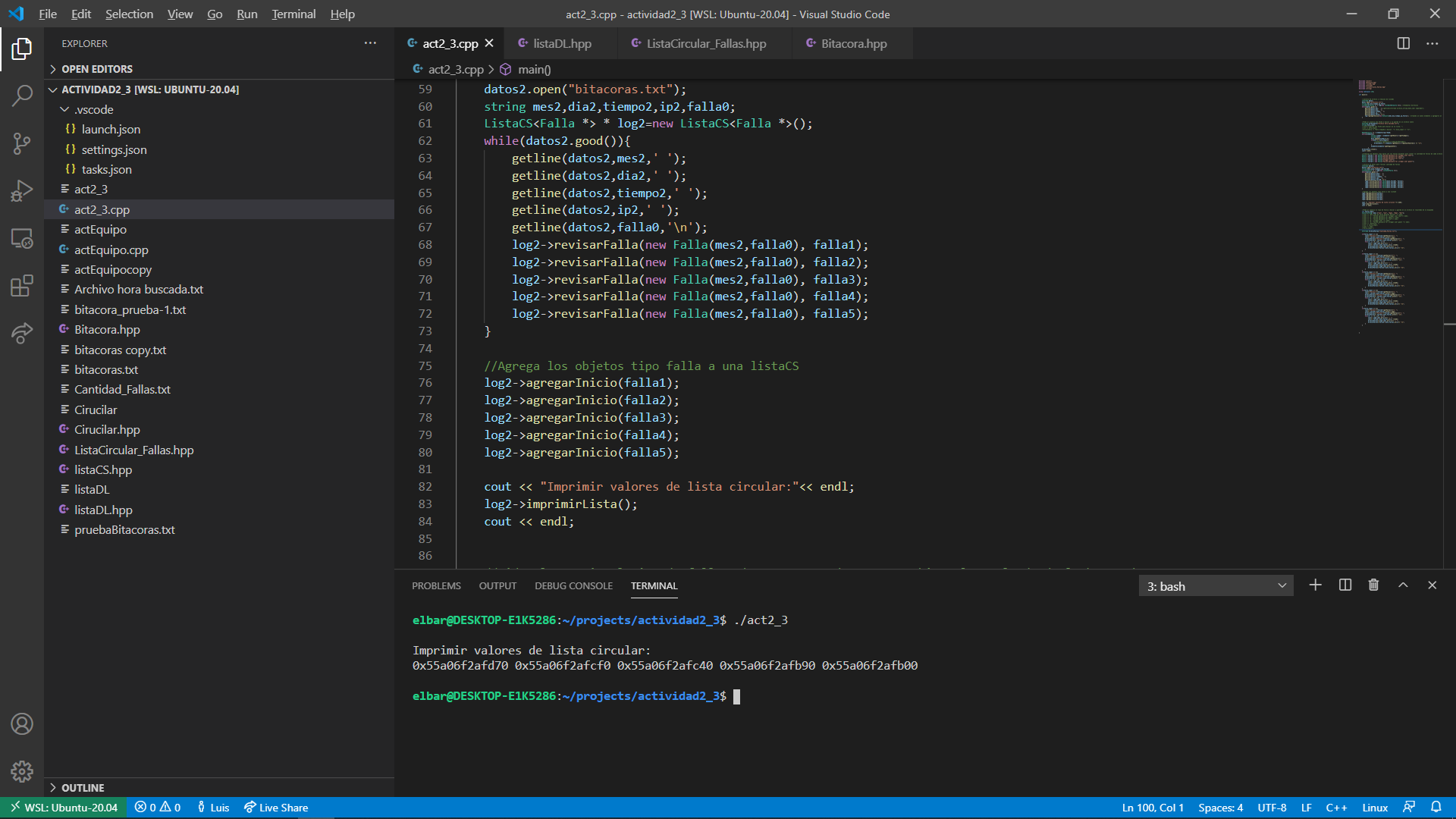
En estructura de datos y en específico un algoritmo de las listas doblemente ligadas es una lista enlazada a partir de nodos, la importancia de estas listas es que son mayormente utilizadas para las operaciones de inserción y eliminación de datos en la lista. Cada nodo de la lista doblemente ligada usa además del su dato, dos punteros, cuyos punteros se usan para especificar cual es el elemento antecesor y sucesor, permiten moverse dentro de las listas ya sea un registro adelante o un registro atrás a medida que tomen las direcciones de uno u otro puntero.

Hay dos tipos de listas doblemente ligadas las cuales son listas dobles lineales y las dobles circulares. En el primer caso en este tipo de lista el puntero de la liga inicial(inicio de la lista) apunta a *Null* y el otro puntero señala liga final (final de la lista) a *Null*. Y en el segundo caso este tipo de lista a diferencia del anterior el puntero de la liga inicial apunta al último elemento de la lista y el último elemento de la lista se ordena en el puntero de la liga final al primer elemento de la lista.

Para la operaciones de inserción y eliminación de nodos para este proceso es más sencillo ya que es suficiente con localizar al nodo para poder añadirlo y/o eliminarlo ya sea el anterior o el posterior, teniendo en cuenta en todo instante la dirección del nodo anterior.

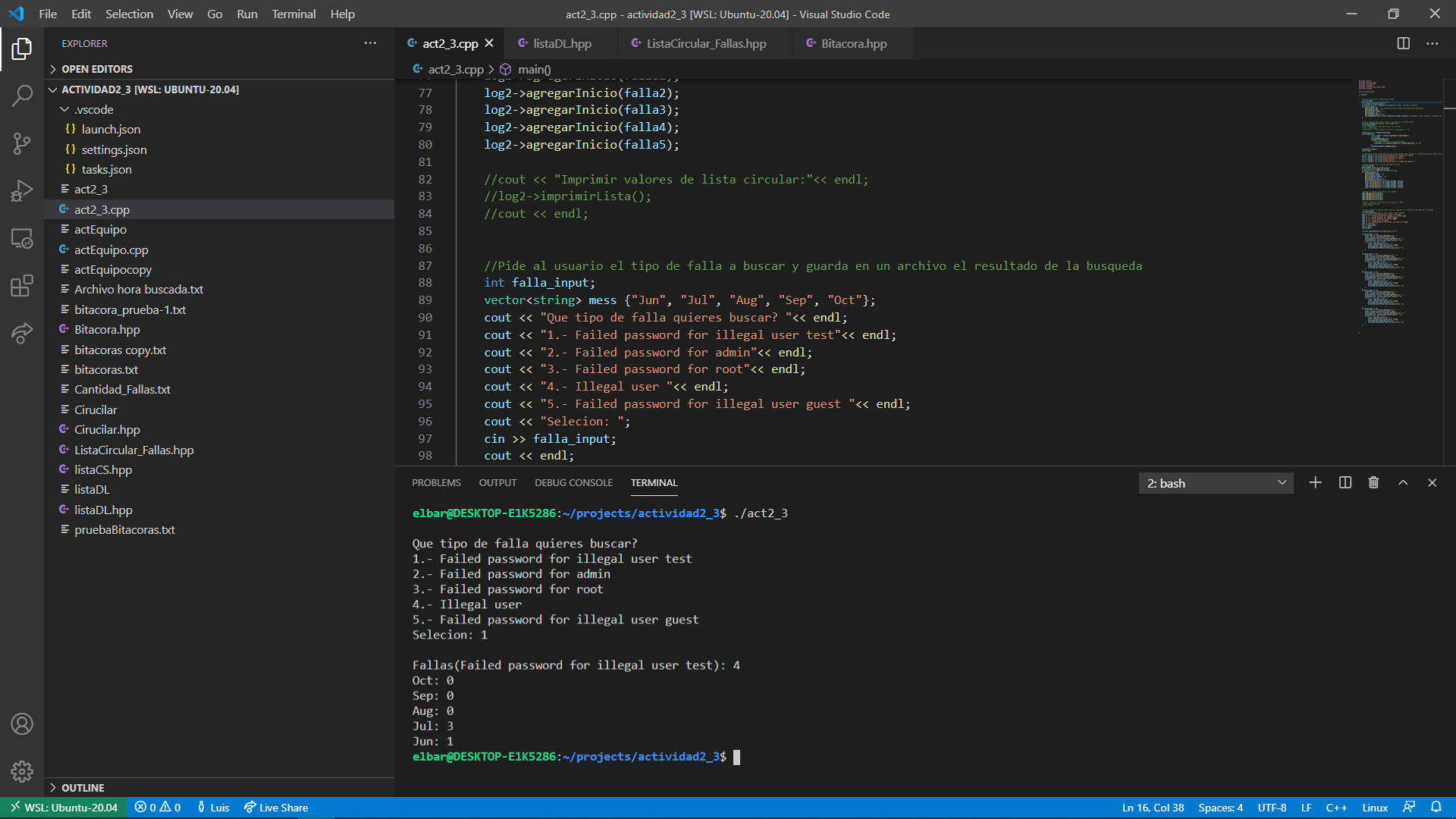
Screenshots:

* Lista circular

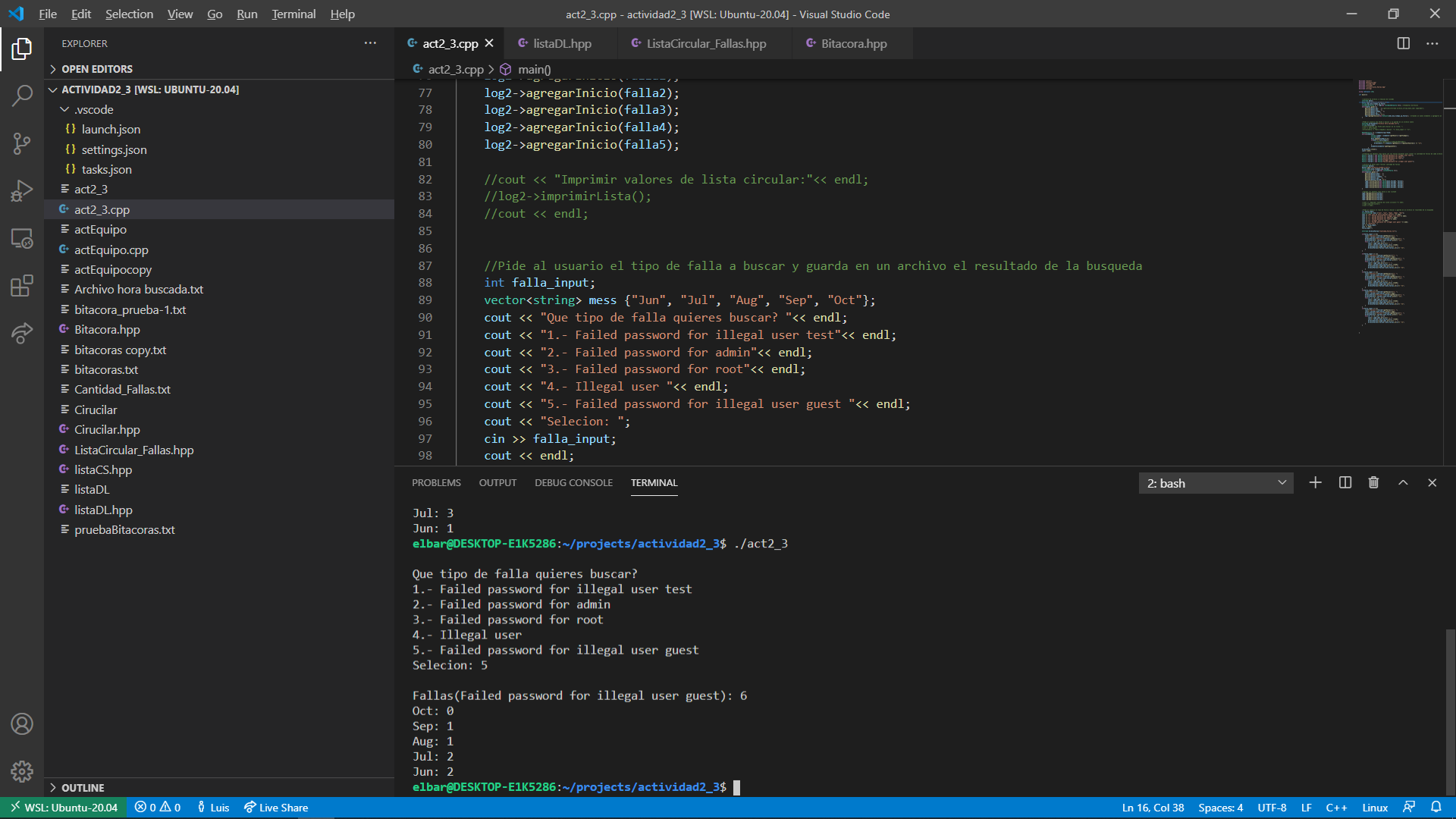


* Solicite al usuario el tipo de falla y muestre el número de ocurrencias totales y la información del número de ocurrencias divididos por mes.

*Ejemplo1:*

**

*Ejemplo 2:*

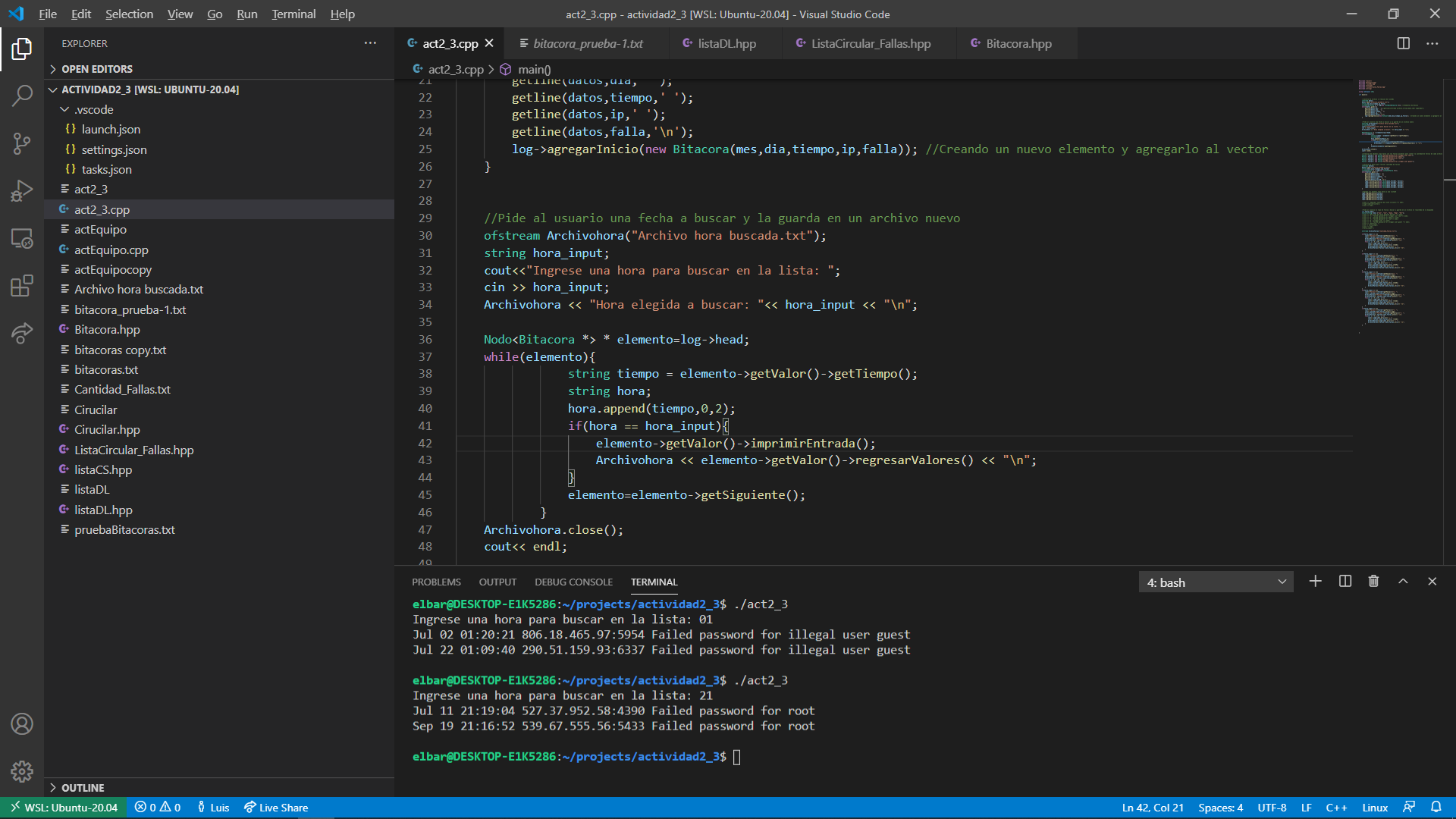
**

Complejidad:

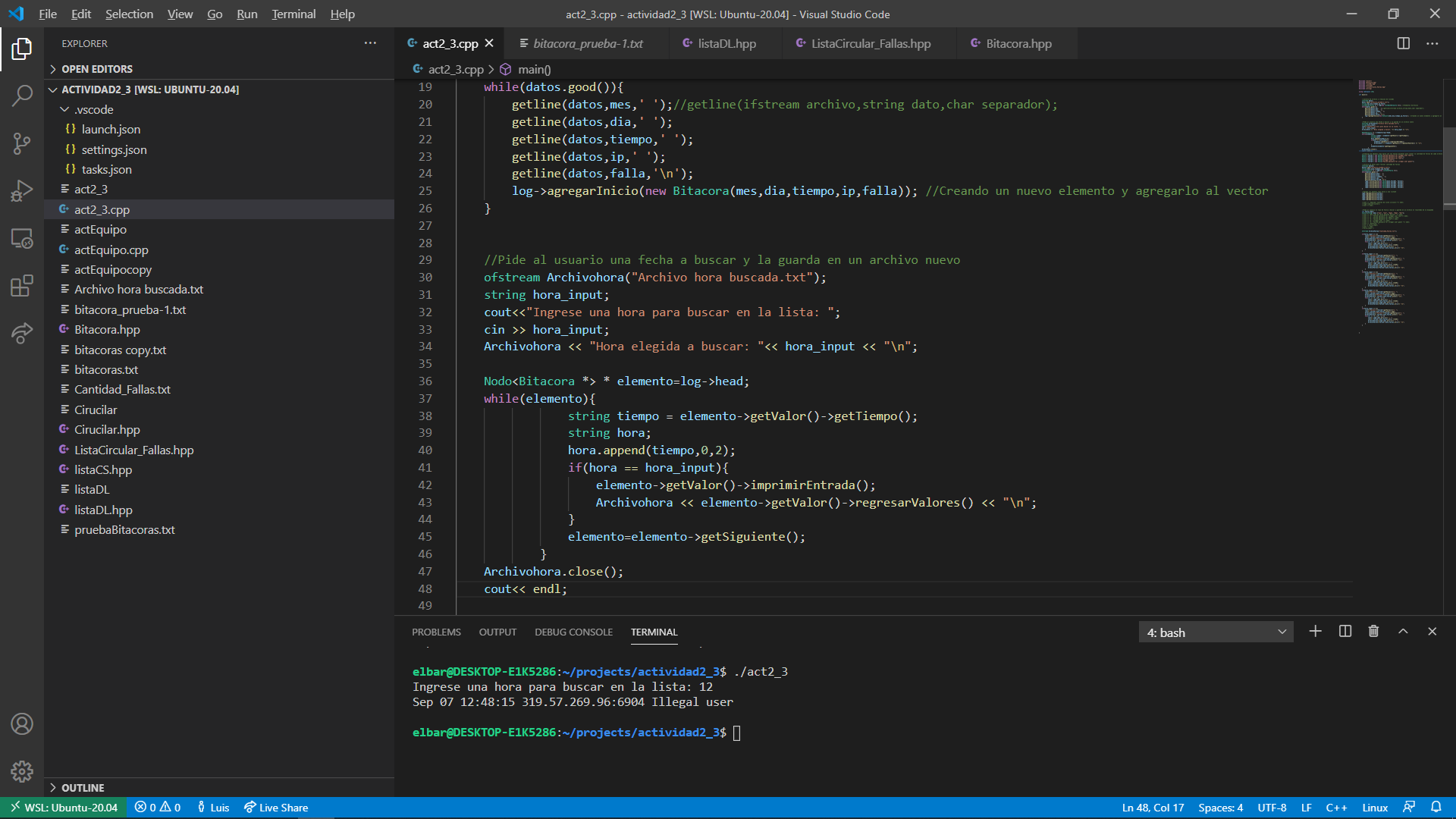
La complejidad de este parte del código utiliza la función agregar al inicio en la lista circular 5 veces para agregar los 5 objetos tipo falla a la lista circular. Este código también recorre todas las bitácoras del archivo y las suma a las variables de cada objeto que falla en la lista circular para conseguir el número de ocurrencias que tiene cada falla, esto utiliza la función revisar falla en la lista circular la cual tiene una complejidad de BigO(n). Al usar la función agregar al inicio tenemos una complejidad BigO(1) y un BigO(n) al tener como máxima complejidad BigO(n) esta parte del código tiene complejidad de BigO(n).

* Solicite al usuario una hora por ejemplo 01, 13, 21 y posteriormente deberá desplegar todos los registros de la lista doblemente ligada que tengan esa hora en específico. Los minutos y segundos no importan.

*Ejemplo1:*

**

*Ejemplo2:*

**

Complejidad:

En esta parte del código se recorre la lista doblemente ligada para revisar cada objeto, sacar su valor e imprimirlo, esto se hace de una manera secuencial buscando por cada nodo de la lista. Si tenemos en cuenta la complejidad de crear los objetos tipo bitácora, agregarlos a la lista como nodo con la función de agregar al Inicio, volver a revisar la lista y buscar la información de hora tendríamos como resultado que la complejidad de este código es BigO(n) ya que todas las operaciones anteriores no excedan de tener una complejidad de BigO(n).

Conclusión:

* Luis Ángel Terrazas García:

En esta actividad nos enfrentamos a diferentes retos como lo fueron conseguir acceder y agregar objetos en la lista circular y doblemente ligada. Con esta actividad aprendí bastante sobre el manejo y la utilidad de estas listas, durante la creación del código en equipo debatimos y reflexionamos sobre cómo hacer el código y cómo resolver el problema de contar las fallas y dividirlas por meses. Como conclusión me pareció una actividad bastante interesante la cual implementó bastante bien los conocimientos que obtuvimos en clase desde lectura de archivo, operadores, tipos de listas, etc..

* Carlos Antonio Buendia López:

En la actividad aprendimos mucho de la implementación de este tipo de algoritmos en el lenguaje de programación de C++, nos enfrentamos a distintos problemas y discutimos en equipo de la forma en cómo se debería de implementar los algoritmos. Algo que se nos complicó mucho fue aprender cómo introducir objetos a la lista doblemente ligada y la circular, con prueba y error fuimos implementando las ideas hasta que salió. Aprendí cómo implementar de una mejor manera los punteros, como manejar los objetos en el lenguaje de programación, los tipos de listas, los diferentes usos que le podemos dar, los algoritmos que están detrás de la programación de los mismos,etc.

Referencias:

1. Joaquín Fernández Valdivia, Exposito Lopez Daniel, Abraham García Soto, Martin Gomez Antonio Jose. (N/A). *LISTAS DOBLEMENTE ENLAZADAS*. 04/11/2020, de Universidad de Granada Sitio web: <http://decsai.ugr.es/~jfv/ed1/tedi/cdrom/docs/ldoble.html>
2. José R. García-Bermejo Giner. (2006). *ASIGNACIÓN DINÁMICA: LISTAS DOBLEMENTE ENLAZADAS*. 04/11/2020, de Universidad de Salamanca y Departamento de Informática y Automática Sitio web: <http://maxus.fis.usal.es/FICHAS_C.WEB/11xx_PAGS/1104.html>
3. Ing. Bruno López Takeyas, M.C.. (N/A). *Estructuras de Datos Orientada a Objetos*. 04/11/2020, de Instituto Tecnológico de Nuevo Laredo Sitio web: <http://www.itnuevolaredo.edu.mx/takeyas/Apuntes/Estructura%20de%20Datos/Apuntes/03-ListasDobles.pdf>