**Tarea corta 2, Estructuras de Datos, grupo 3**

**Profesor: Juan Carlos Gómez**

**Alumnos:**

**Luis Elizondo Varela 2017096778**

**Marco …..**

**Fecha de entrega : 15 de marzo 2019**



Tabla de contenido

[**Introducción:** 3](#_Toc3314340)

[**Código y explicación de clase ListB:** 3](#_Toc3314344)

[**Uso y aplicación de GitHub con Visual Studio:** 5](#_Toc3314345)

[**Resultados Obtenidos:** 6](#_Toc3314346)

[**Conclusiones:** 6](#_Toc3314347)

[**Referencias:** 6](#_Toc3314348)

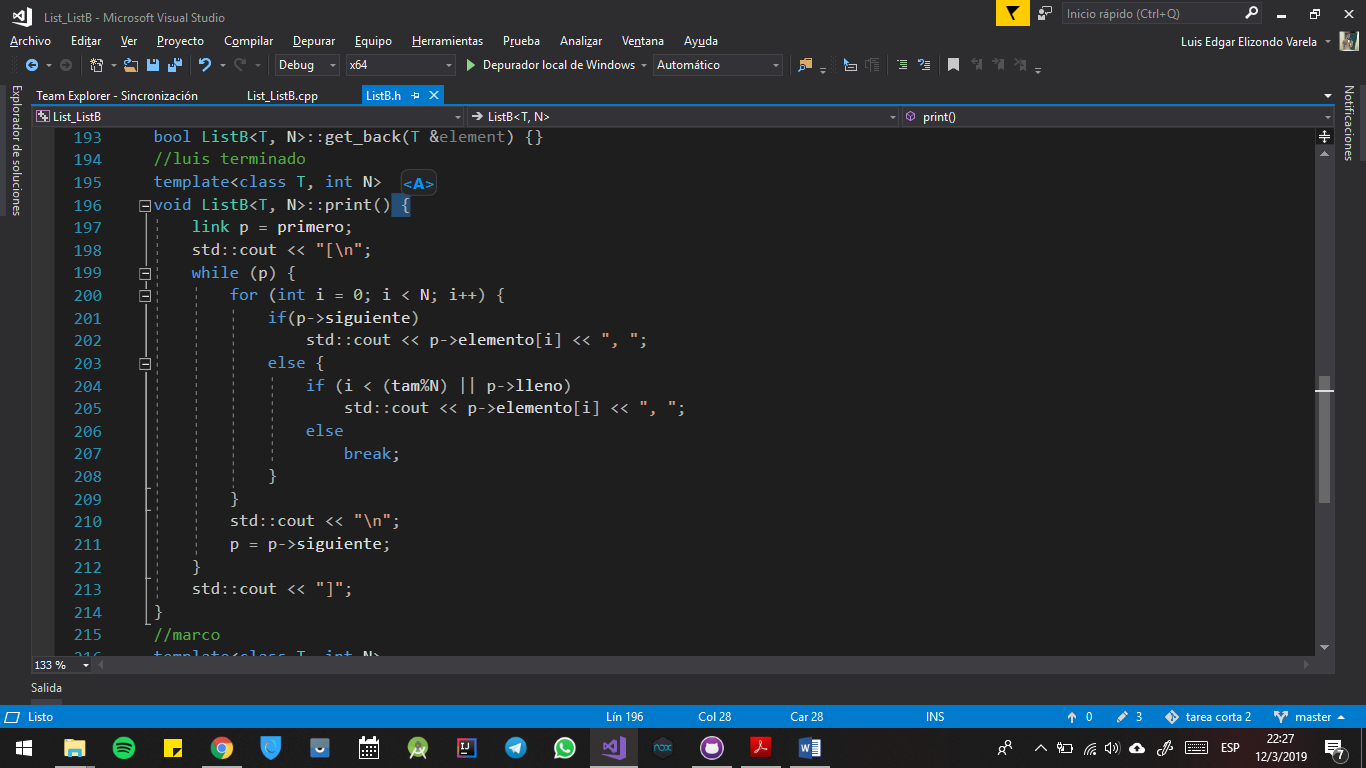
**Introducción:**

En esta tarea corta se elaborará un programa en el cual se pondrá a prueba los conocimientos del estudiante para la aplicación de punteros en C++, se crearan los métodos de dos clases, las cuales corresponden a clases de listas enlazadas, la primer clase es una lista enlazada simple que recibe un elemento de tipo T usando templates, la segunda clase es muy similar a la primera, solo que en el template recibe un N el cual será el numero de elementos que se puede guardar en un nodo.

Como el trabajo es en un grupo de dos, los métodos de las dos clases se dividirán equitativamente.

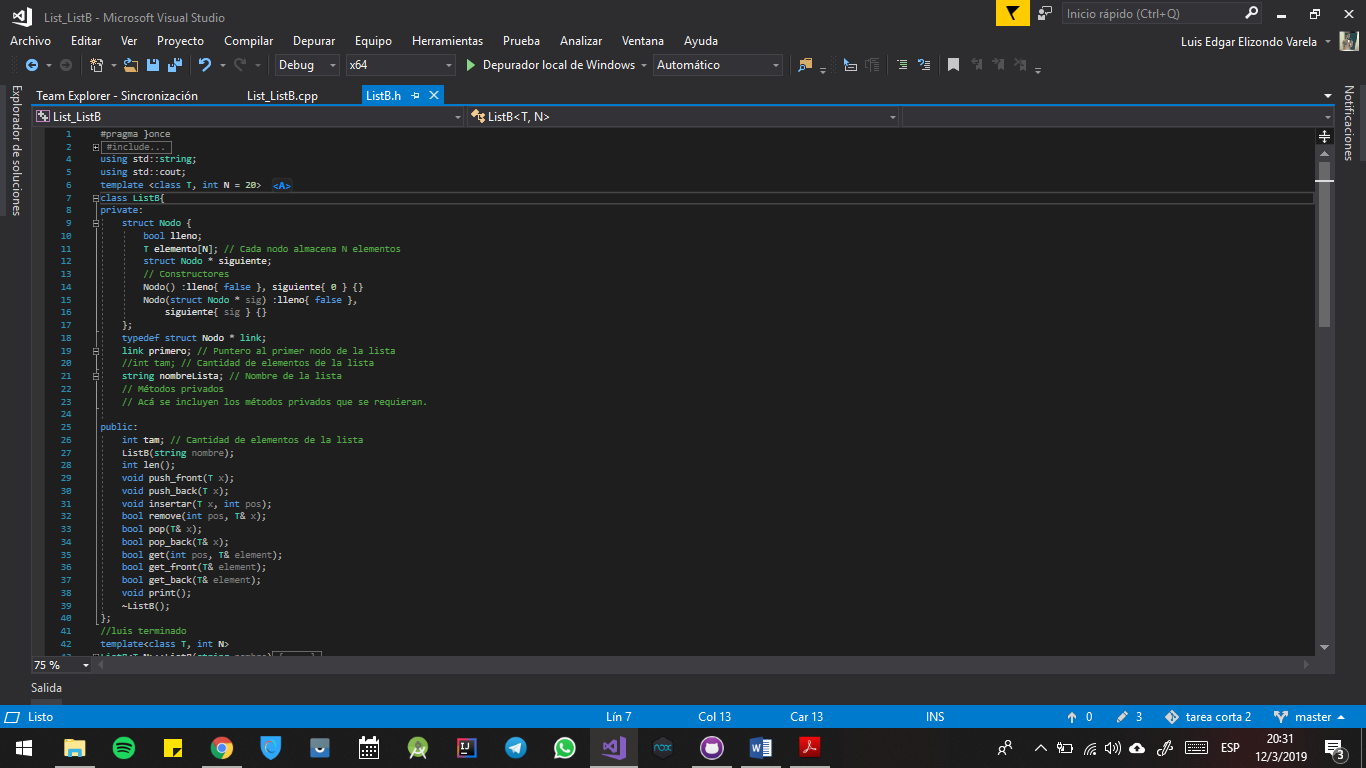
Para control de código se usara GitHub y el ambiente de desarrollo Visual Studio, de igual manera se tenia que investigar como vincular el proyecto de GitHub con Visual Studio, para así tener una mayor comodidad a la hora de trabajar en grupo al mismo tiempo en un mismo proyecto.

**Código y explicación de clase ListB:**

****

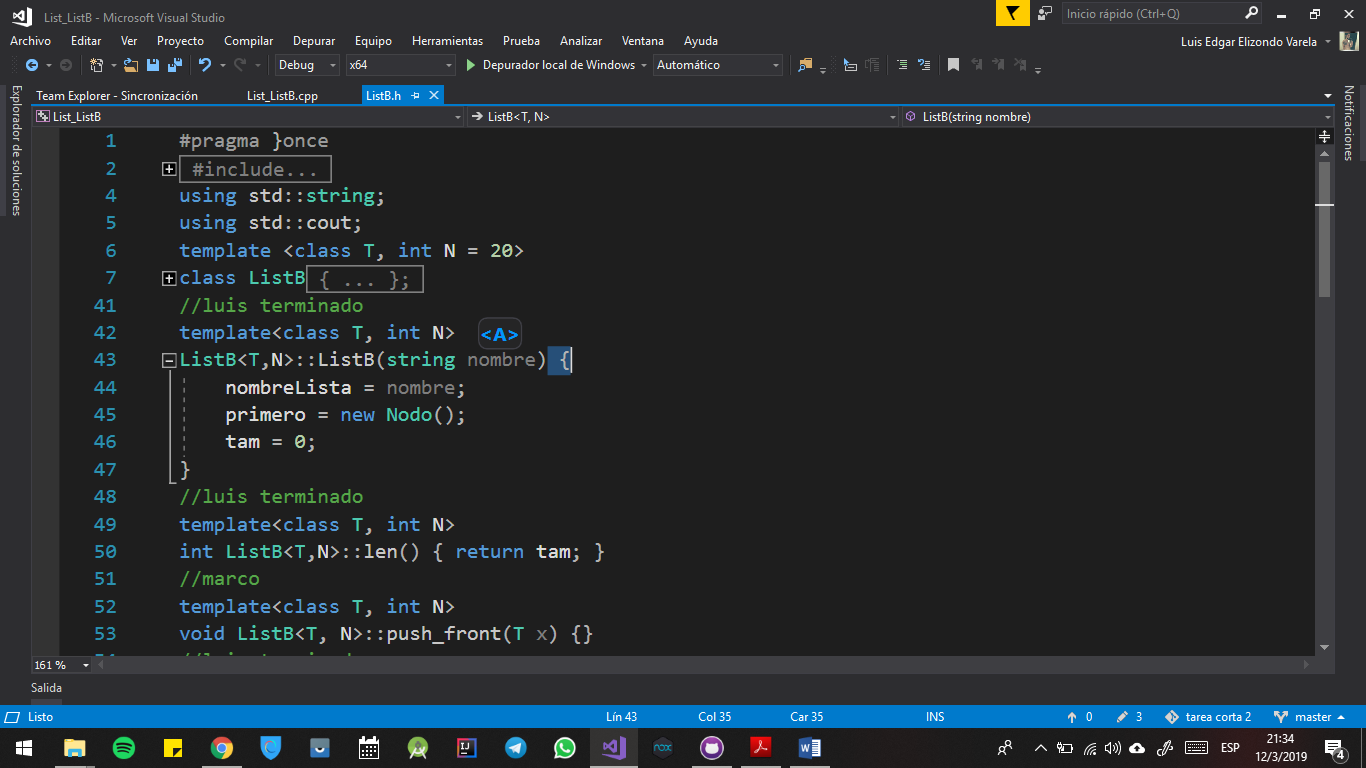
A continuación, se mostrará cada uno de los métodos de ListB y la explicación del algoritmo:

Cuerpo de la clase:



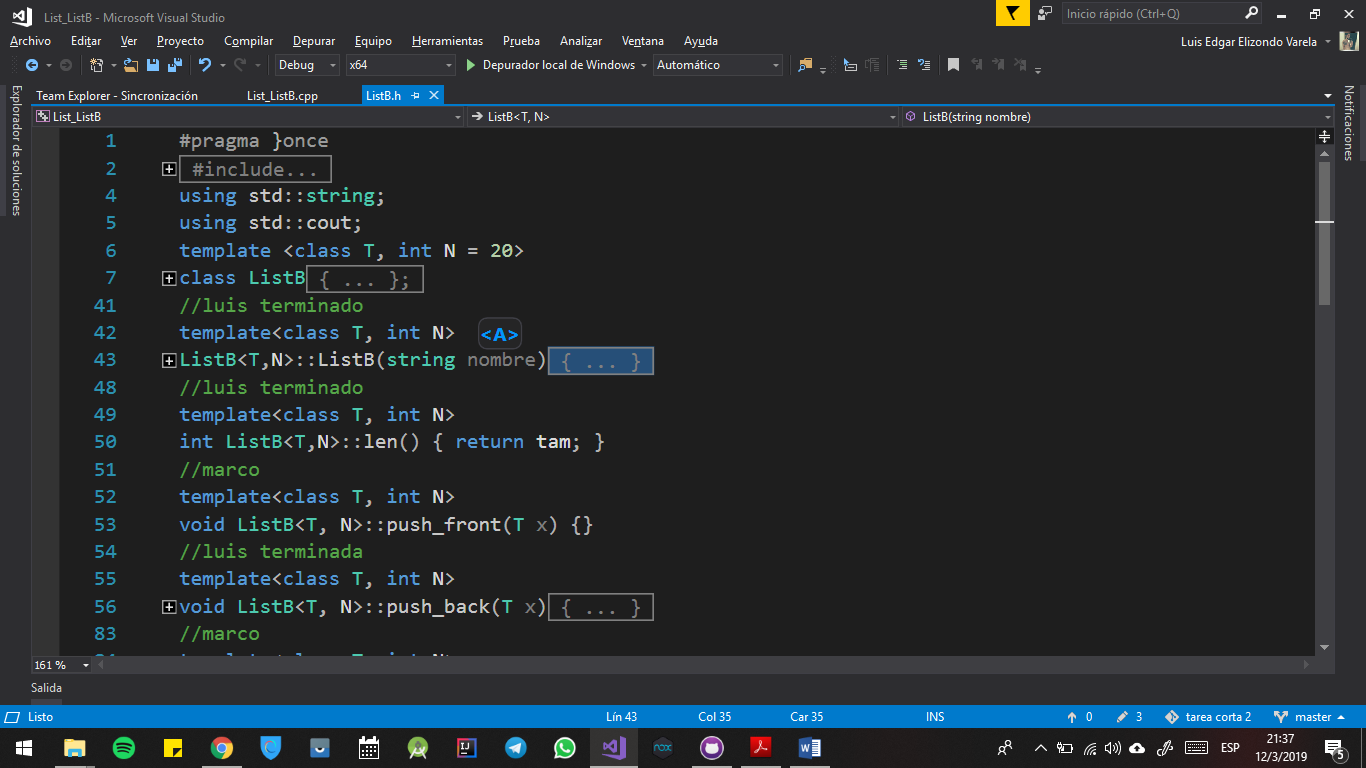
Como podemos ver, utiliza un template para el tipo de elemento que guardara en los nodos y un int que representaría la cantidad de elementos a guardar por nodo, la clase contiene el struct nodo, su arreglo de elementos de tipo T, sus constructores, un boleano para saber si se lleno el nodo, su puntero al inicio de la lista enlazada y un string nombre que hace referencia al nombre propio de la lista; más abajo están todos los métodos, contructor y destructor.

Constructor:



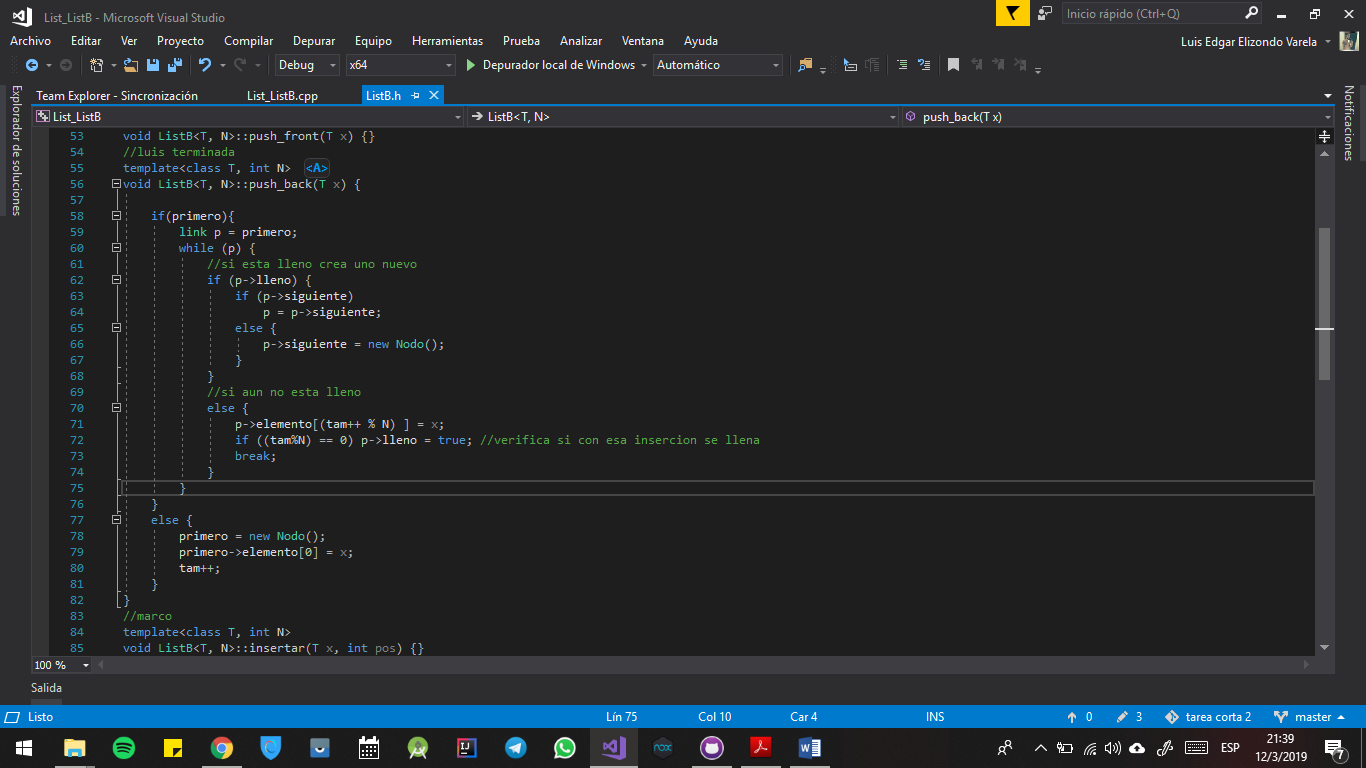
Aquí se inicializa la lista enlazada, su tamaño y su nombre propio.

Función que retorna el largo:



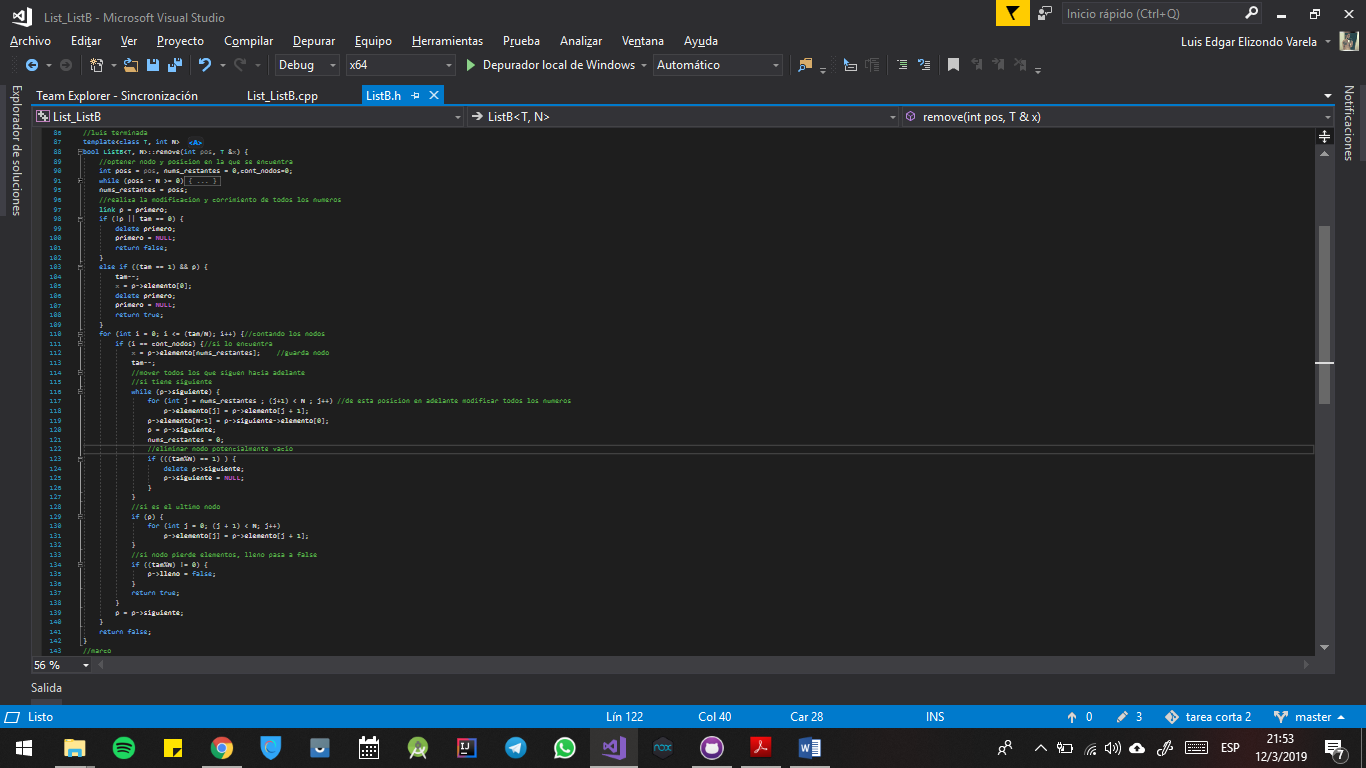
Retorna un int con la información del largo que tiene la lista enlazada, no lo cuenta por nodos, si no por elementos guardados en cada uno de los nodos.

Función que mete al final de la lista:



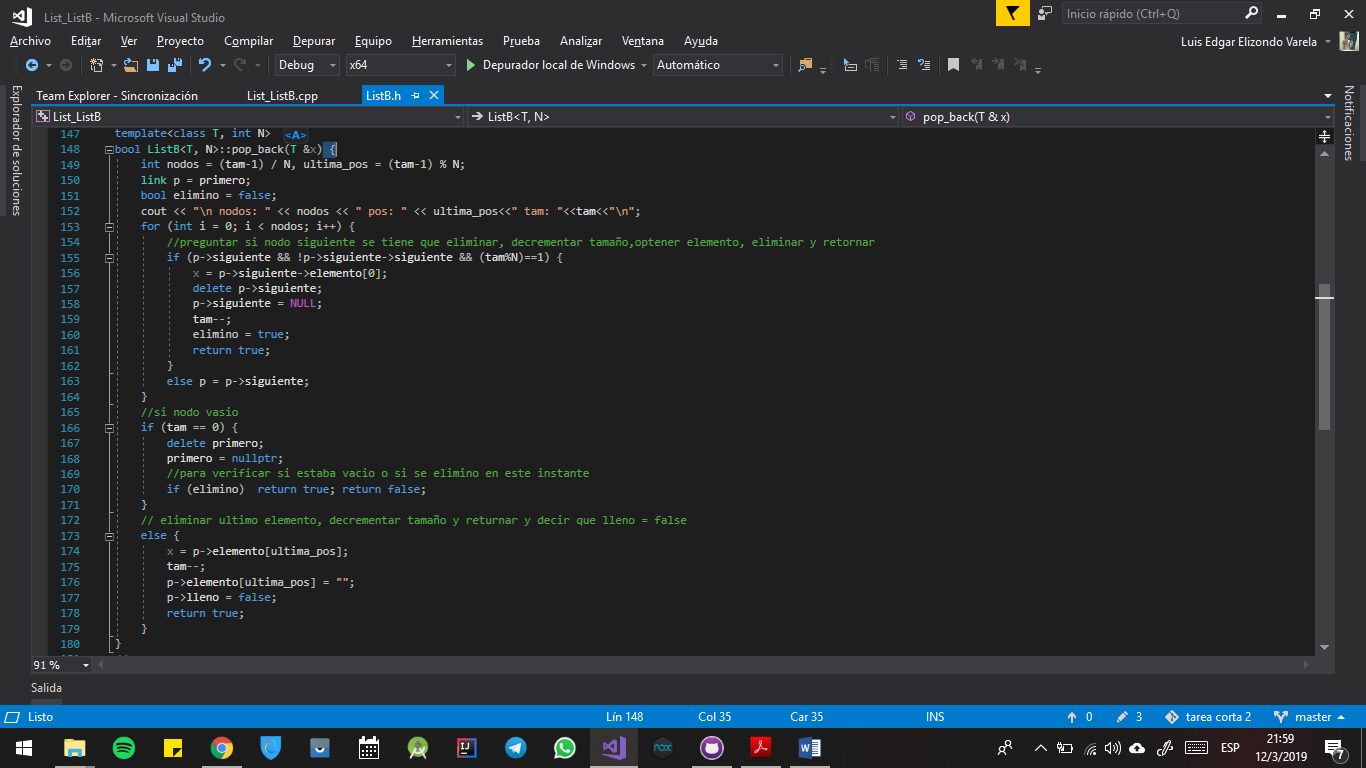
En esta función, el algoritmo recorre todos los nodos, verificando si esta lleno o si es el ultimo nodo para posicionar el elemento, igual mente si el último elemento que añadió hace el nodo lleno cambia el atributo a true, si recorre todos los nodos y no haya donde meter el valor, crea un nodo nuevo y agrega el valor, esta última opción funciona en el caso de que la lista este totalmente vacía.

Función eliminar en posición:



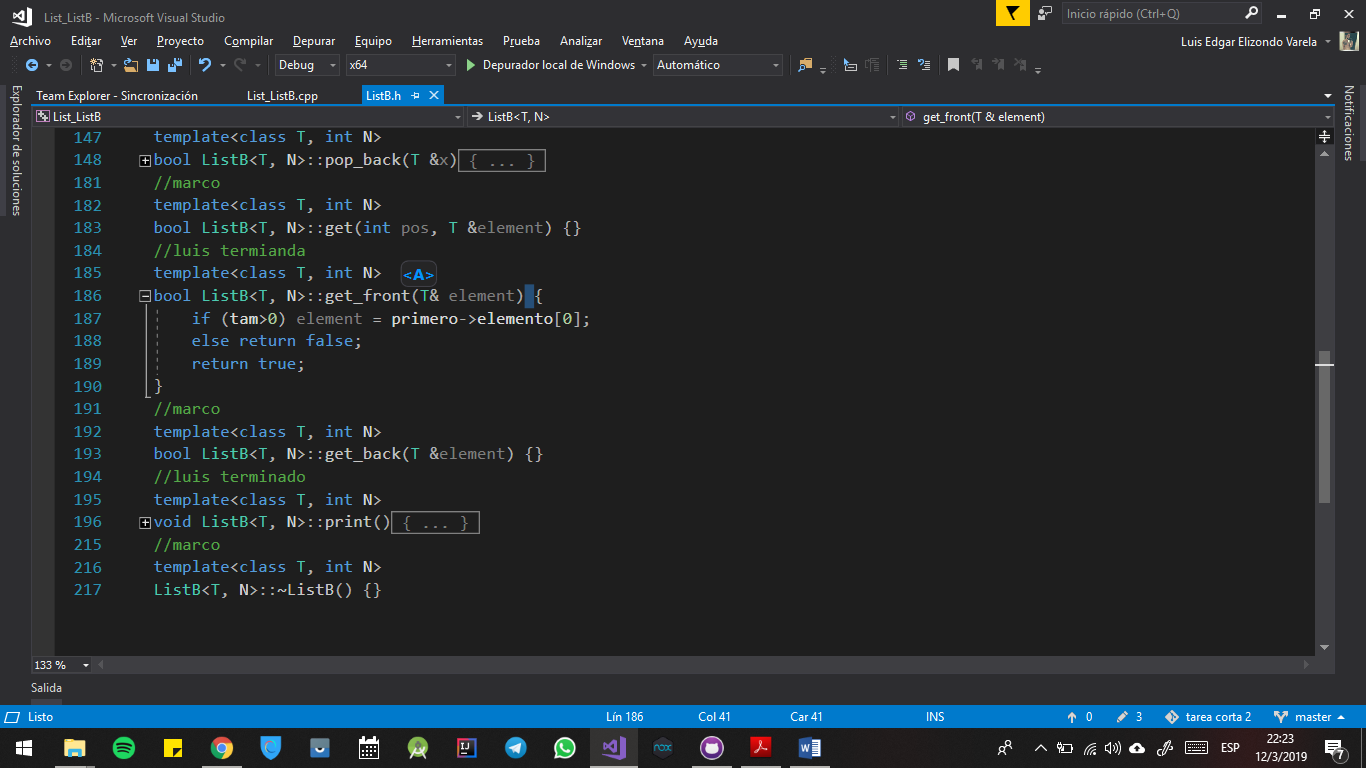
Esta función encuentra el valor a eliminar, y recorre toda la lista hasta el ultimo elemento corriendo un lugar hacia adelante todos los valores, verifica si el ultimo nodo tiene un solo elemento para que al terminar el corrimiento sea eliminado. Al igual se hacen los casos de si esta vacío. Para la ubicación del nodo a eliminar se realiza una operación de modulo y de división con el atributo tam, de tamaño, para saber cuantos nodos recorrer y en qué posición del nodo esta el elemento a eliminar.

Función elimina último elemento:



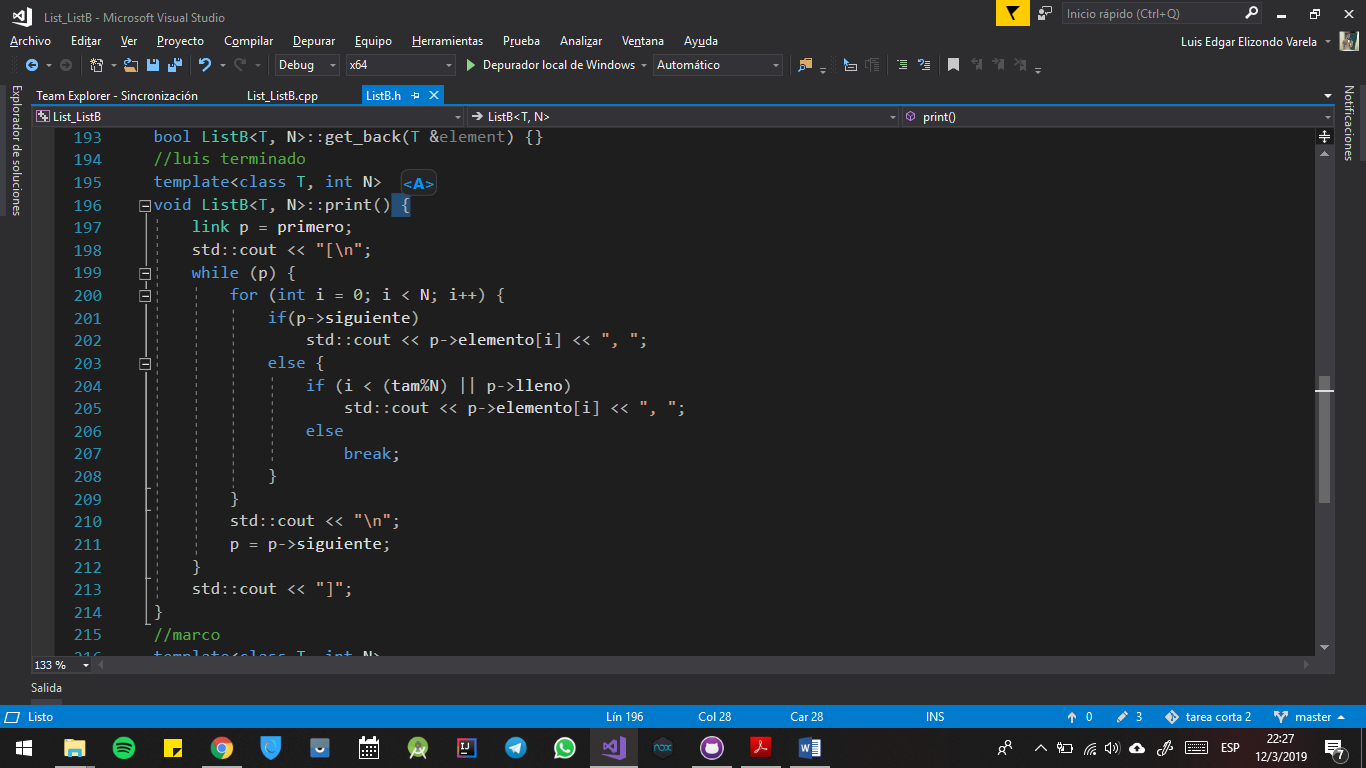
Esta función llega hasta el ultimo nodo y pregunta si tiene un único elemento para eliminar el ultimo nodo, y si no elimina el ultimo elemento de ese nodo, al igual hace validaciones para lista vacía y para nodo cambiar lleno a false.

Función obtener primer elemento:



En esta función se pregunta si hay elementos y se guarda el primer elemento y si no retorna false.

Función imprimir lista:



Esta función saca la cantidad de nodos de acuerdo con el tamaño con una división, y de acuerdo con eso recorre con un for imprimiendo por cada renglón todos los elementos de un nodo, igual hace las validaciones para cuando esta vacío.

**Uso y aplicación de GitHub con Visual Studio:**

**Resultados Obtenidos:**

Se abordo cada problema con éxito, haciendo una buena estrategia de pruebas para cada uno de los métodos de la clase, los métodos funcionan de la mejor manera.

**Conclusiones:**

Se practico los punteros en C++, de esta manera se pudo ver los errores comunes que se cometen a la hora de programar, y también una mejor manera de implementarlos o usarlos

**Referencias:**