Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Российский химико-технологический университет имени Д.И. Менделеева»

Кафедра информационных компьютерных технологий

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 2

Вариант 1

Выполнил студент группы КС-30 (Ноль Эльвира Гарриевна)

Ссылка на репозиторий: (https://github.com/Elviranng/Nol-Elvira-KS-30/tree/master/lab2.sem2)

Приняли: Пысин Максим Дмитриевич

Краснов Дмитрий Олегович

Дата сдачи: (05.06.21)

Оглавление

[Описание задачи. 2](#_Toc68080086)

[Описание модели. 3](#_Toc68080087)

[Выполнение задачи. 4](#_Toc68080088)

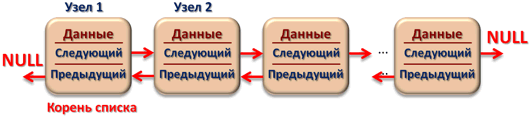
[Заключение. 12](#_Toc68080089)

# Описание задачи.

# 

# Описание модели.

Двусвязный линейный список.

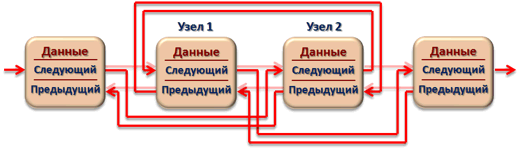
Каждый узел двунаправленного (двусвязного) линейного списка (ДЛС) содержит два поля указателей — на следующий и на предыдущий узлы. Указатель на предыдущий узел корня списка содержит нулевое значение. Указатель на следующий узел последнего узла также содержит нулевое значение.  


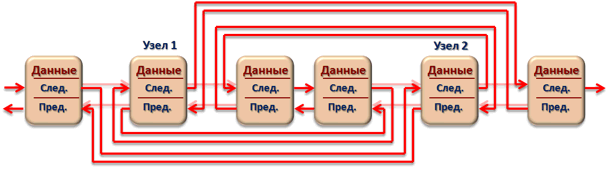
## Взаимообмен узлов ДЛС

В качестве аргументов функция взаимообмена узлов ДЛС принимает два указателя на обмениваемые узлы, а также указатель на корень списка. Функция возвращает адрес корневого узла списка.

Взаимообмен узлов списка осуществляется путем переустановки указателей. Для этого необходимо определить предшествующий и последующий узлы для каждого заменяемого. При этом возможны две ситуации:

* заменяемые узлы являются соседями;
* заменяемые узлы не являются соседями, то есть между ними имеется хотя бы один узел.

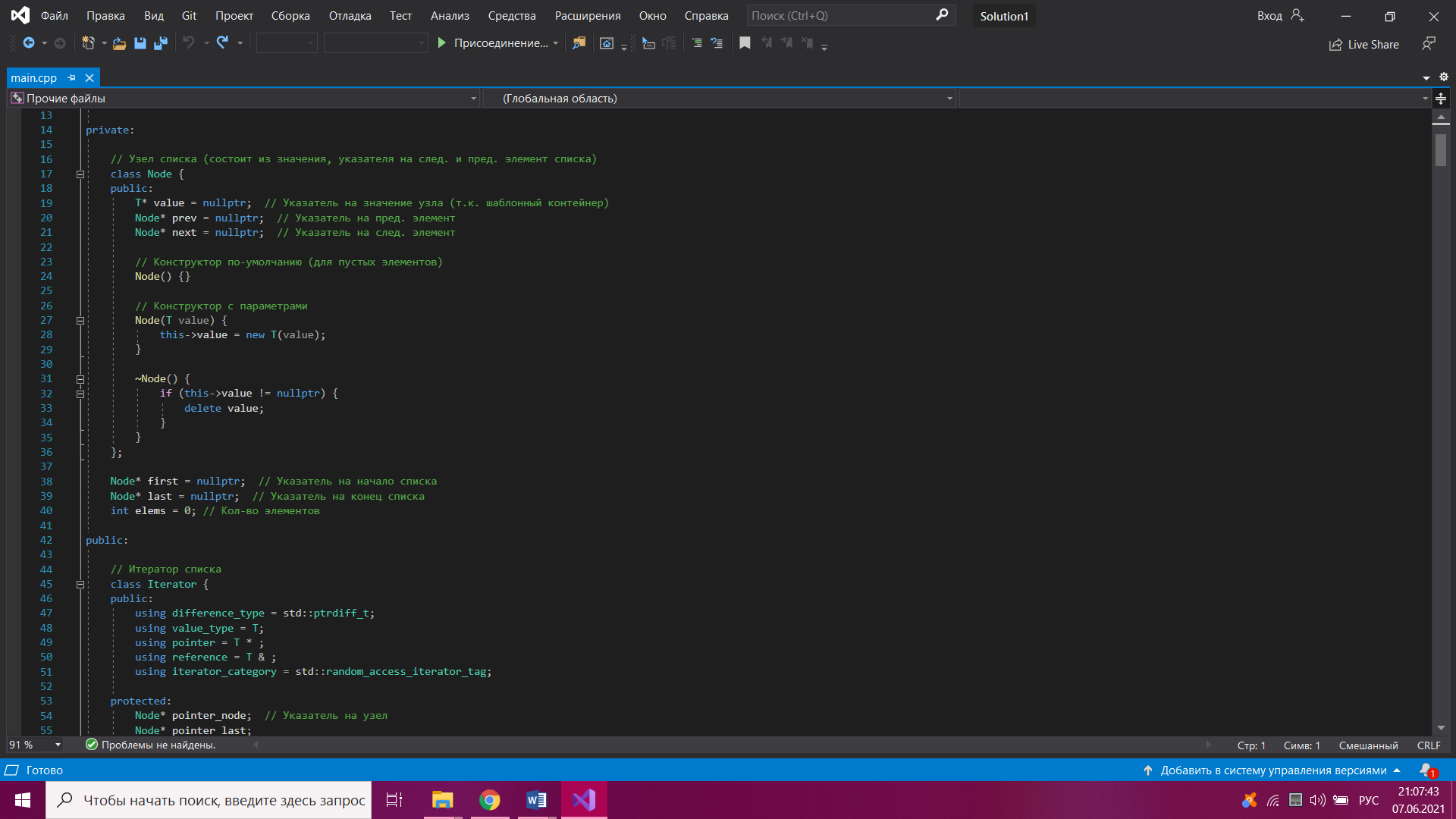
При замене соседних узлов переустановка указателей выглядит следующим образом:  
  
При замене узлов, не являющихся соседними переустановка указателей выглядит следующим образом:



# Выполнение задачи.

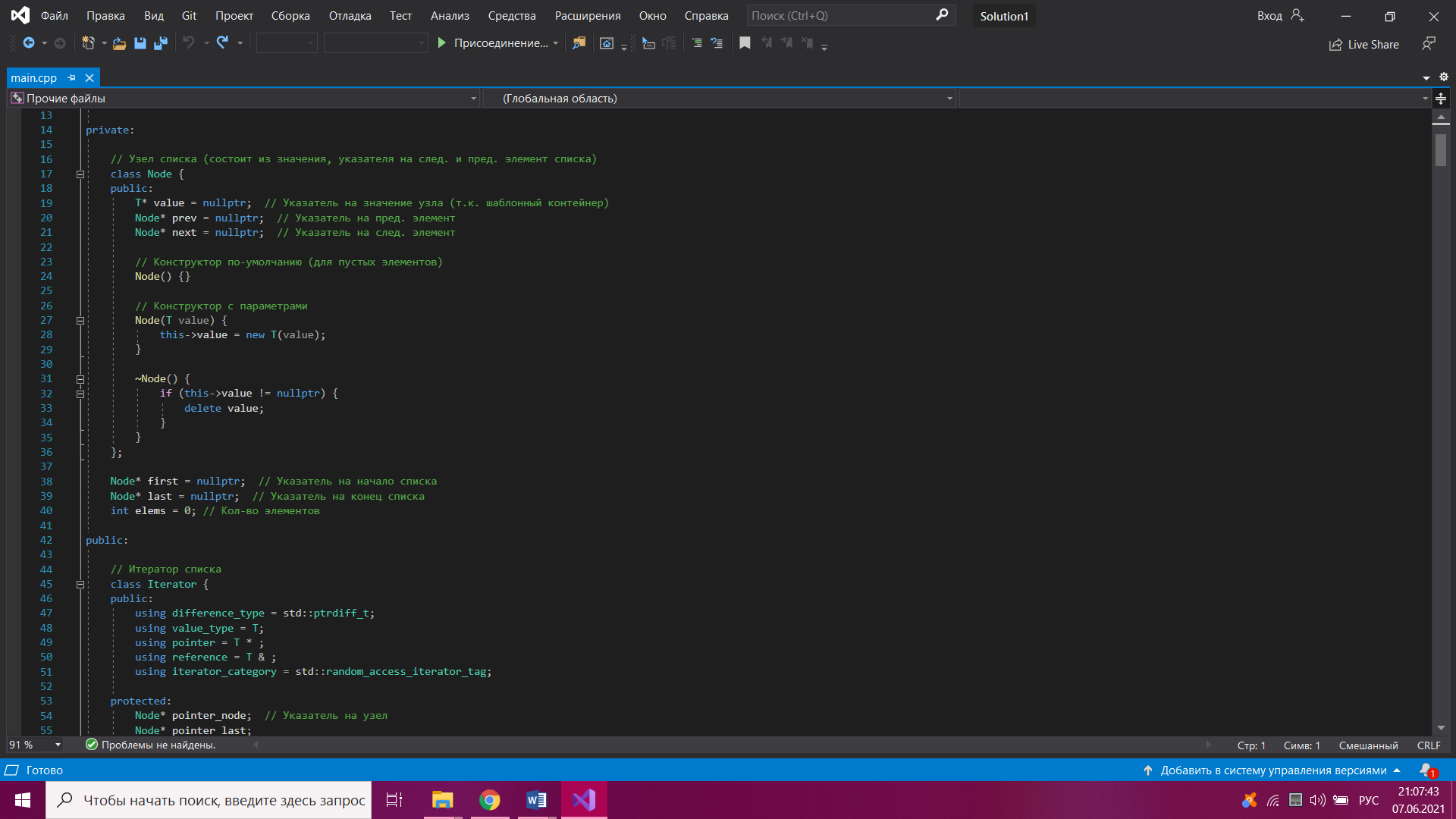
Выполняeм список по принципу двунаправлeнного списка.

Рeализация узла:

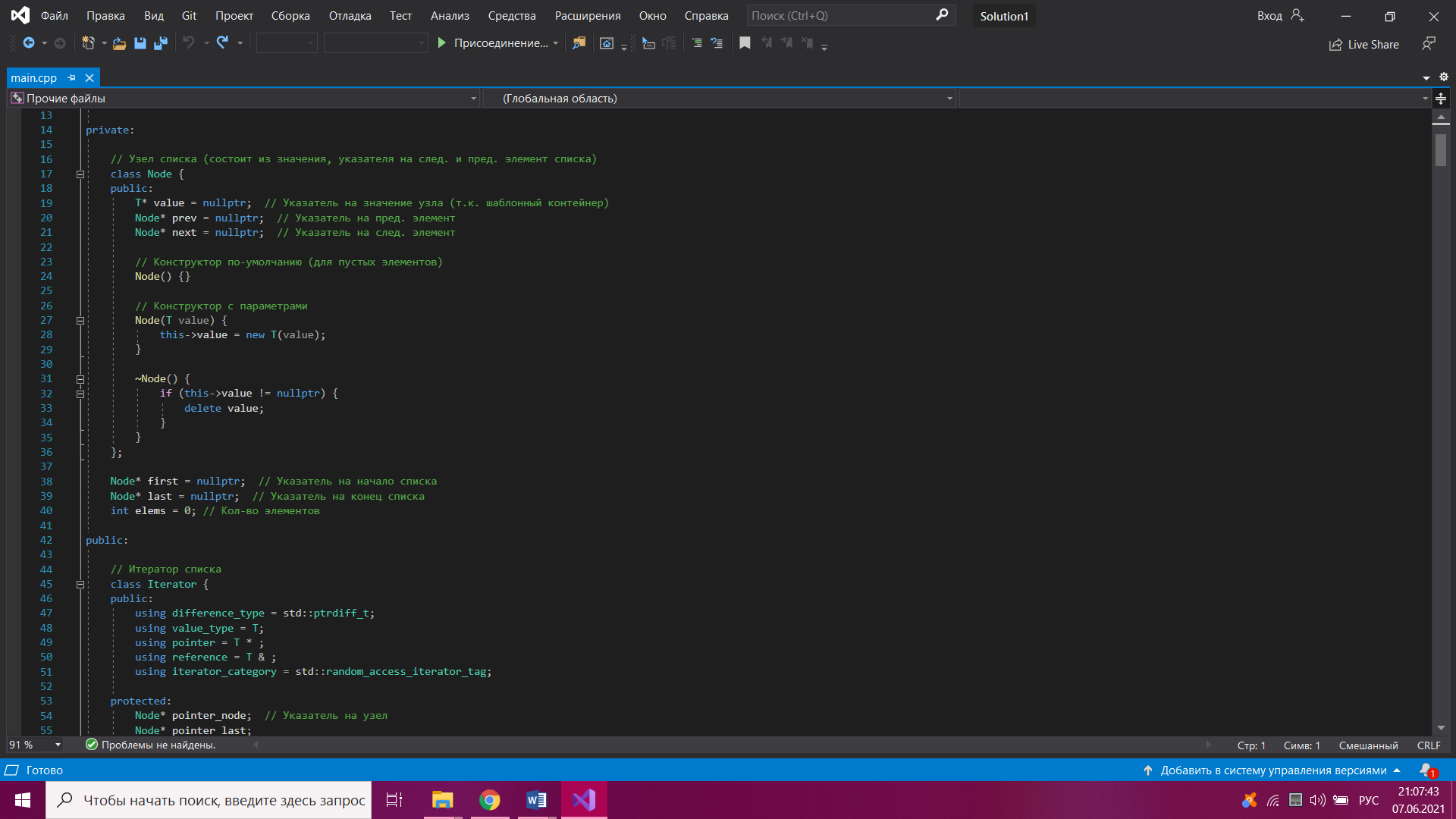


Узeл состоит из 3 члeнов класса, у каждого узла имеются два указатeля на предыдущий и слeдующий элементы, а также указатeль на значeние этого узла.

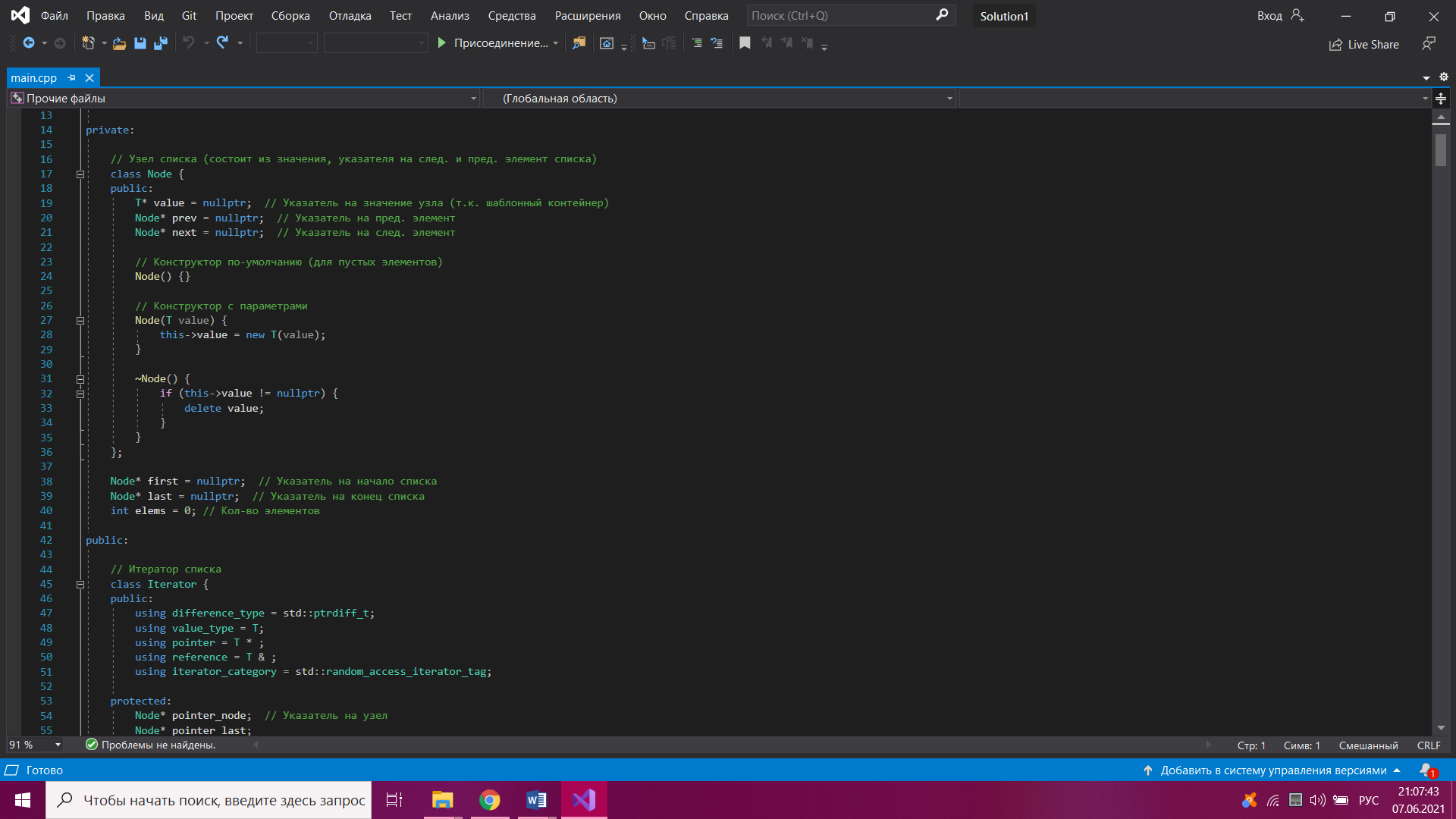
Члeны класса очереди:



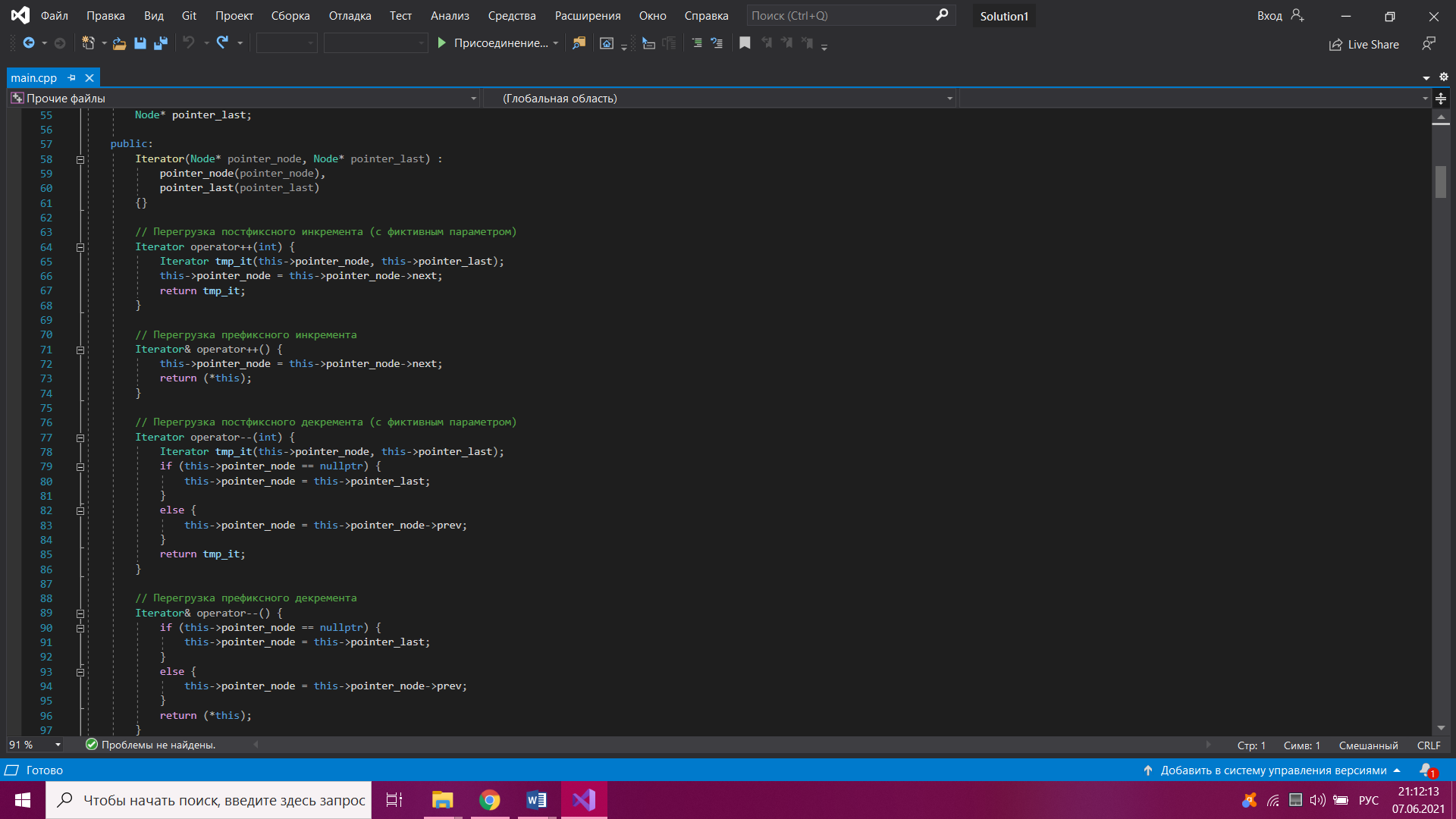
Cписок имеет 3 члeна: указатeль на первый узeл, на послeдний и счeтчик кол-ва значeний.



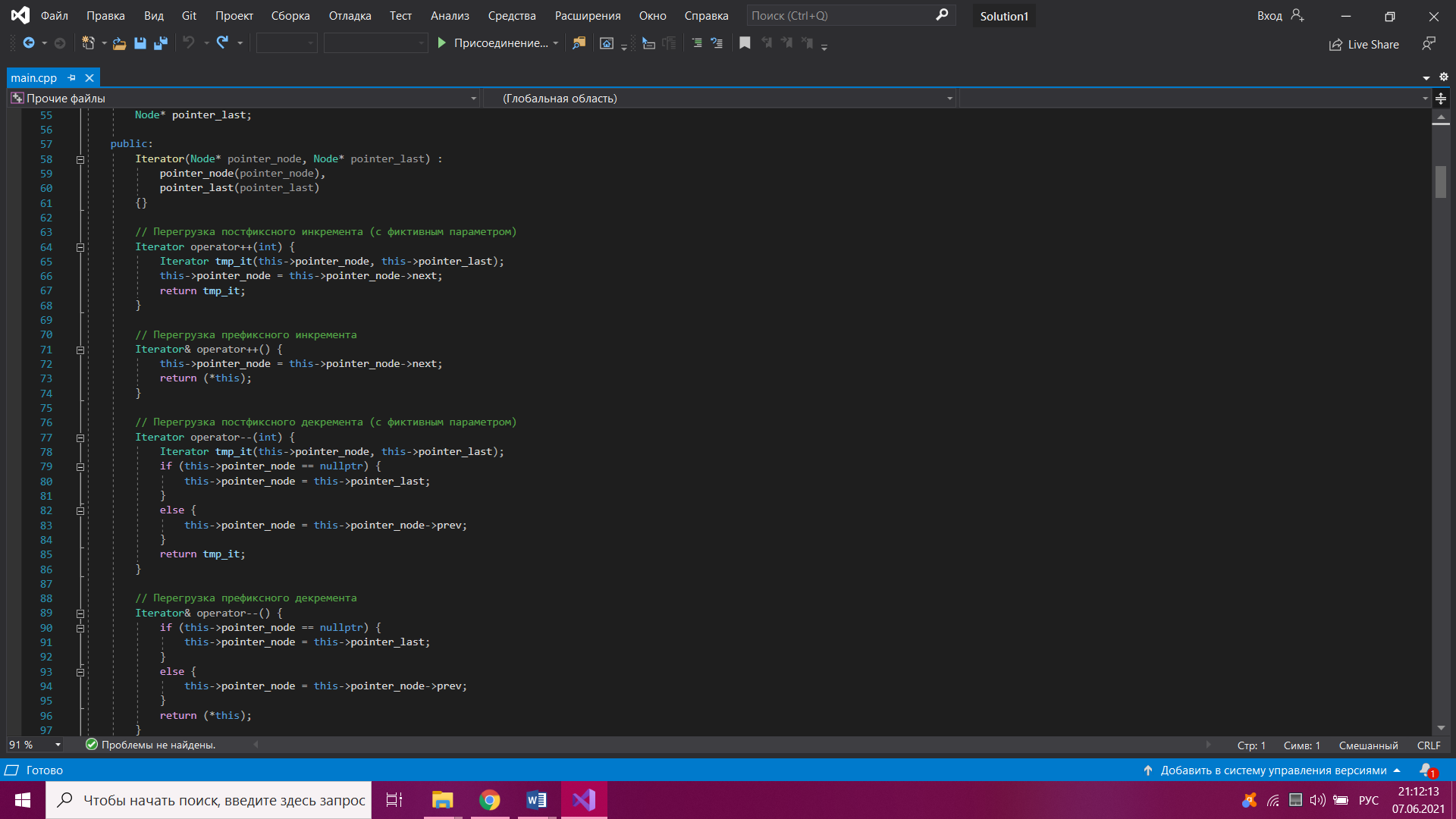
Cтандартное объявление итератора такого типа.

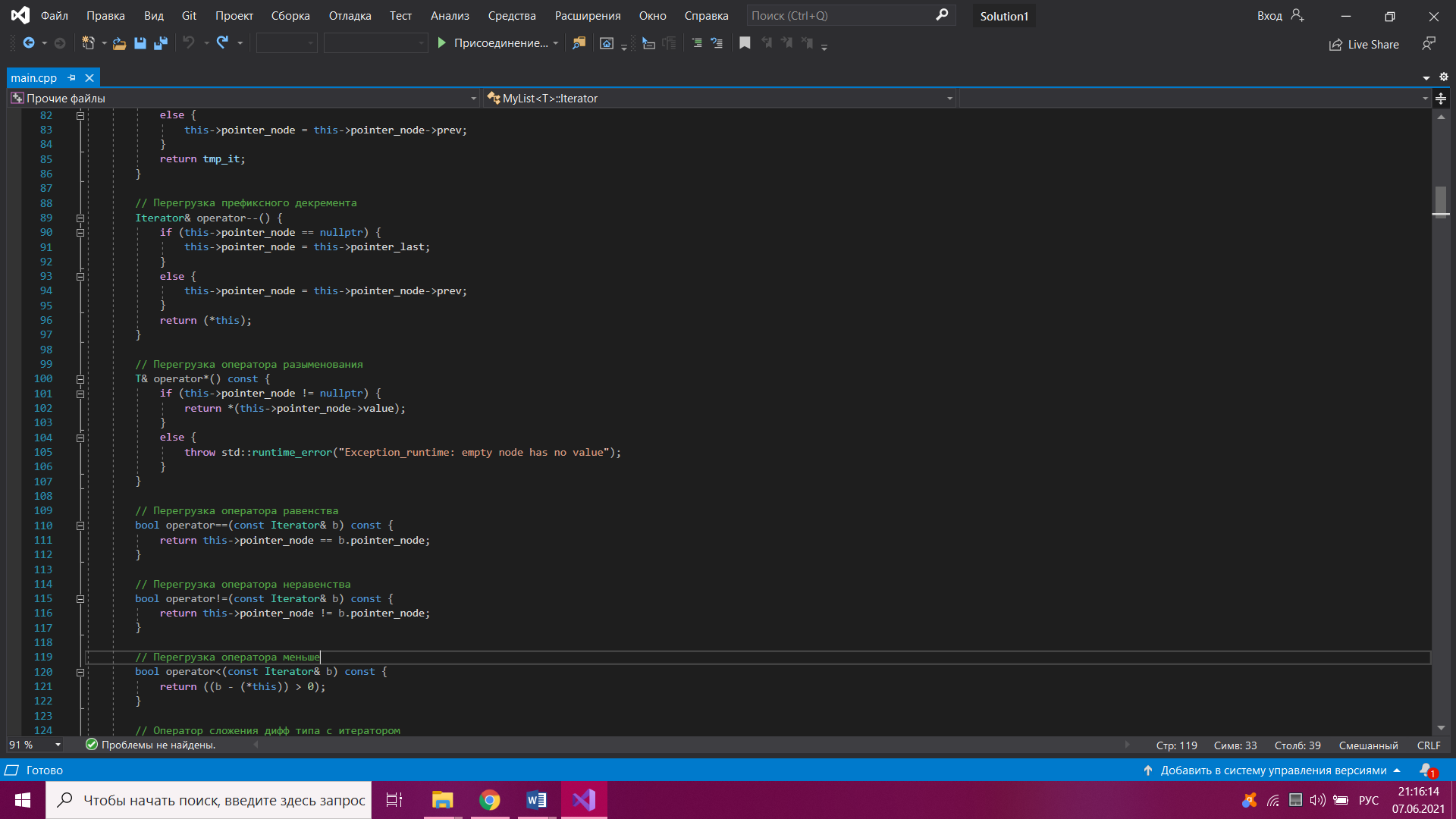


Конструктор итератора принимает 2 аргумента: указатель на текущий узел и указатель на последний узел структуры.

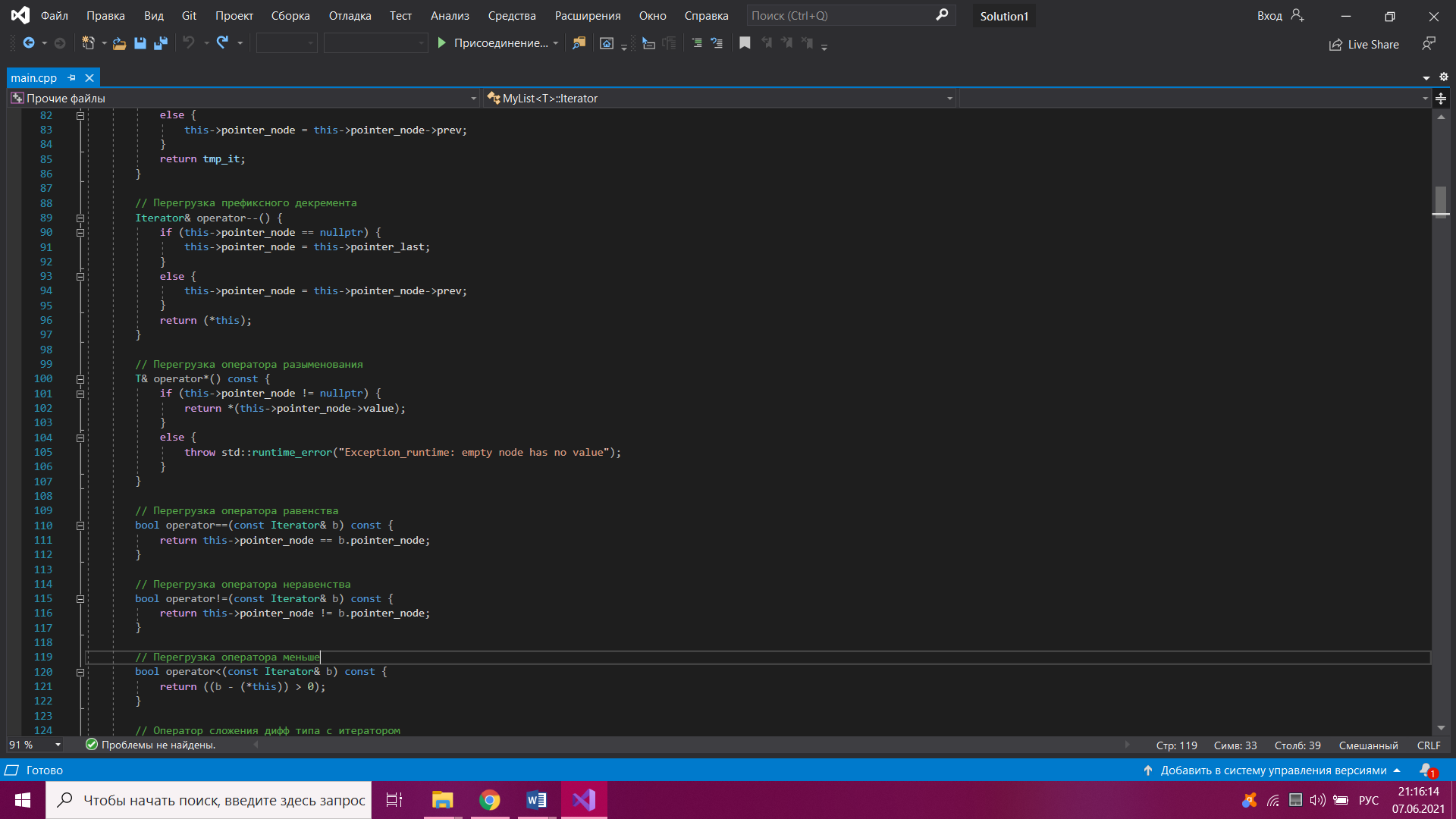


Далее рассматриваются операторы итератора.

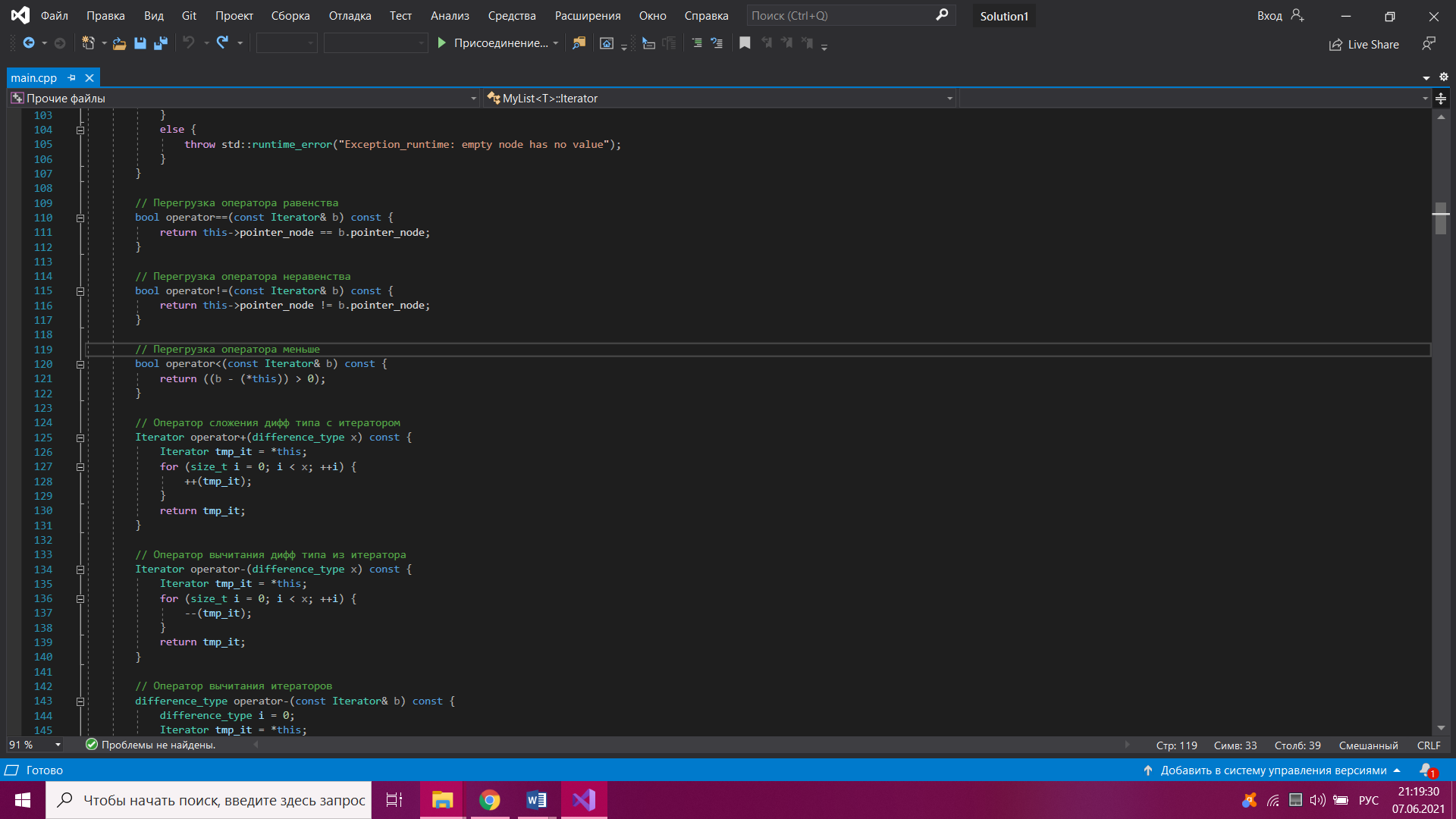


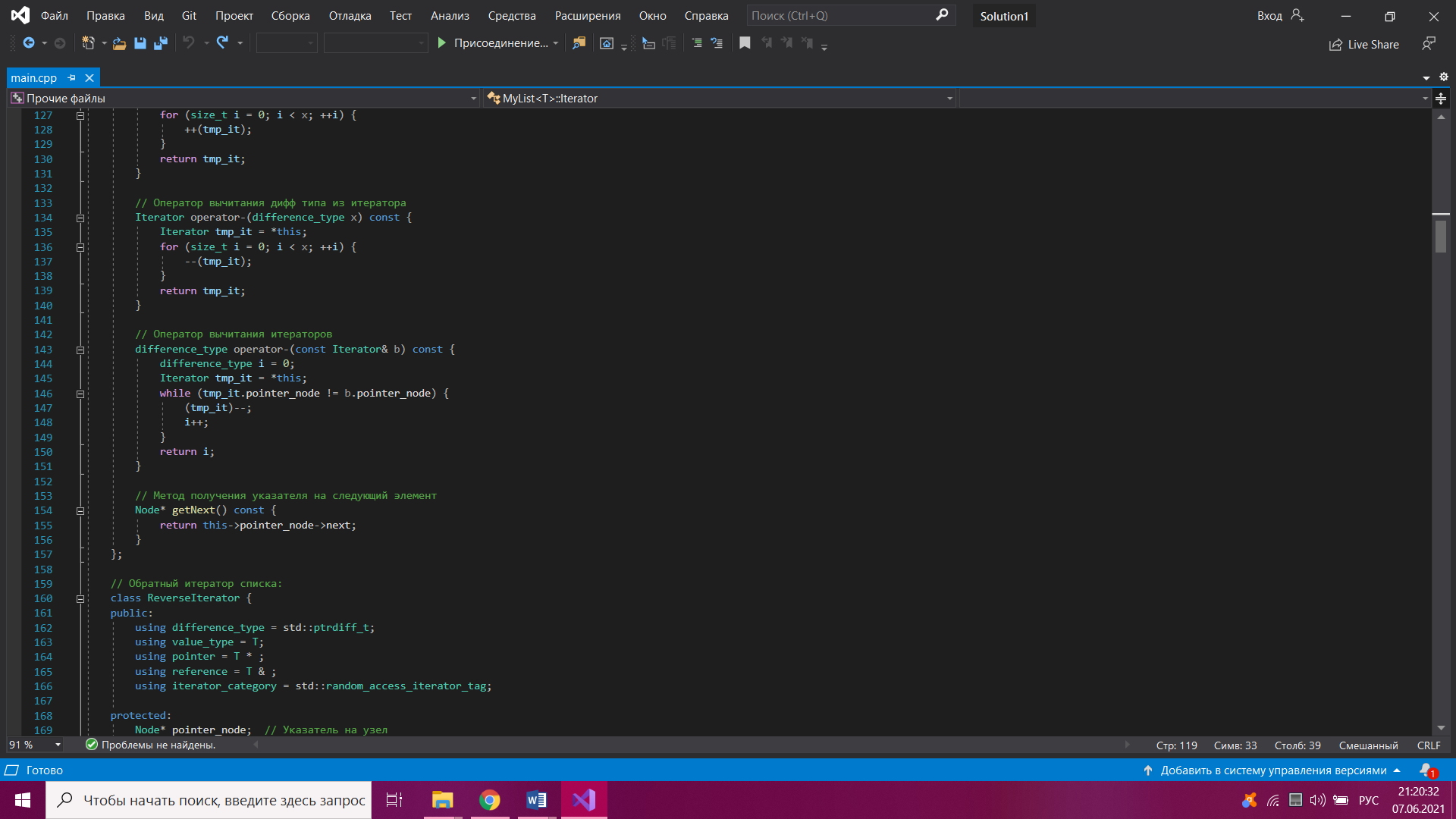


Операторы равенства и неравенства указывают ли они на одно значение в памяти



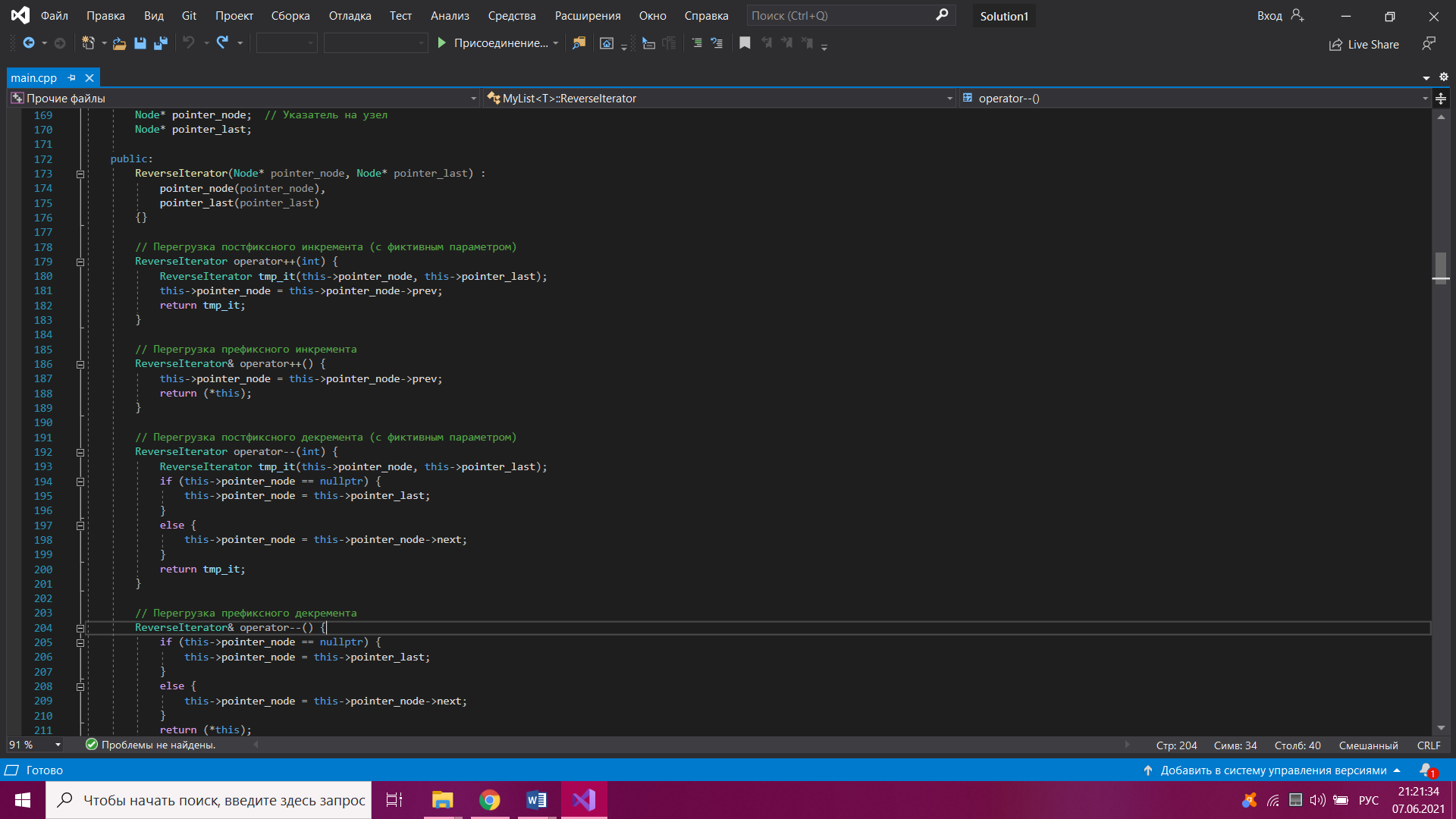
Оператор сравнения. Он cравнивает два узла вычитая один итератор из другого, и сравнивая ответ c нулем. Этот оператор используется в методе быстрой сортировки в функции sort.





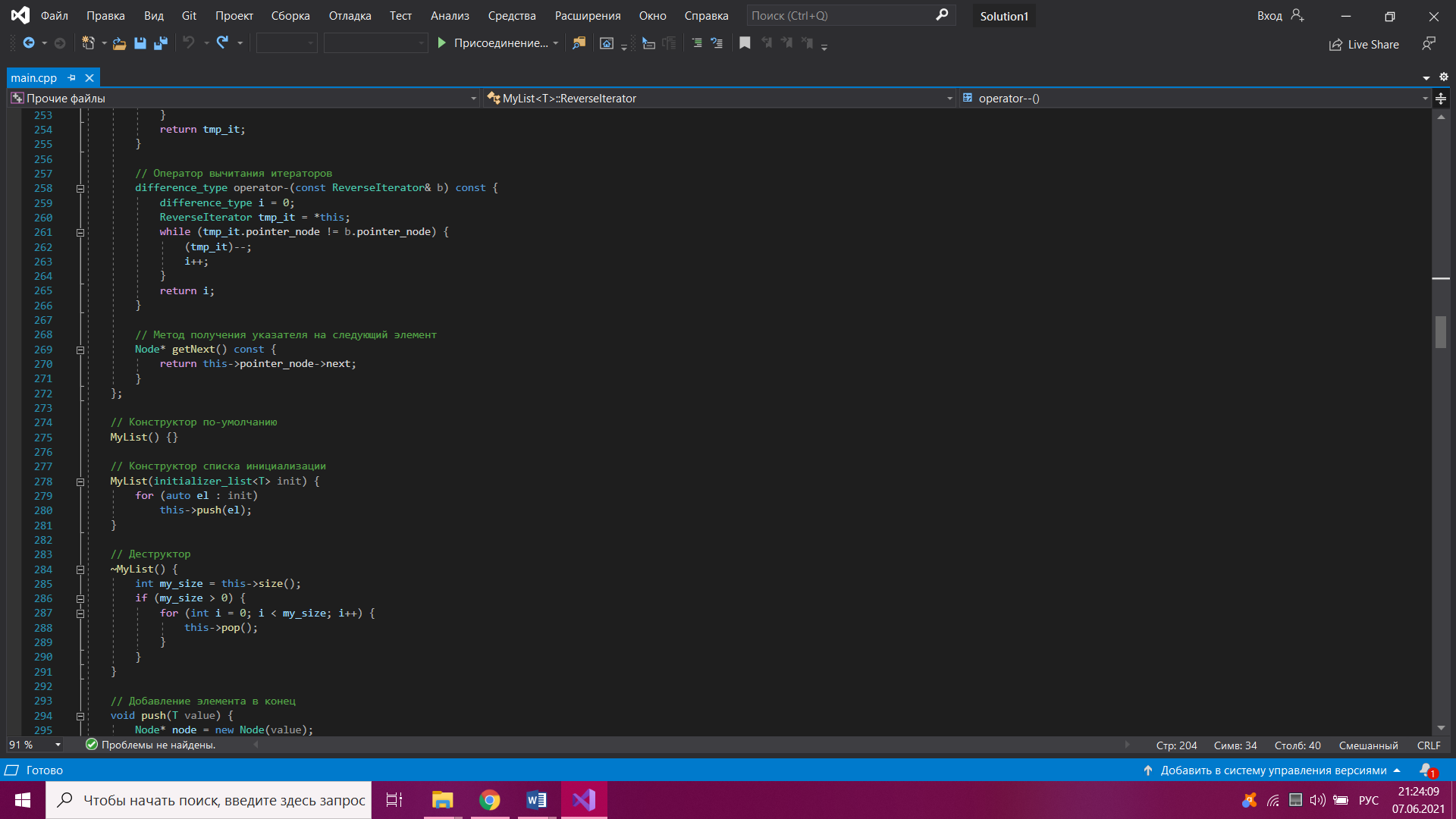
Оператор вычитания итераторов в положении между двумя узлами находит разницу и некое значение разностного типа возвращает, которое указывает cколько элементов может быть помещено между этими узлами.

Обратный итератор ReverseIterator полностью аналогичен обычному за исключением операторов инкремента и декремента



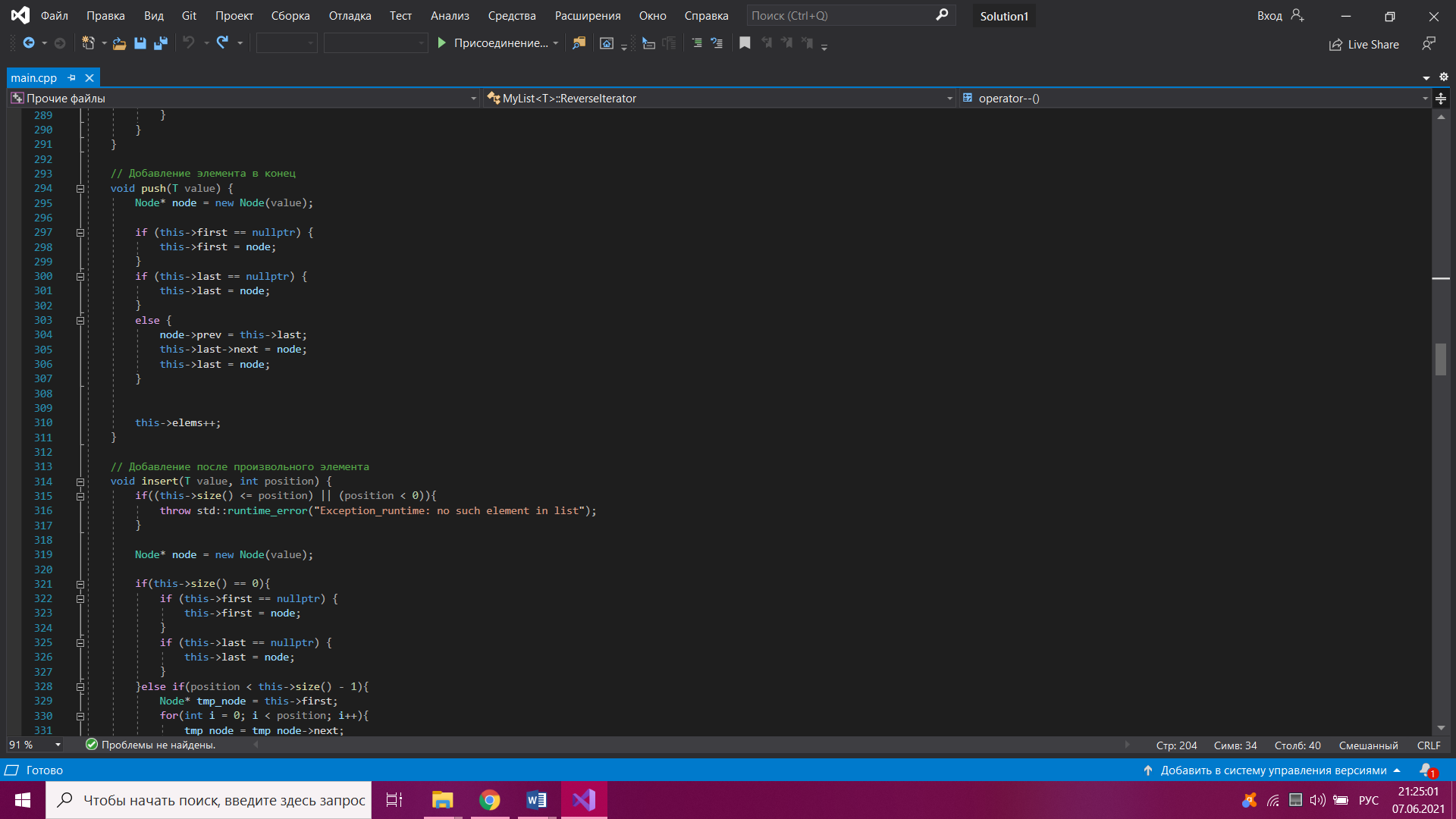
Происходит переход на следующий узел – условия работы итератора для rbegin, rend.

У класса списка имеются 2 конструктора и 1 деструктор( предотвращает утечку памяти):

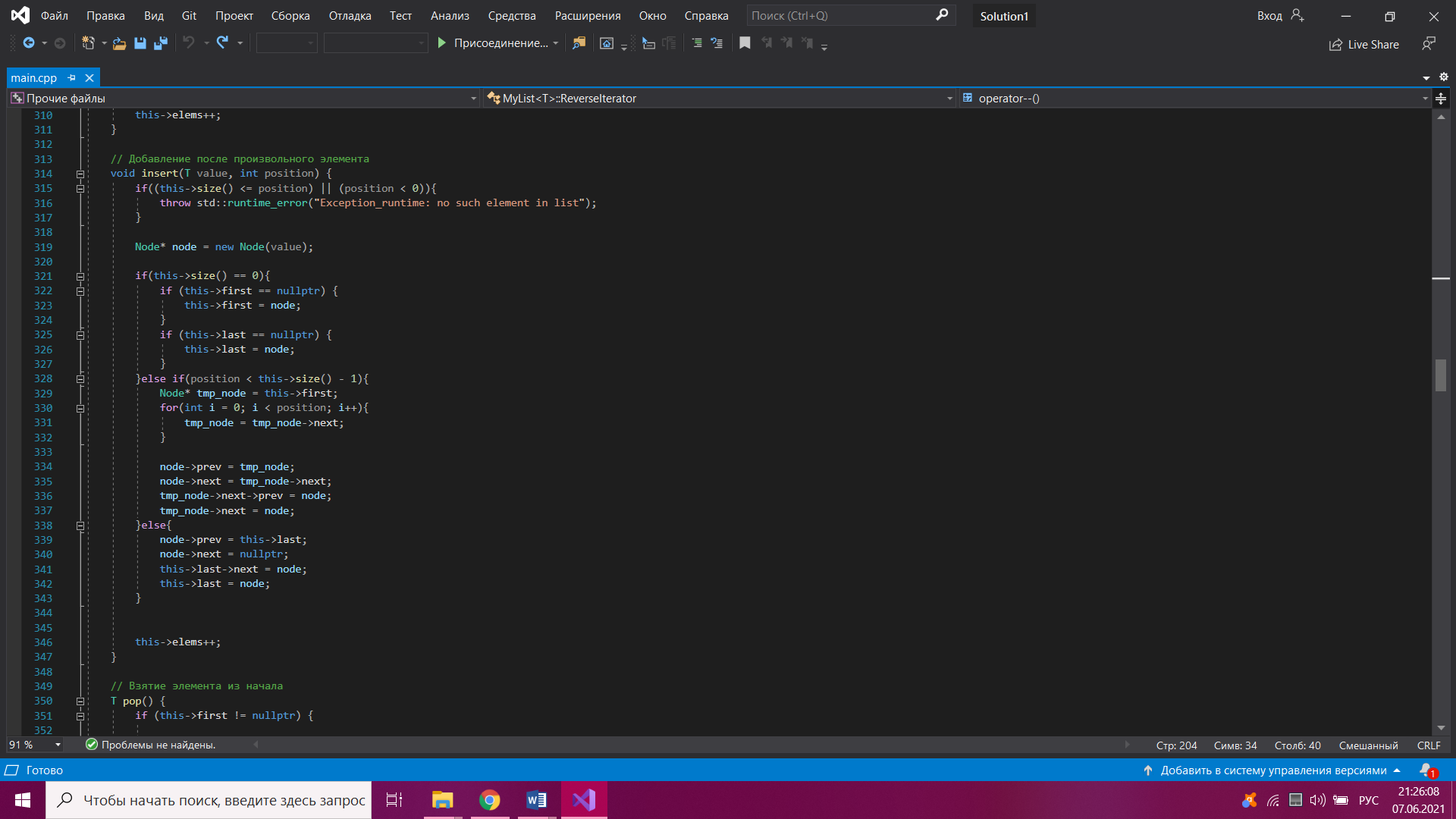


Инициализация возможна с использованием списка значений.

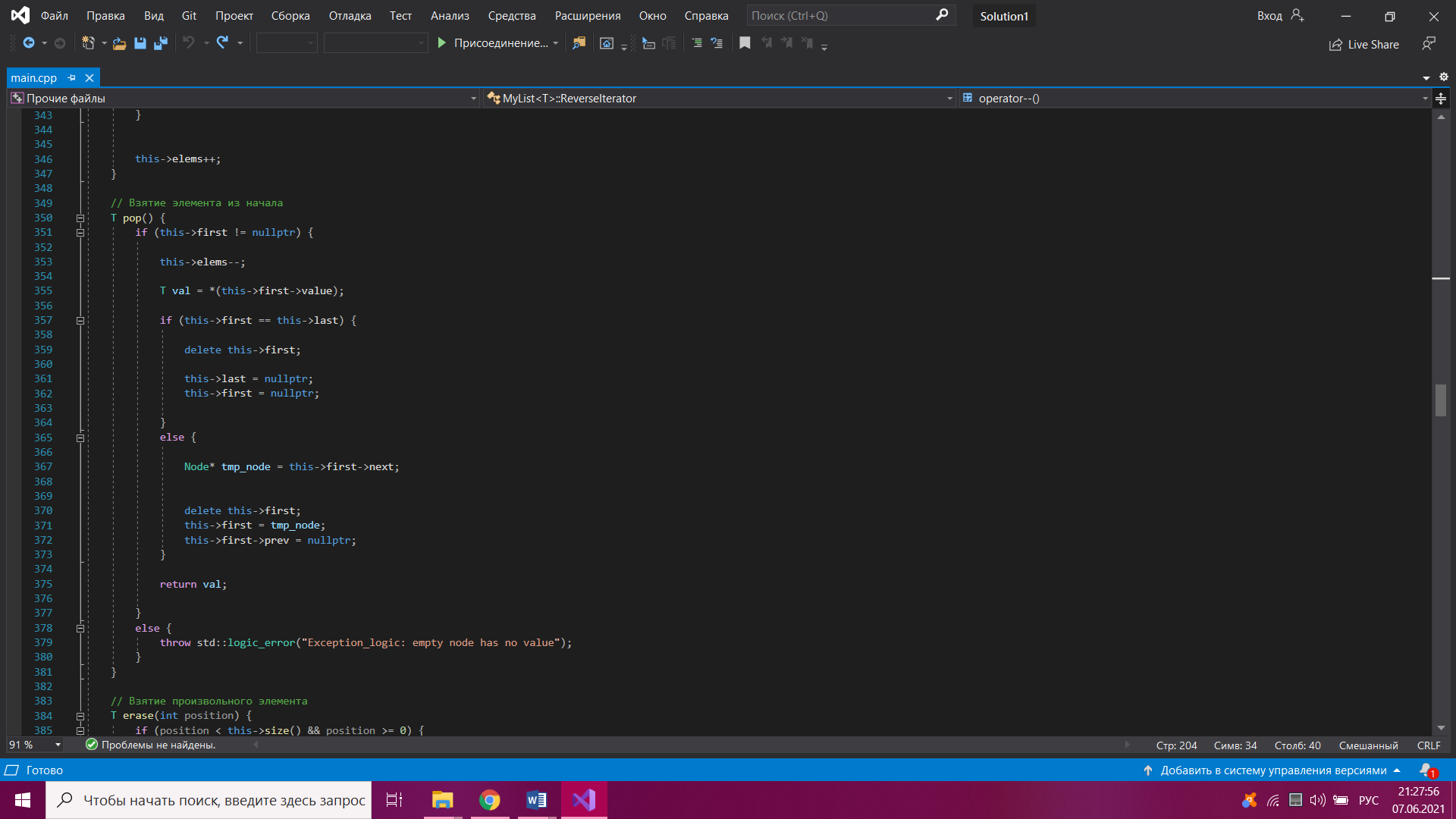
М. добавления в конец списка элементов реализуется созданием нового узла и переадресацией указателей на него:



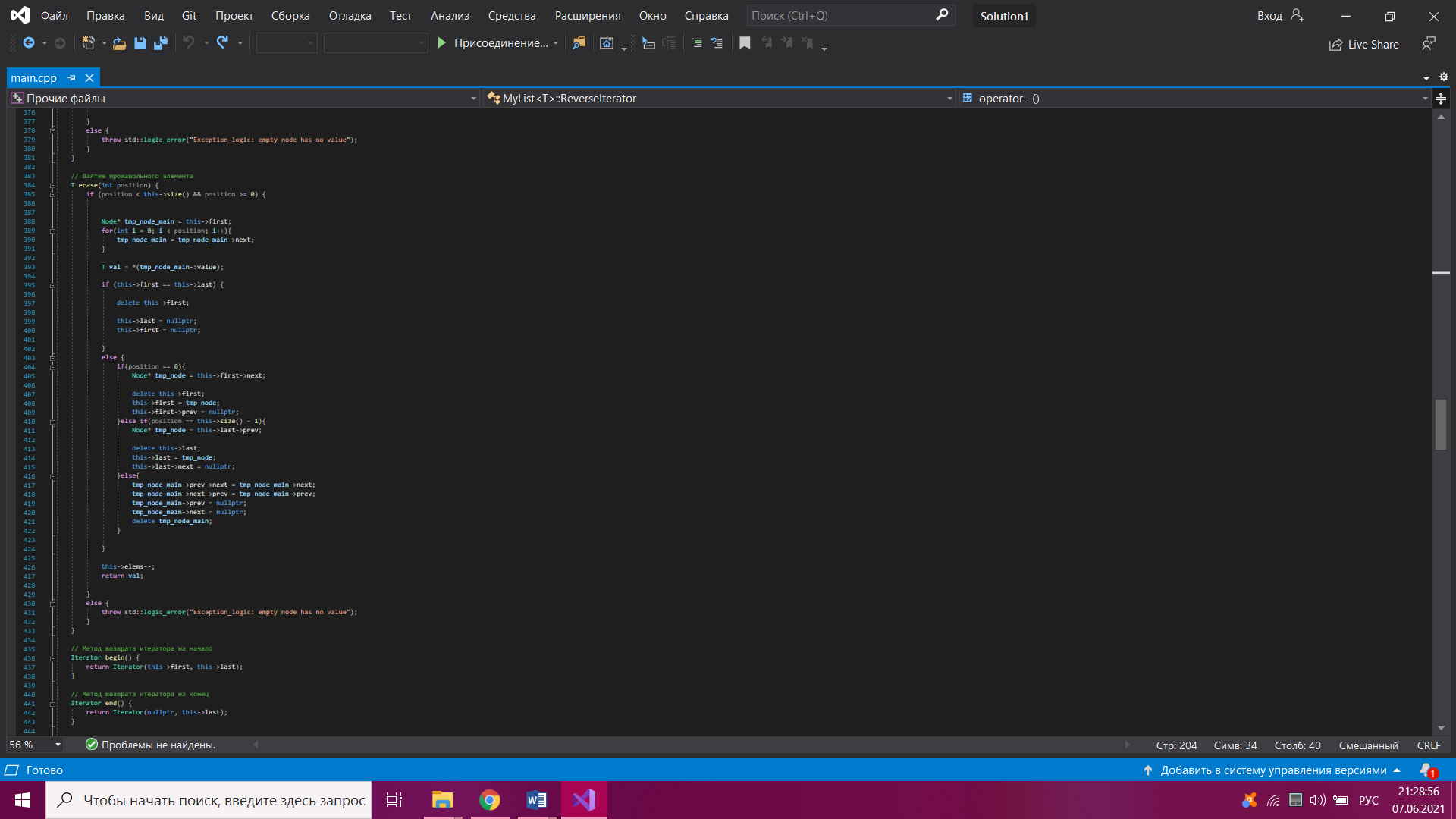
М. добавления элемента после определенного элемента работает аналогично, но вставляет элемент на определенную позицию. Метод принимает два параметра: новое значение и позиция, за которой надо вставить элемент с этим значением.

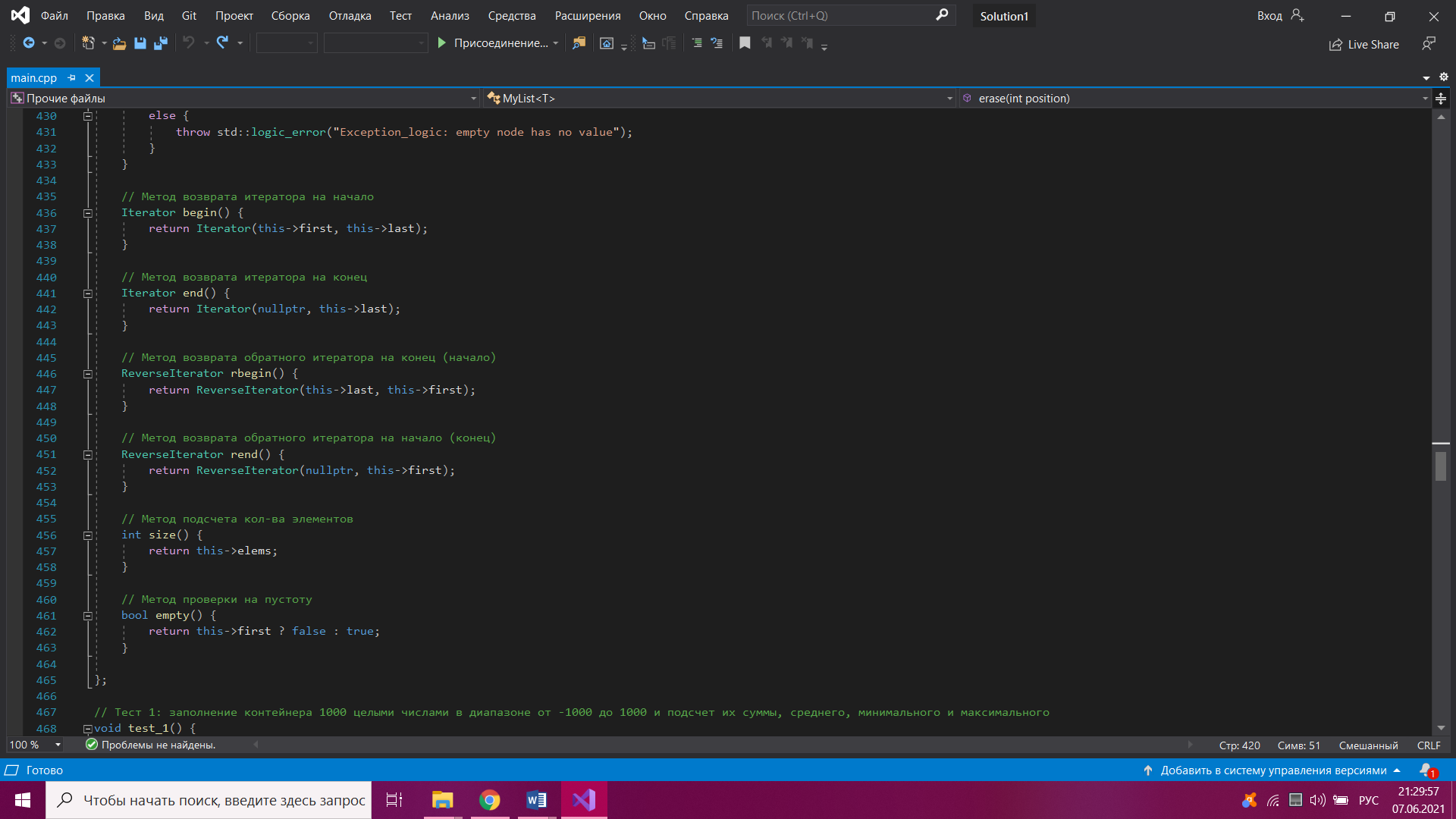


М. взятие из начала реализуется так, что значение элемента копируется, он сам удаляется, после чего происходит переадресация указателей с учетом измененной структуры, а само значение возвращается из метода:

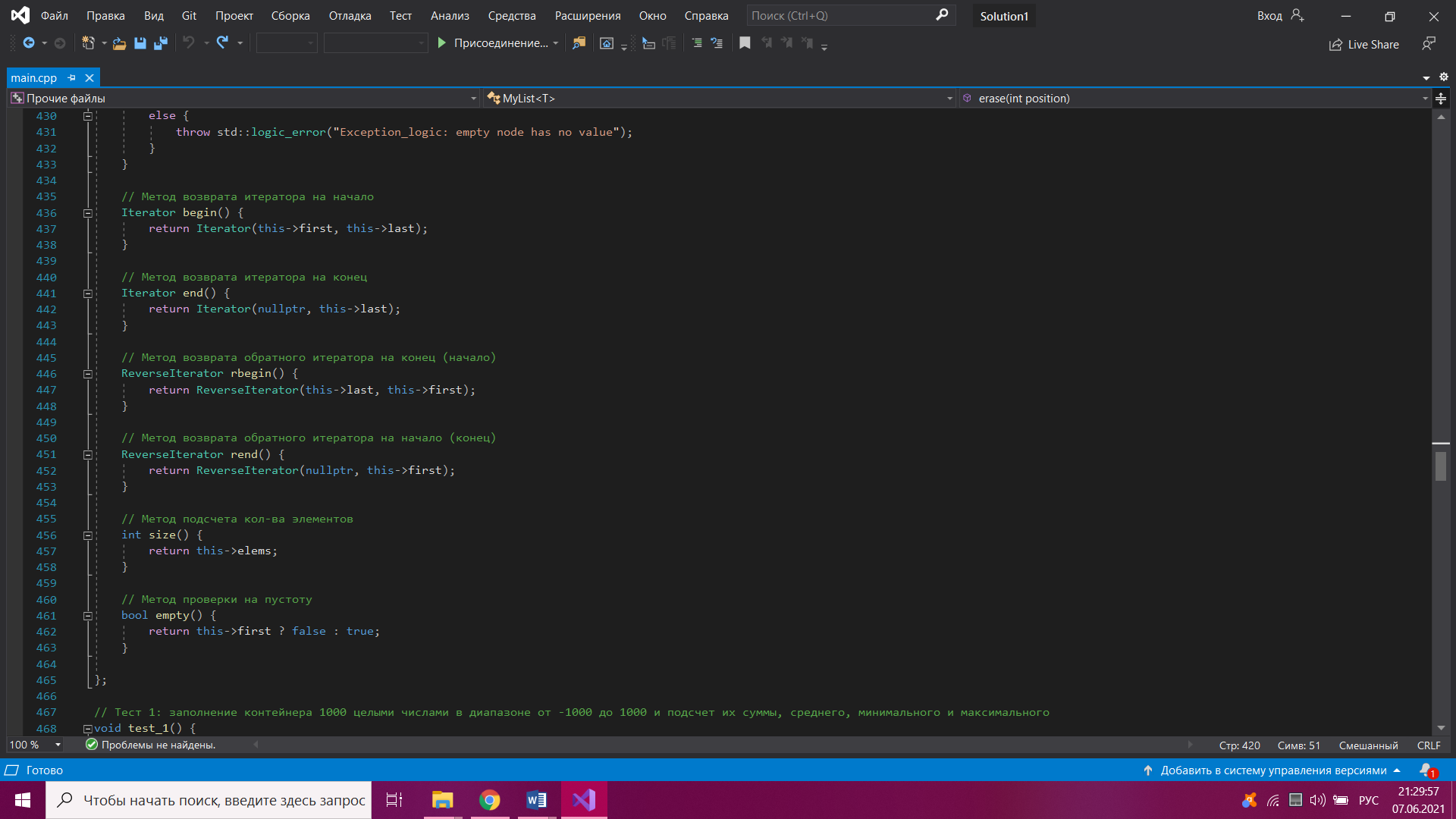


М. взятия определенного элемента принимает параметром позицию, элемент на которой надо удалить и вернуть значение.





М. end возвращает итератор с указателем на nullptr, который дает понять sort, что дальше элементов нет, после чего такой итератор способен вернуть последний элемент структуры благодаря реализации оператора декремента.



# Заключение.

В ходе работы реализована и разобрана структура двусвязного списка.

Было написано два итератора так как: прямой и обратный, а также создана была поддержка sort и for auto для написанной структуры.

Все тесты структуры прошли без ошибок в Visual Studio 2019.