Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Программирование сетевых приложений

Студент: Немкович А.В.

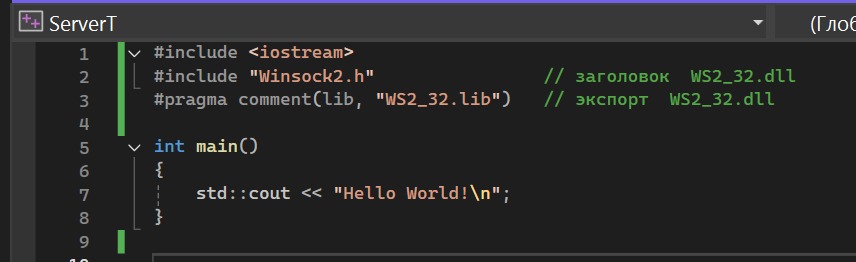
ФИТ 3 курс 1 группа

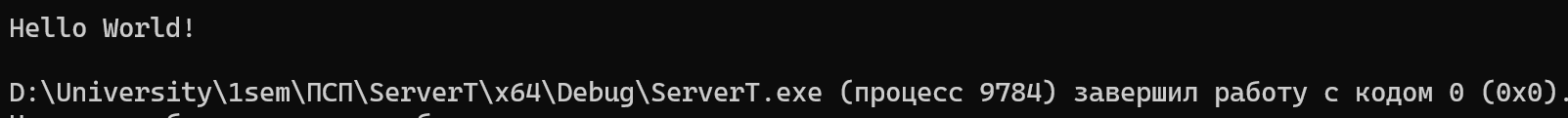
Преподаватель: Некрасова А.П.

Минск 2024

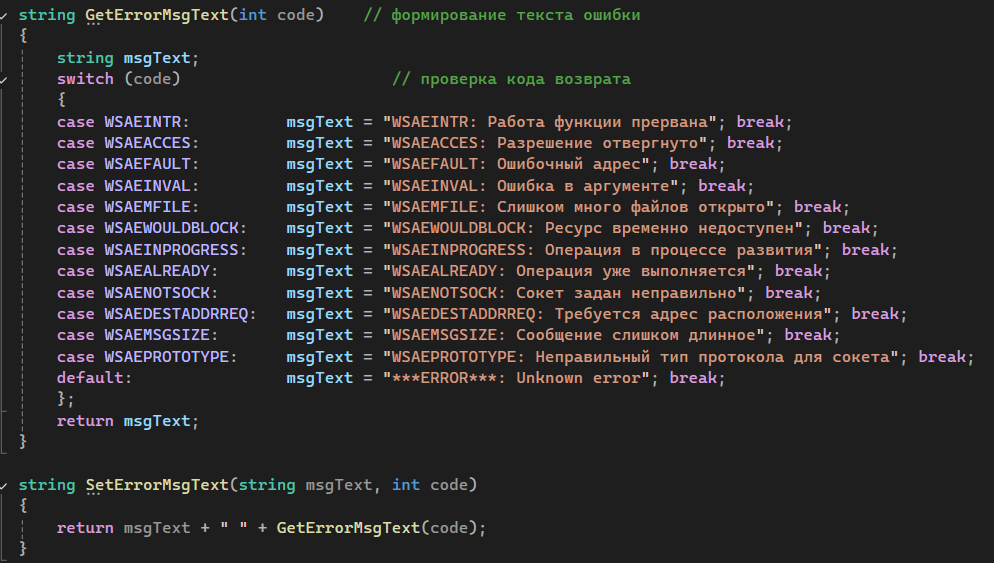
**Лабораторная работа №2**

**Задание 1.** Ознакомьтесь со схемой сервера, изображенной на рисунке 3.4.2 пособия. Создайте с помощью Visual Studio консольное приложение ServerT (наименование проекта), которое будет использовано для построения серверной части приложения (сервера). Включите необходимые директивы компилятора (указанные в разделе 3.2 пособия) для подключения динамической библиотеки **WS2\_32.LIB**. Откомпилируйте приложение, убедитесь в отсутствии ошибок.

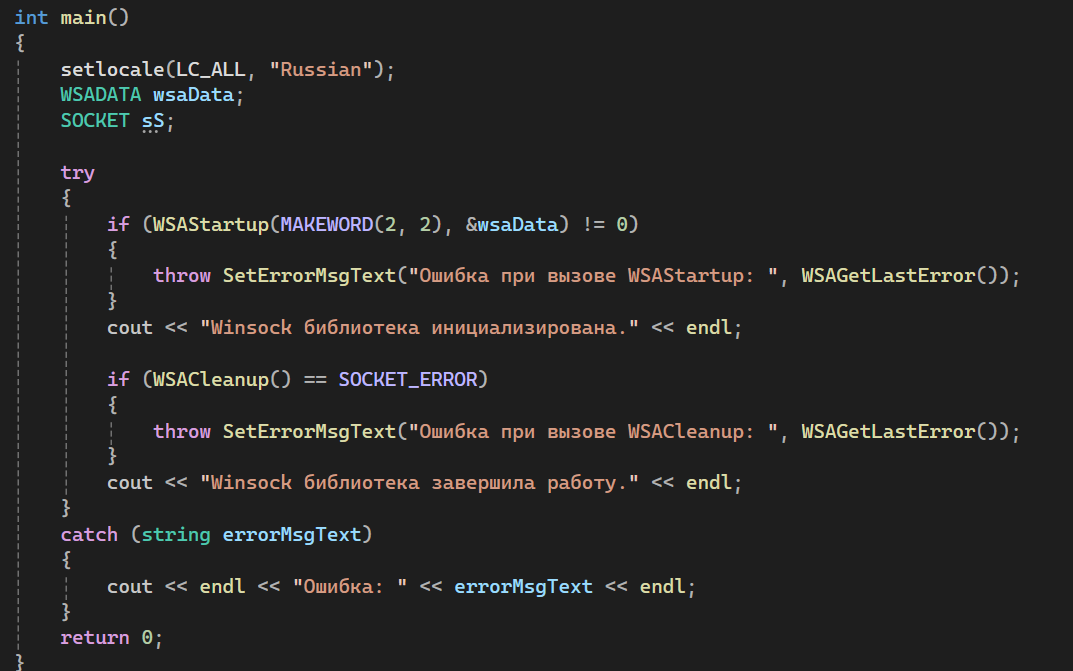




**Задание 2.** В рамках приложения ServerT, созданного в задании 1, разработайте о функцию **SetErrorMsgTex**t, предназначенную для обработки стандартных ошибок библиотеки WS2\_32.LIB. Предполагается, что функция SetErrorMsgText будет использоваться в операторе throw для генерации исключения при возникновении ошибок в функциях интерфейса Winsock2.

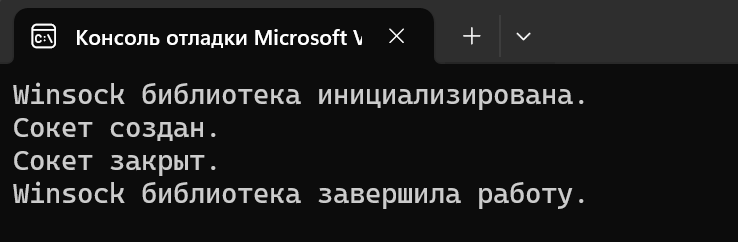


**Задание 3**. Доработайте приложение ServerT таким образом, чтобы оно только инициализировало библиотеку WS2\_32.LIB и завешало работу с этой библиотекой. Для этого используйте функции **WSAStartup** и **WSACleanup**, описанные в разделах 3.5 и 3.6 пособия. Обработку ошибок осуществите с помощью конструкции try-catch и функции SetErrorMsgText, разработанной в задании 2.

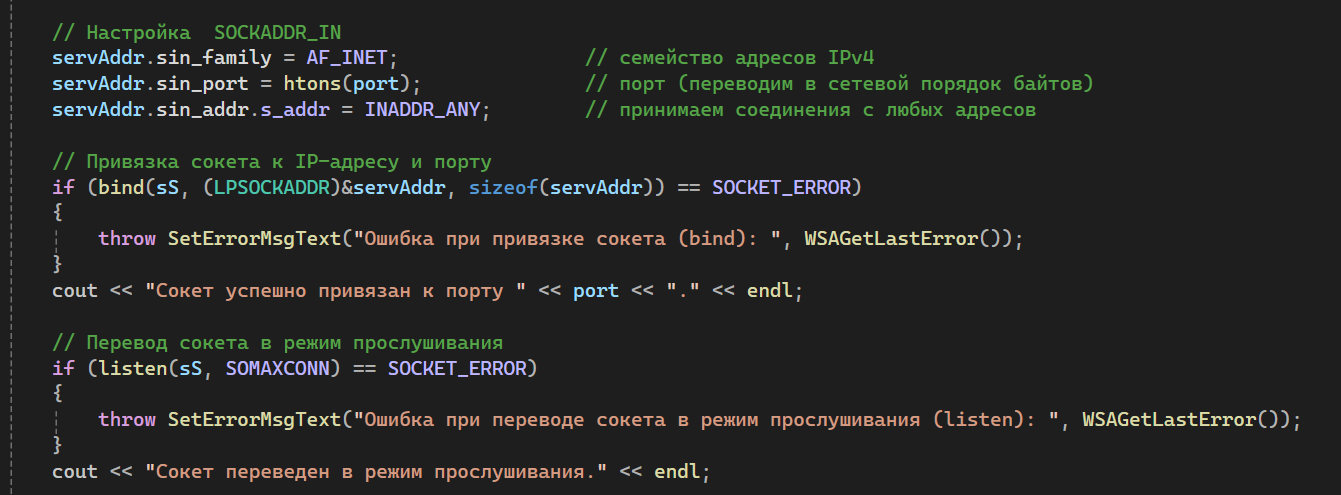


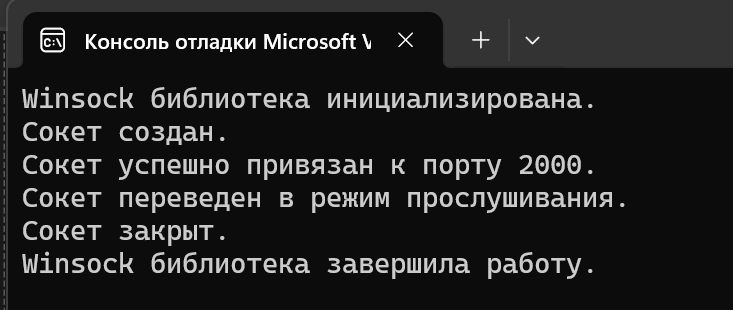
**Задание 4.**  Доработайте приложение ServerT таким образом, чтобы оно создавало и закрывало сокет, предназначенный для ориентированного на поток соединения. Для этого используйте функции socket и closesocket, описанные в разделе 3.8. Обратите внимание на параметр type функции socket, указывающий тип соединения. Воспользуйтесь примером из раздела 3.7 пособия. Убедитесь в работоспособности приложения.



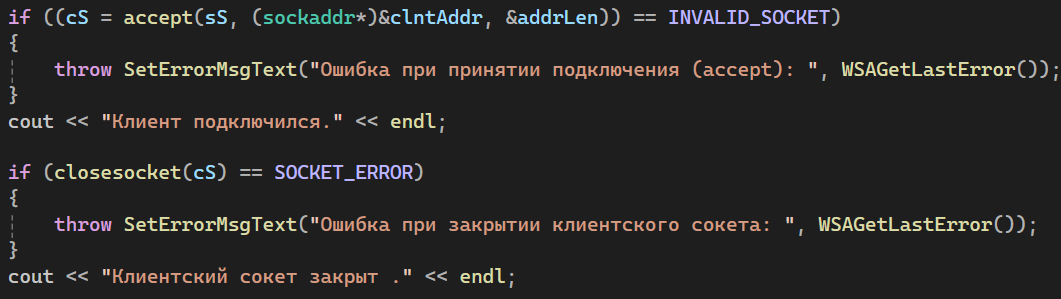


**Задание 5.** Добавьте в приложение ServerT вызов функций **bind** и **listen** для установки параметров сокета и перевода его в режим прослушивания. Функция bind описана в разделе 3.8, а функция listen в разделе 3.9 пособия. Используйте порт 2000, в качестве параметра сокета. Установка параметров сокета осуществляется с помощью структуры SOCKADDR\_IN. Описание этой структуры приводится в разделе 3.8 пособия. Сверьте схему полученной программы со схемой сервера, изображенной на рисунке 3.4.2. Выполните приложение, убедитесь в его работоспособности.



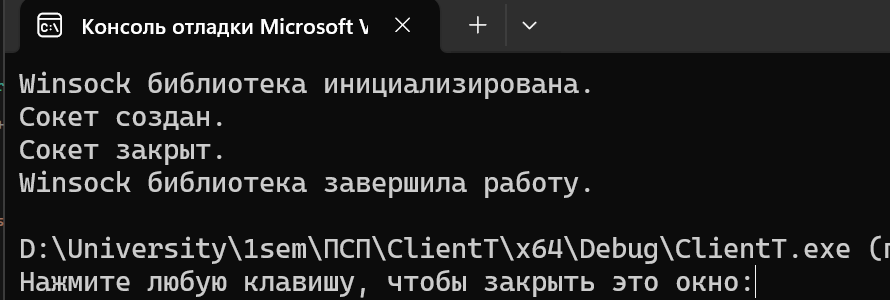


**Задание 6.** Добавьте в приложение ServerT вызов функции **accept**, описание которой приводится в разделе 3.10 пособия. Следует обратить внимание на: 1) успешным результатом работы функции accept является новый сокет; 2) первым параметром функции accept является уже созданный ранее сокет; 3) второй параметр функции accept – указатель на структуру SOCKADDR\_IN (не надо ее путать с уже применяемой выше), предназначенную для приема параметров, подключившегося сокета со стороны клиента сокета. Запустите приложение в режиме отладки (Debug) и убедитесь, что после выполнения функции accept, программа переходит в режим ожидания (зависает). Завершите приложение. Сохраните программу ServerT для дальнейшего применения.

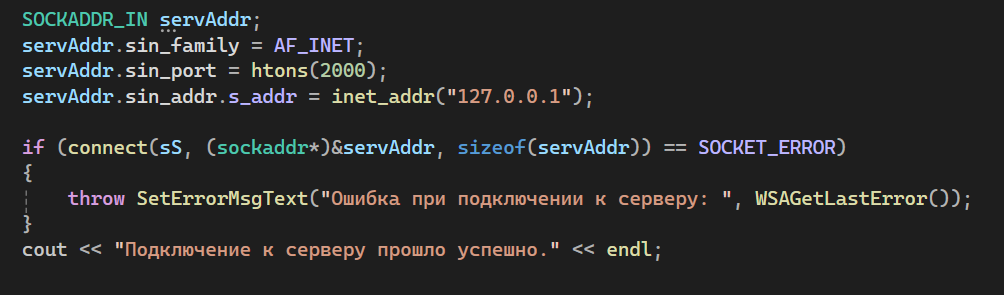


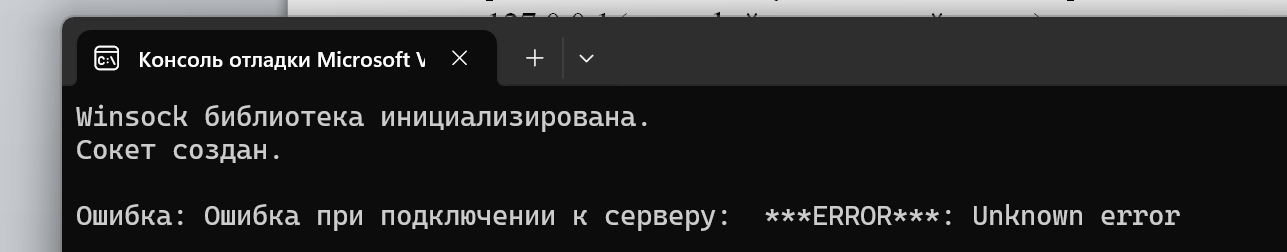
**Задание 7.** Ознакомьтесь со схемой клиента, изображенной на рисунке 3.4.2 пособия. Создайте с помощью Visual Studio новое консольное приложение ClientT (наименование проекта), которое будет использовано для построения клиентской части приложения (клиента). Повторите все те же действия для этого приложения, которые были сделаны в заданиях 2-4. Убедитесь в работоспособности приложения ClientT.



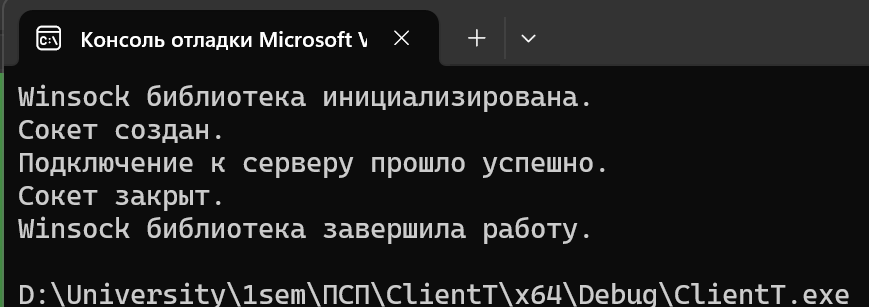


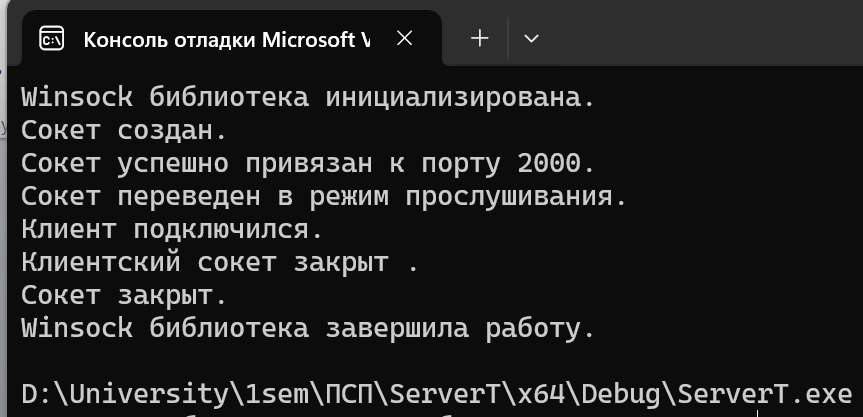
**Задание 8.** Добавьте в приложение ClientT, вызов функции connect. Описание функции и примера ее использования приводится в разделе 3.10 пособия. Следует обратить внимание на следующее: 1) параметры сокета сервера устанавливаются в структуре SOCKADDR\_IN; 2) для номера порта необходимо установить значение 2000 (такой же номер, что установлен при параметризации сокета сервера а задании 5); 3) для установки номера порта используются специальные функции, описание которых приводится в разделе 3.8. Используйте в качестве IP-адреса собственный адрес компьютера 127.0.0.1 (интерфейс внутренней петли) – это даст возможность отладки приложения на одном компьютере. Запустите приложение на выполнение. Убедитесь, что функция connect завершилась с ошибкой и обработка ошибок осуществляется корректно. Найдите полученный код ошибки в таблице 3.3.1 пособия и проанализируйте его.



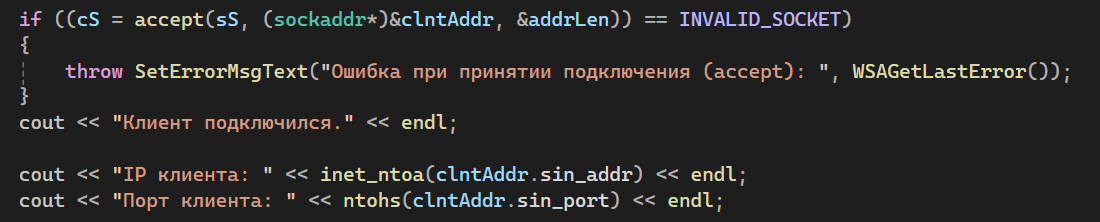


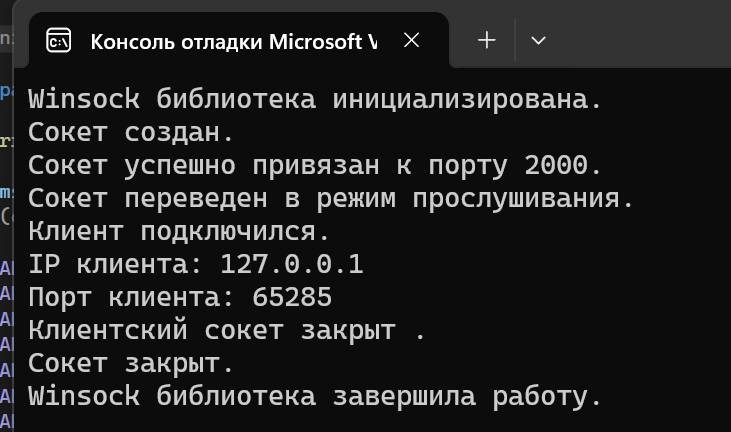
**Задание 9.** Запустите на выполнение приложение ServerT и убедитесь, что оно приостановилось на вызове функции accept. Запустите на выполнение на этом же компьютере (используется интерфейс внутренней петли) приложение ClientT. Убедитесь, что сервер ServerT, вышел из состояния ожидания, а клиент ClientT завершился без ошибок.



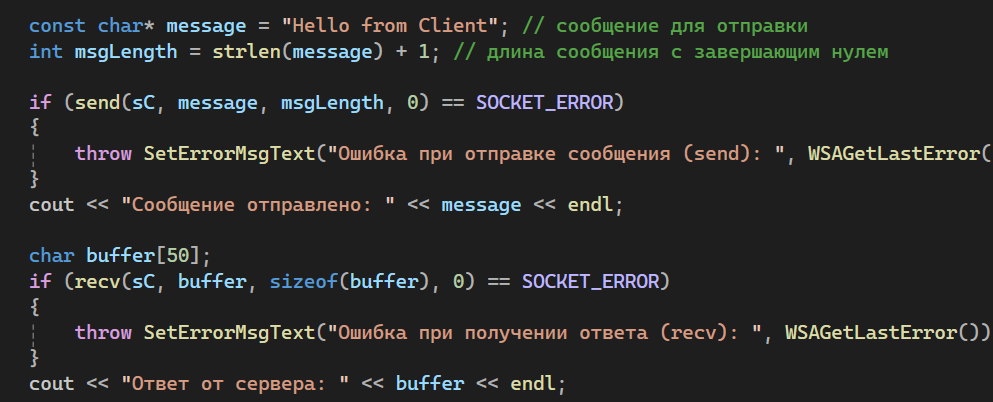


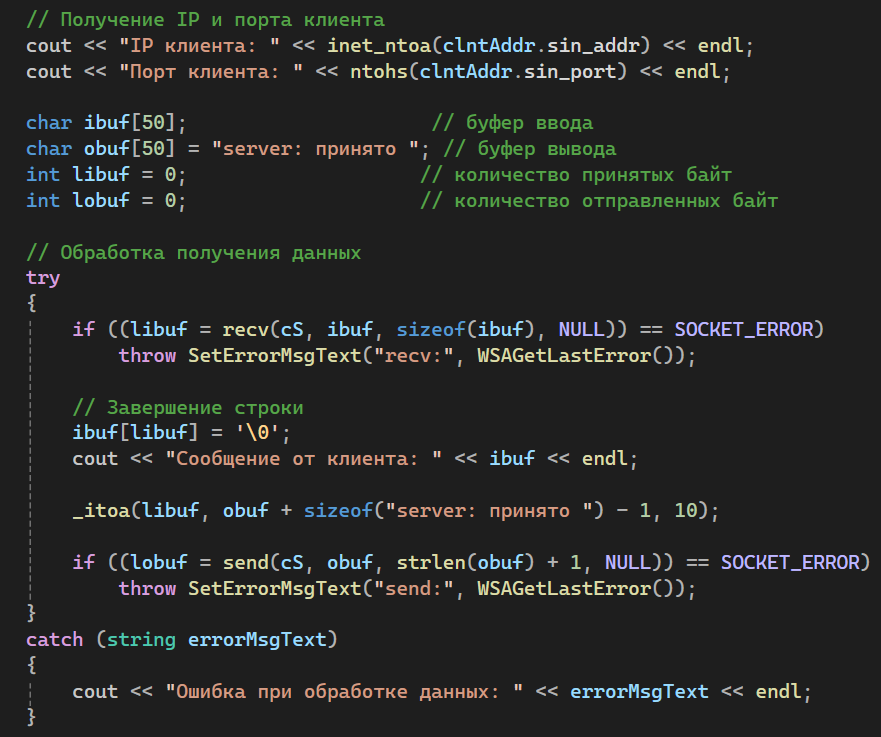
**Задание 10.** Доработайте программу сервера ServerT таким образом, чтобы после подсоединения клиента на экран консоли ServerT выводился IP-адрес и порт, подсоединившегося клиента. Необходимые значения находятся в структуре SOCKADDR\_IN, которая заполняется функцией accept. Используйте функции htons и inet\_ntoa, описанные в разделе 3.8. Убедитесь в работоспособности распределенного приложения ClientT-ServеrT.

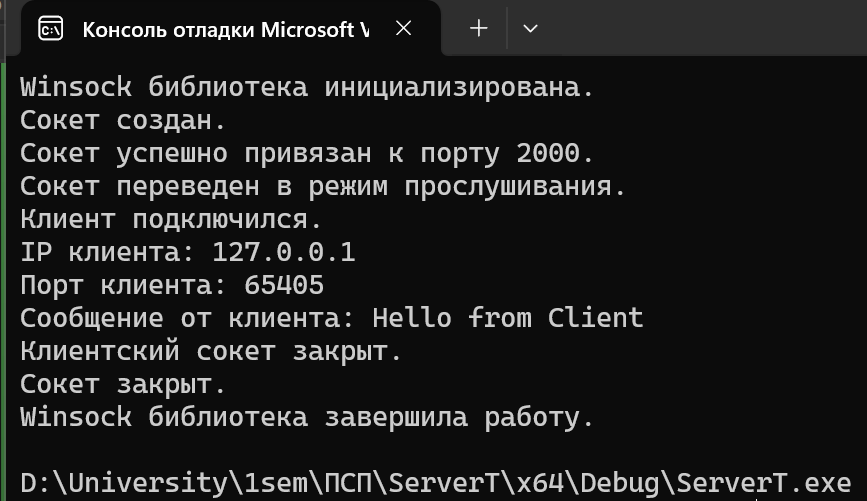




**Задание 11.** Добавьте в программу сервера ServerT вызов функции recv, а в программу клиента вызов функции send. Описание этих функций приводится в разделе 3.11 пособия. Перешлите текст Hello from Client от клиента серверу. Выведите полученный сервером текст на экран консоли. Обратите внимание на то, что команда recv в программе сервера использует сокет, созданный функцией accept, а не созданный ранее с помощью функции socket.

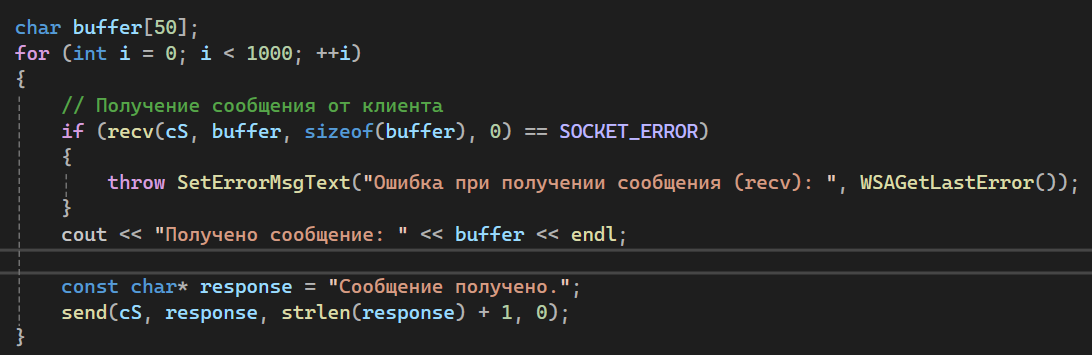


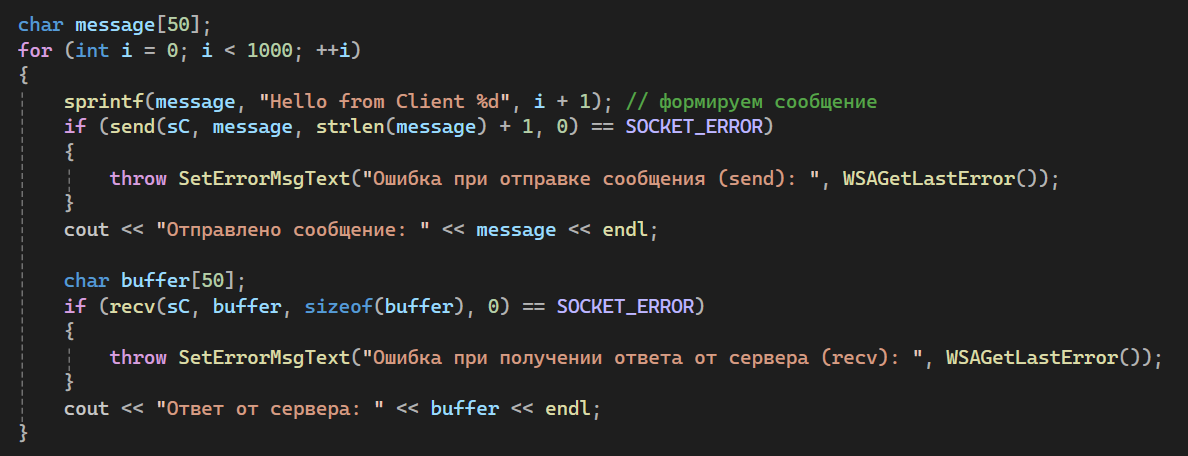


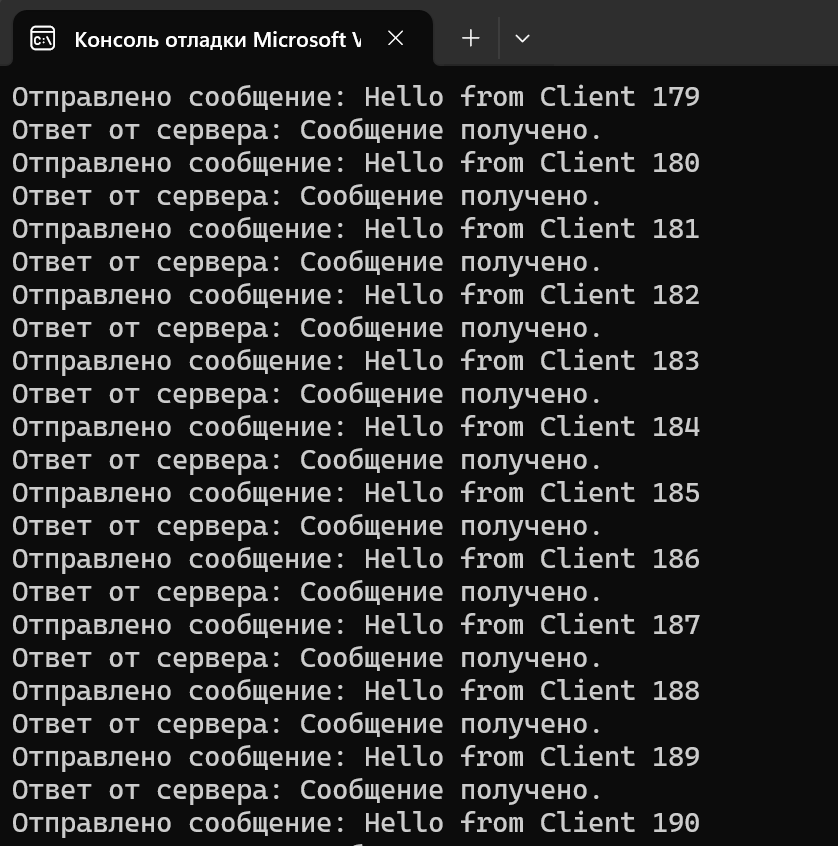


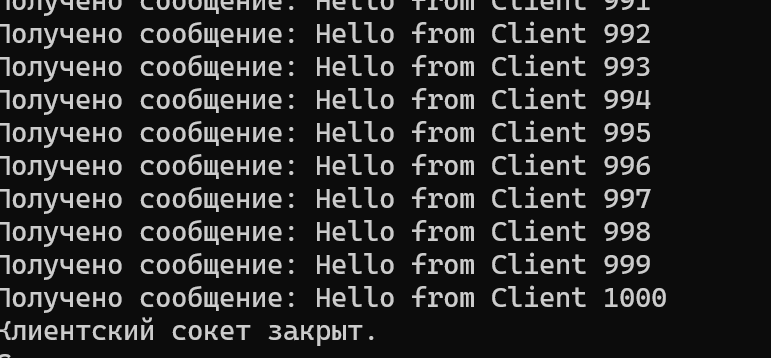
**Задание 12**. Установите программу ServerT на другой компьютер локальной сети, а в программу ClientT внесите необходимые изменения, позволяющие ей установить связь с сервером. Убедитесь в работоспособности распределенного в локальной сети приложения ClientT-ServеrT.

**Задание 13.** Внесите изменения в программы ClientT и ServerT, позволяющие 1000 раз передать сообщение типа Hello from Client xxx (xxx – номер сообщения) от клиента серверу. Убедитесь в работоспособности в сети.

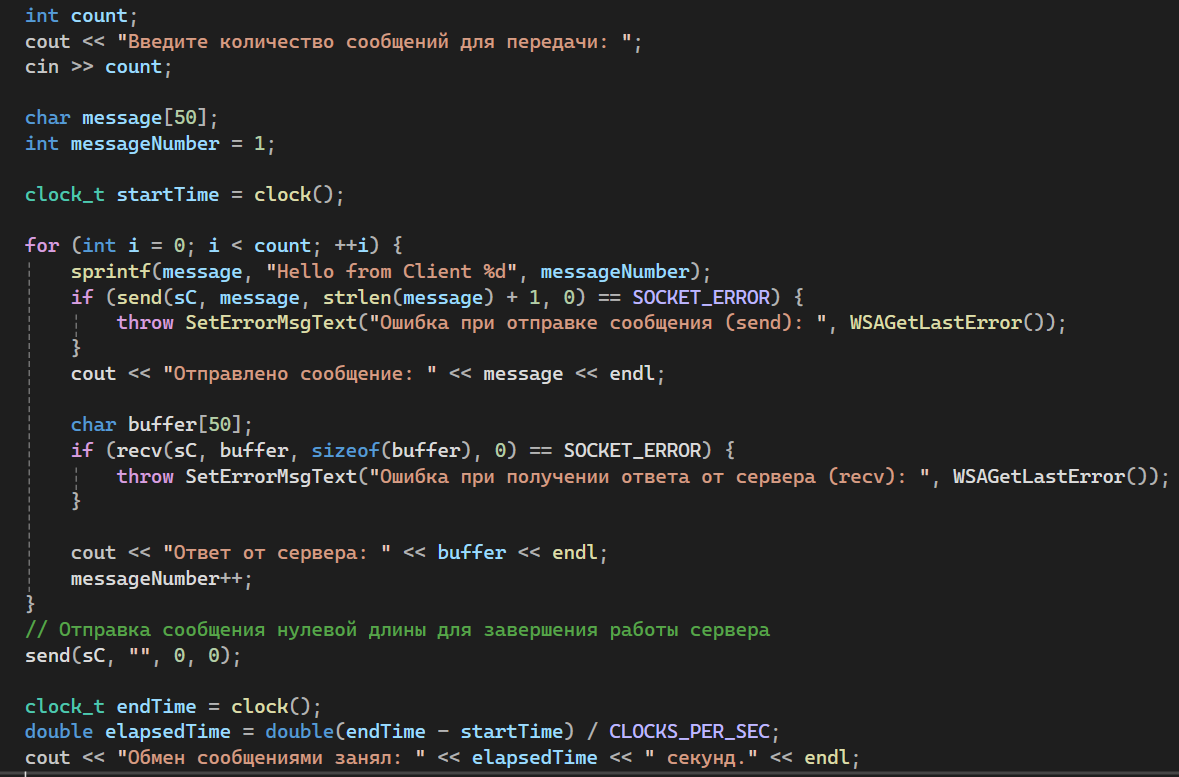


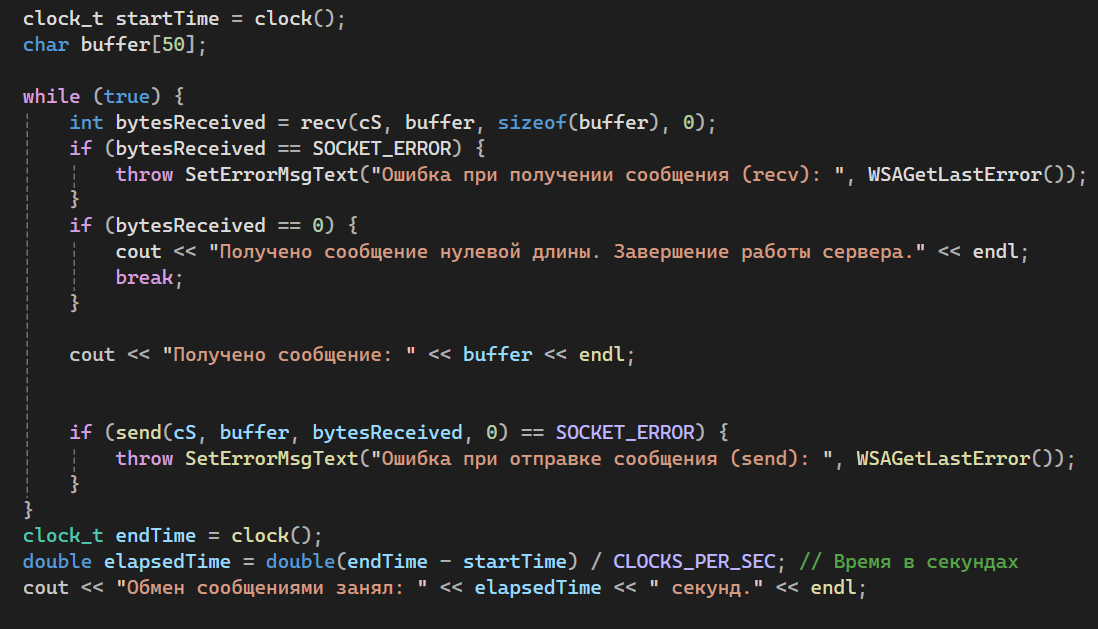


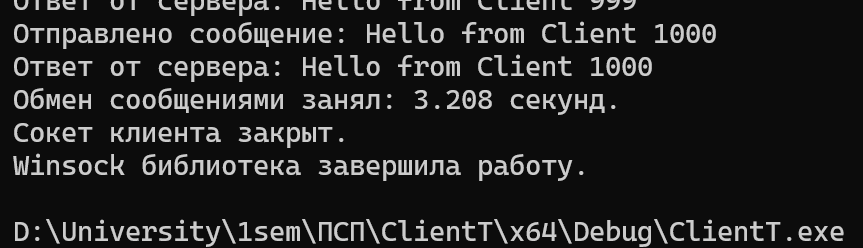


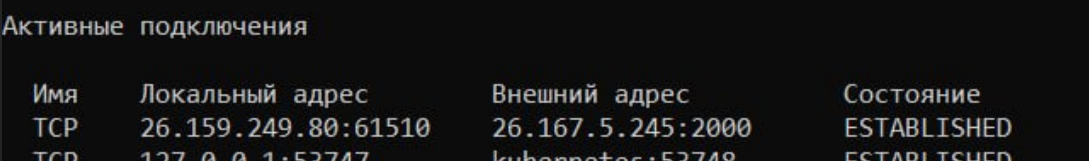


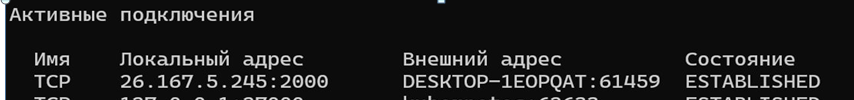
**Задание 14.** Доработайте программы ClientT и ServerT таким образом, чтобы программа ServerT принимала последовательность сообщений вида Hello from Client xxx от программы ClientT и отправляла их без изменения обратно программе ClientT. Программа ClientT, получив вернувшееся сообщение, должна увеличить в нем счетчик xxx на единицу и вновь направить в адрес ServerT. Количество передаваемых сообщений введите через консоль программы ClientT. Условием окончания работы для программы ServerT является получение сообщения нулевой длины. Оцените время обмена 1000 сообщениями (с помощью функций clock) между ClientT и ServerT через локальную сеть. Запустите утилиту netstat на компьютерах клиента и сервера, проанализируйте отчет и найдите информацию о приложении ClientT- ServerT.











Listening – server

Established - client

**Задание 15.** Доработайте программу ServerT таким образом, чтобы отключения клиента она могла снова установить соединение с другим клиентом и продолжила свою работу.

