**Практическая работа № 3. Обмен данными без установки**

**соединения**

**7.4.1. Цель и задачи работы**

Основнойцелью практической работы является приобретение навыков разработки простейшего распределенного приложения архитектуры клиент-сервер, осуществляющего обмен данными в локальной сети через Windows Sockets без установки соединения (с помощью UDP-сообщений).

Результатом практической работы является разработанное распределенное приложение со схемой взаимодействия процессов, описанной в разделе 3.4 и изображенной на рисунке 3.4.1 пособия.

**7.4.2. Теоретические сведения**

Теоретические сведения необходимые для выполнения практической работы изложены в разделах 3.2-3.12, 3.14 пособия.

**7.4.3. Разработка серверной части распределенного приложения**

**Задание 1**. Ознакомьтесь со схемой взаимодействия процессов без установки соединения в распределенном приложении, приведенной разделе 3.4 пособия (рисунок 3.4.1). Определите основные отличия этой схемы от схемы взаимодействия процессов с установкой соединения. Разработайте программу **ServerU**, реализующую блоки 1, 2 и 5 схемы сервера, изображенной на рисунке 3.4.1. Подключите функции обработки ошибок, разработанные в практической работе № 2 (с применением команд структурной обработки ошибок **try-throw-catch**). Обратите внимание 1) на параметр **type** функции **socket**; 2) на отсутствие функций **listen** и **accept**, которые применялись в приложении с соединением. Убедитесь, что разработанная программа выполняет все функции Winsock2 без ошибок.



Рисунок 1 – Реализация блоков 1, 2 и 5 схемы сервера.

**Примечание.** При разработке программ в заданиях этой практической работы рекомендуется использовать тексты программ, разработанных в предыдущей практической работе.

**Задание 2**. Реализуйте в программе **ServerU** блок 3 схемы сервера, изображенной на рисунке 3.4.1. Используемая в блоке функция **recvfrom** описана в разделе 3.12 пособия. Установите номер серверного сокета равным **2000**. Убедитесь, что при запуске программа **ServerU**  приостанавливаетсвое выполнение (переходит в состояние ожидания) сразу после вызова функции **recvfrom**.Завершите программу.

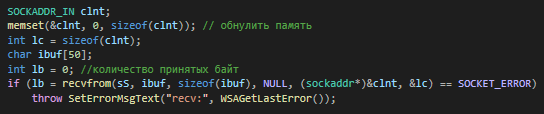


Рисунок 2 – Применение функции recvfrom в коде программы.

**7.4.4. Разработка клиентской части распределенного приложения**

**Задание 3**. Создайте новое С++ -приложение с именем **ClientU**. Реализуйте блоки 1, 2, 3 и 5 схемы клиента, изображенной на рисунке 3.4.1. Подключите функции обработки ошибок, разработанные в практической работе № 2. В параметре **to** команды **sendto** (раздел 3.12), установите адрес структуры **SOCKADDR\_IN**, содержащей IP-адрес равный **127.0.0.1** и номер порта равный **2000**. Обеспечьте пересылку сообщения ***Hello from ClientU***. Запустите на выполнение программу **ClientU** при отсутствующем сервере. Проанализируйте полученный код возврата.

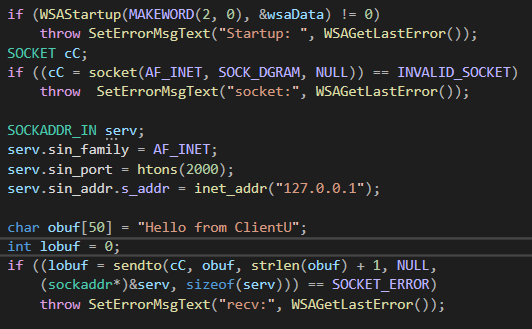


Рисунок 3 – Реализация клиента.

**7.4.5. Обмен данными между сервером и клиентом**

**Задание 4**. Запустите на выполнение программу **ServerU** и убедитесь, что она приостановила свое выполнение. Запустите на этом же компьютере программу **ClientU** и убедитесь, что программы сервера получила сообщение и завершилась нормально.

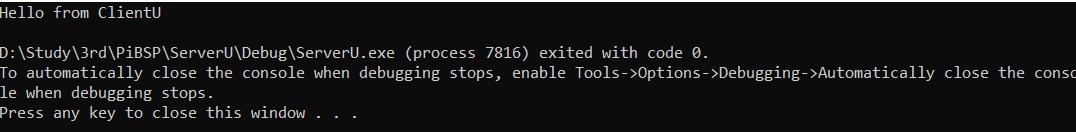


Рисунок 4 – Результат выполнения программ.

**Задание 5**. Реализуйте блоки 4 в обеих программах. Перешлите полученное сервером сообщение обратно в адрес клиента и убедитесь, что сообщение получено.

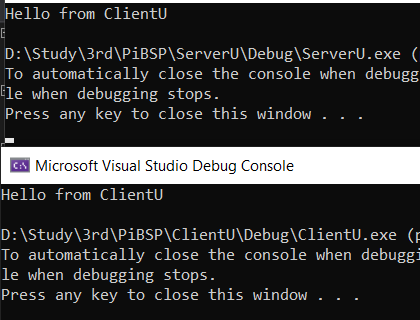


Рисунок 5 – Результат обмена сообщениями между программами.

**Задание 6**. Внесите необходимые изменения в программу **ClientU** для того, чтобы программы можно было бы расположить на разных компьютерах локальной сети. Убедитесь в работоспособности приложения.

**Задание 7**. Реализуйте последовательную пересылку данных от клиента к серверу и обратно по тому же принципу как это было сделано в заданиях 13, 14 практической работы № 2. Проведите измерения аналогичные оценки скорости передачи, сравните результаты.

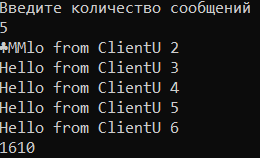
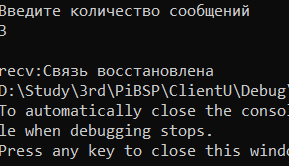


Рисунок 6 – Результат работы изменённых программ.

**Задание 8.** Запустите сервер **ServerU** на одном из компьютеров и одновременно два клиента на двух других компьютерах локальной сети. Оцените количество сообщений, которые успел передать и получить каждый из клиентов.

**Задание 9.** Запустите сервер **ServerT** (разработанный в практической работе № 2) и программу клиента **ClientU**. Объясните полученный результат.



**Задание 10.** Запустите сервер **ServerU** и клиент **ClientТ** (разработанный в практической работе № 2). Объясните полученный результат.

