ПЕРМСКИЙ ФИЛИАЛ ФЕДЕРАЛЬНОГО ГОСУДАРСТВЕННОГО  
АВТОНОМНОГО ОБРАЗОВАТЕЛЬНОГО УЧРЕЖДЕНИЯ ВЫСШЕГО  
ОБРАЗОВАНИЯ

«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ

«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

Факультет социально-экономических и компьютерных наук

Берсенёв Илья Иванович

**Лабораторная работа №1**

Теория и алгоритмы высокопроизводительных вычислений

г. Пермь, 2025 год

**Задание**

1. Изучите разработанное приложение. Запустите исполняемый файл сервера и клиента на одной машине, затем перенесите один из компонентов распределенного приложения на другой вычислительный узел (лучше на ноутбук, т.к. администратор ЛВС вуза может запретить обмен сообщения между различными узлами ЛВС) и попробуйте его запустить. Запустите несколько клиентов и попробуйте отправить сообщения одному серверу.

2. Запустите несколько серверов на одной машине. Отправьте им сообщения от нескольких клиентов. Объясните, почему приложение перестало работать.

3. Модифицируйте приложение так, чтобы существовала возможность на сервере идентифицировать клиентов не по имени вычислительного узла, а по нику/логину пользователя.

4. Модифицируйте приложение так, чтобы получился полноценный чат. Клиент может отправлять сообщения всем клиентам, участвующим в беседе. Для этого каждый клиент должен иметь возможность просмотра всех сообщений от всех клиентов, а сервер должен содержать список клиентов, которые хотят участвовать в беседе, чтобы каждый раз выполнять им рассылку сообщений.

**Решение**

**Задание 1**

При отправке нескольких сообщений с клиентов на один сервер все работает корректно, т.к. запись в пайп не блокирующая

**Задание 2**

поведение при нескольких серверах сервера читают из очереди попеременно. это происходит из-за следующего: a, b, c - сервер 1, сервер 2, сервер 3 a() lock at 0s b() lock at 2s c() lock at 5s

send message

a - got message, lock at 6s

b - lock at 2s

c - lock at 5s

send message

a - got message, lock at 6s

b - got message, lock at 9s

c - lock at 5s

send message

a - got message, lock at 6s

b - got message, lock at 9s

c - got message, lock at 15s

send message

a - got message, lock at 20s

b - got message, lock at 9s

c - got message, lock at 15s

то есть: рид из пайпа блокирует процесс. т.к. процессы не запускаются одновременно они блокируются в порядке запуска и встают в системный шедулер. шедулер шедулит процесс который ждал больше всего, самый ранний.

**Задания 3, 4**

Чат был реализован следующим образом:

1. Клиент запрашивает логин у пользователя и открывает именованный пайп с именем, имя состоит строго из латиницы
2. Клиент открывает пайп из п.1 на чтение как на сервере, это пайп для приема сообщений
3. При отправке сообщения клиент посылает запакованное сообщение с именем, днс и текстом
4. Сервер получает сообщение читая из пайпа, распаковывает его и добавляет в список адресатов, меняя поле «участники». После чтения и вывода сервер запускает рассылку этого сообщения по всем участникам, указывая текущее время, отправителя и текст.

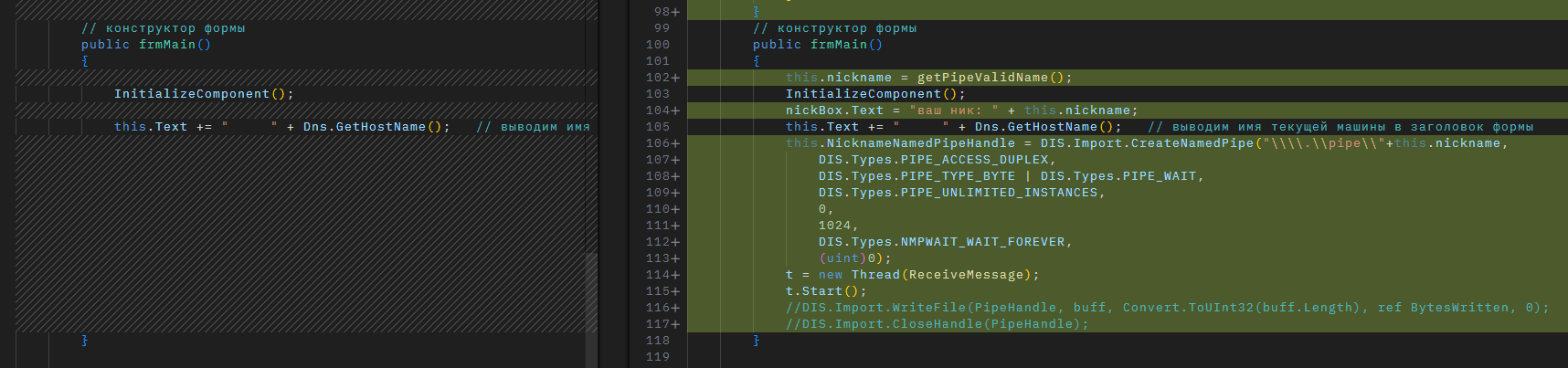
Также в последнем коммите было добавлено удаление клиента из рассылки если он закрыл пайп.

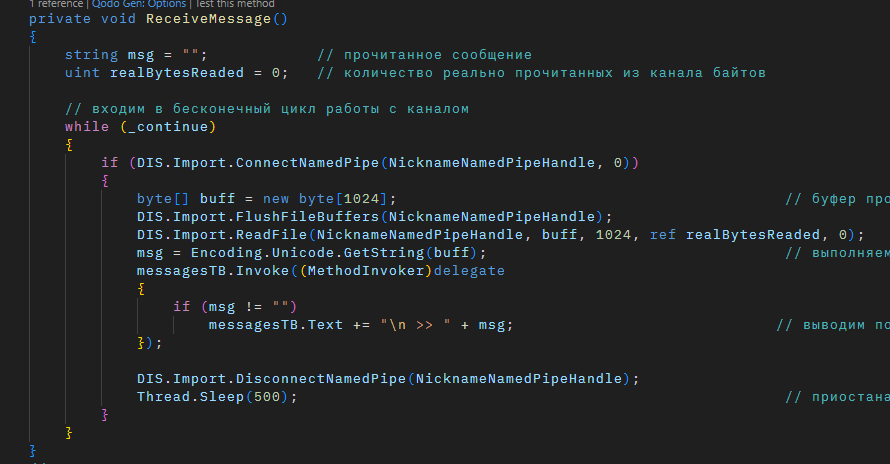
**Листинг работы**

Код реализации доступен в репозитории github по адресу https://github.com/ressiwage/LEARN-HSE-HPCTA-1-4

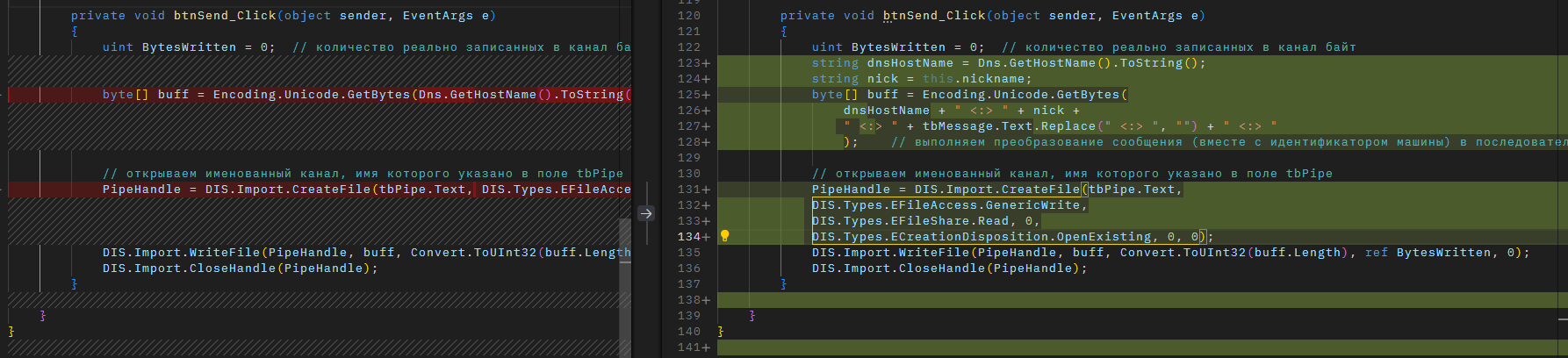
**Детали реализации**

Пункты 1,2, код клиента. Изменен конструктор формы, добавлена функция получения сообщения в треде.

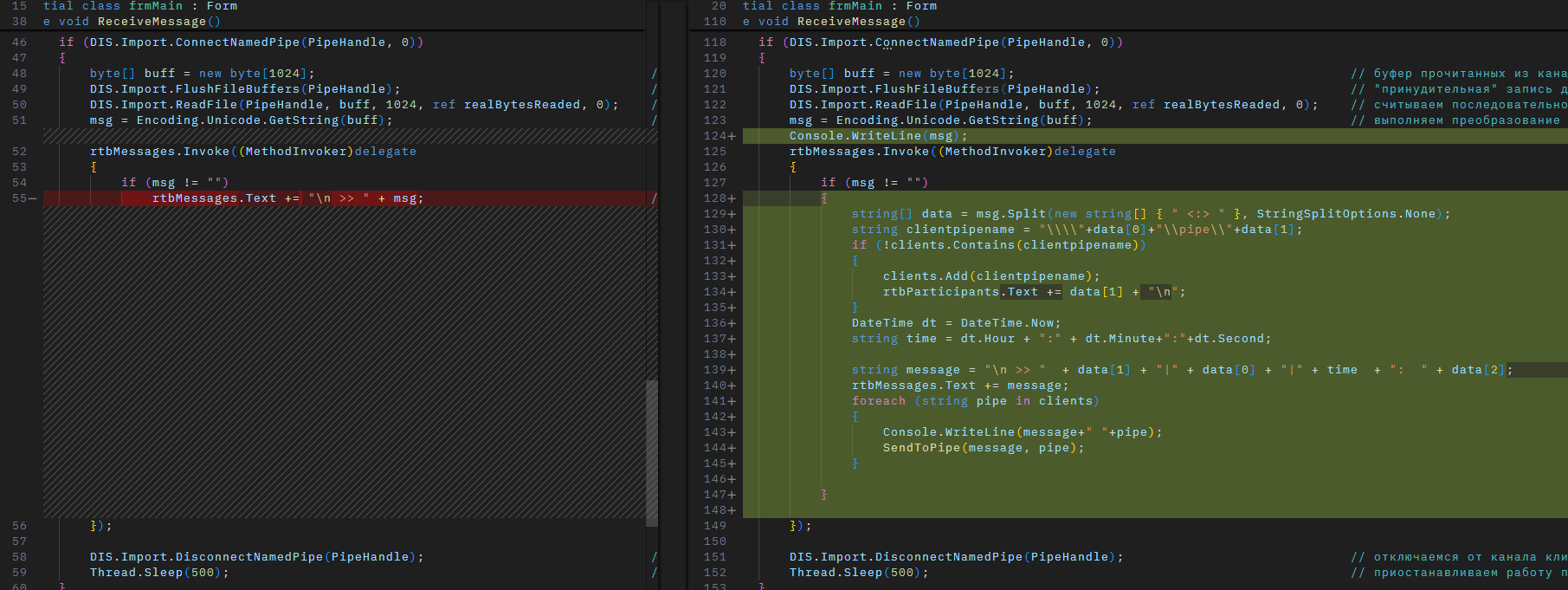




Пункт 3, код клиента, изменена логика обработки нажатия «отправить», чтобы отправлялось не только сообщение но и доп. Данные.

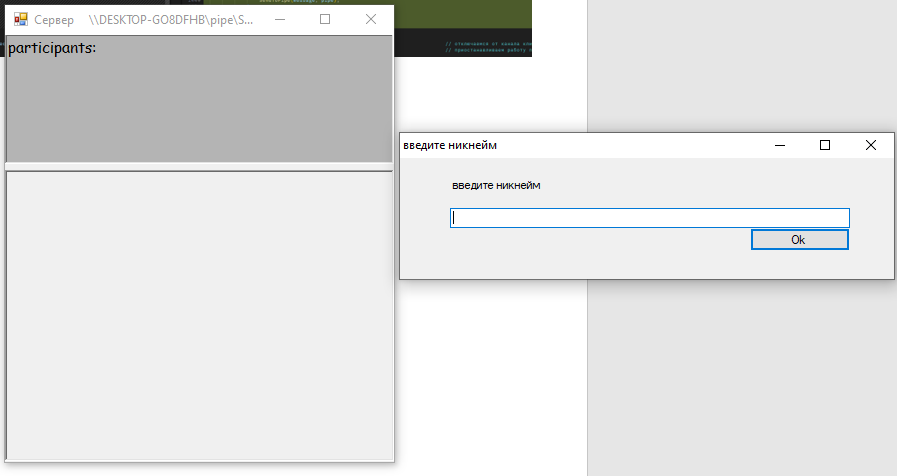


Пункт 4, код сервера, добавлена распаковка сообщения и рассылка. Скрыты функции SendToPipe, getPipeValidName, FormatValid, IsBasicLetter ввиду их неинтересности для отчета. Они нужны соответственно для отправки в пайп (взята стандартная реализация из предоставленного кода); запроса диалогового окна для получения никнейма для пайпа; функция для фильтра невалидного никнейма; низкоуровневая функция для фильтра невалидного никнейма.

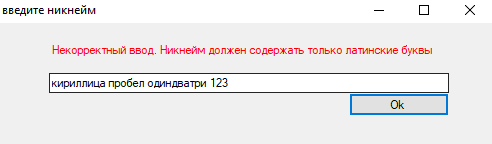


**Итоговый результат**

Внешний вид окон при запуске

****

Обработка некорректного ника в клиенте



Внешний вид окон после отправки сообщений от 2 клиентов. Второй клиент отправил сообщение позже первого, так что первое сообщение не попало в его рассылку.

