

Московский Авиационный Институт
(Национальный Исследовательский Университет)

Кафедра 806 «Вычислительная информатика и программирование»
Факультет: «Информационные технологии и прикладная математика»

Лабораторная работа
Дисциплина: «Операционные системы»
3 семестр
Задание 6

Группа:	М8О-208Б-18, №12
Студент:	Коростелев Дмитрий Васильевич
Преподаватель:	Миронов Евгений Сергеевич
Оценка:	
Дата:	25.02.2020

Москва, 2020

Содержание

- 1. Задание**
- 2. Адрес репозитория на GitHub**
- 3. Код программы**
- 4. Результаты выполнения тестов**
- 5. Объяснение результатов работы программы**
- 6. Вывод**

1.Задание

Вариант 49

Топология 4

Все вычислительные узлы хранятся в бинарном дереве поиска. [parent] — является необязательным параметром.

Набор команд 4

Формат команды:

> exec id > text_string > pattern_string [result] – номера позиций, где найден образец, разделенный точкой с запятой

text_string — текст, в котором искать образец. Алфавит: [A-Za-z0-9].

Максимальная длина строки 108 символов pattern_string — образец

Команда проверки 1

Формат команды: pingall Вывод всех недоступных узлов вывести разделенные через точку запятую. Пример: > pingall Ok: -1 // Все узлы доступны > pingall Ok: 7;10;15 // узлы 7, 10, 15 — недоступны

2.Адрес репозитория на GitHub

<https://github.com/Dmitry4K/labOS/labOS6>

3.Код программы

Lib.cpp – содержит описание классов Publisher и Subscriber – узлов заданной системы, а также методы, с помощью которых осуществляется взаимодействие между процессами.

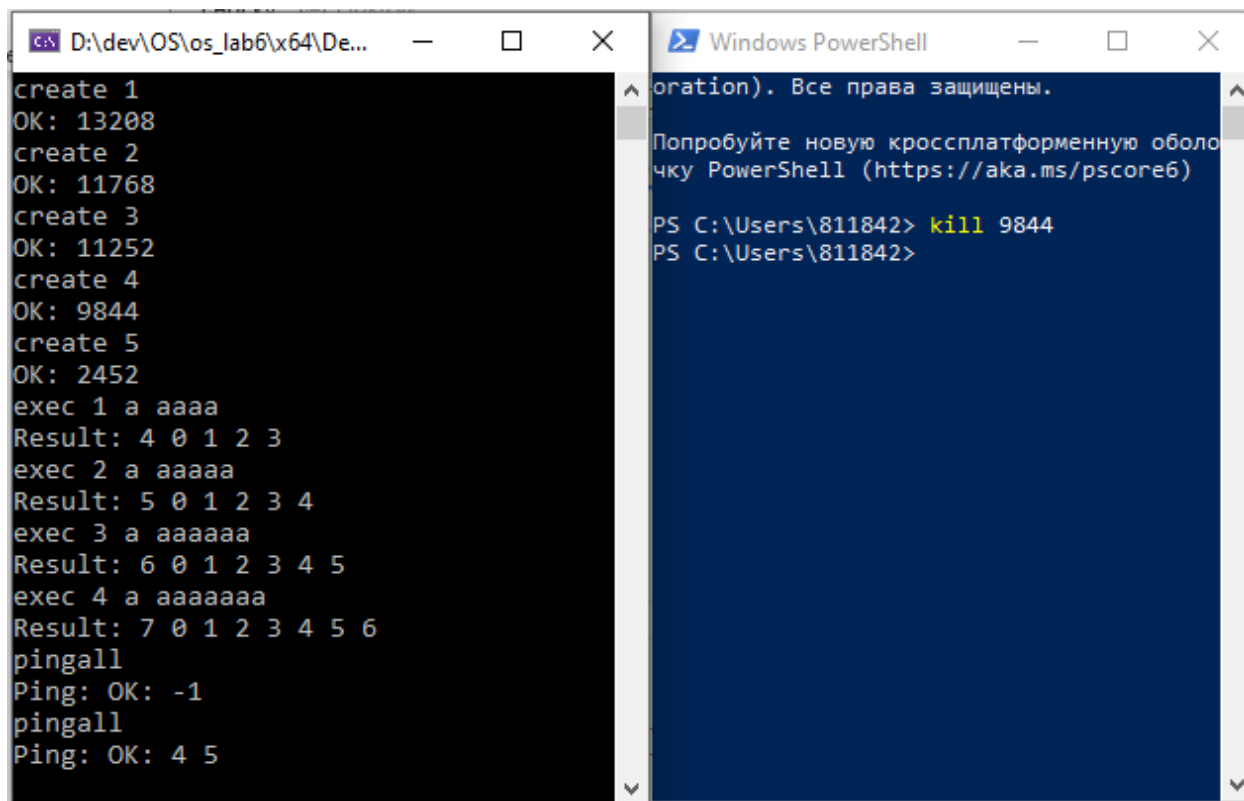
Lib.h – описание классов Publisher и Subscriber а также сигнатуры методов классов

BinarySearchTree.h – специальное бинарное дерево, которое хранит в себе все узлы системы: идентификатор в системе и сокет.

Publisher/main.cpp – правила работы узла publisher, в котором осуществляется чтение и ввод команд, отправка в команд в дочерние узлы.

Subscriber/main.cpp – правила работы узла subscriber, в котором осуществляется чтение и обработка команд, а также отправка команд в дочерние узлы.

4. Результаты выполнения тестов



```
create 1
OK: 13208
create 2
OK: 11768
create 3
OK: 11252
create 4
OK: 9844
create 5
OK: 2452
exec 1 a aaaa
Result: 4 0 1 2 3
exec 2 a aaaaa
Result: 5 0 1 2 3 4
exec 3 a aaaaaa
Result: 6 0 1 2 3 4 5
exec 4 a aaaaaaa
Result: 7 0 1 2 3 4 5 6
pingall
Ping: OK: -1
pingall
Ping: OK: 4 5
```

```
Windows PowerShell
(Information) Все права защищены.
Попробуйте новую кроссплатформенную оболочку PowerShell (https://aka.ms/powershell)
PS C:\Users\811842> kill 9844
PS C:\Users\811842>
```

5. Объяснение результатов работы программы

По заданию требовалось реализовать систему, по своей структуре схожей с бинарным деревом поиска, где самим деревом выступают узлы, которые общаются между собой посредством передачи сообщений с помощью сокетов.

В вершине данной системы лежит узел, который будет считывать команды и передавать их в дочерний узлы. Первый узел – Publisher (потому что он первым получает команду, которая должна обработать система), остальные – Subscriber(узлы, которые распределяют полученные команды между собой и выполняют некоторые из полученных команд. В данном узле хранятся три основных параметра – идентификатор в системе, идентификатор процесса в операционной системе и номер сокета, который хостит данный узел. Кроме этого узла также хранится id и сокет родителя и левого и правого потомка.

Обработка команд осуществляется посредством специальной обработки каждой команды, так как те имеют четкую структуру и каждое слово несет за собой определённый смысл.

Узел Publisher отвечает за ввод команд, их отправки и считывания результата работы системы. Ввод осуществляется с помощью стандартного потока ввода, далее команда преобразуется, генерируется новый сокет, который должен будет хостить новый узел. Отправка, так же как и в узле subscriber осуществляется с помощью средств библиотеки ZeroMQ. Результаты выполнения всех команд отправляются на специальный, заранее забронированный сокет, на отдельном потоке.

6.Вывод

Данная лабораторная работа позволяет учащемуся попробовать спроектировать свою систему, связанных посредством сокетов и общения через них с помощью сообщений. Для реализации данной системы требовалось применить как старые знания о процессорах, многопоточности и синхронизации и т.п. так и новые знания в области сокетов, протоколов соединения и проектирования системы, что оказалось одной из самых сложных задач в данной лабораторной работы.