МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное   
образовательное учреждение высшего образования  
«Самарский национальный исследовательский университет   
имени академика С.П. Королева»

(Самарский университет)

Институт информатики, математики и электроники

Факультет информатики  
Кафедра технической кибернетики

**Отчет по лабораторной работе №4**

Дисциплина: «Большие Данные»

## Тема: «Введение в ZooKeeper»

Выполнил: Бабонин Д.А.

Группа: 6133-010402D

Самара 2020

**ЗАДАНИЕ**

1. Запустить ZooKeeper;
2. изучить директорию с установкой ZooKeeper;
3. запустить интерактивную сессию ZooKeeper CLI и освоить её команды;
4. научиться проводить мониторинг ZooKeeper;
5. разработать приложение с барьерной синхронизацией, основанной на ZooKeeper;
6. запустить и проверить работу приложения.

**ХОД РАБОТЫ**

Загрузим ZooKeeper, переименуем файл zoo\_sample.cfg в директории confв zoo.cfg. Запустим сервер и клиент, подключимся к сессии.

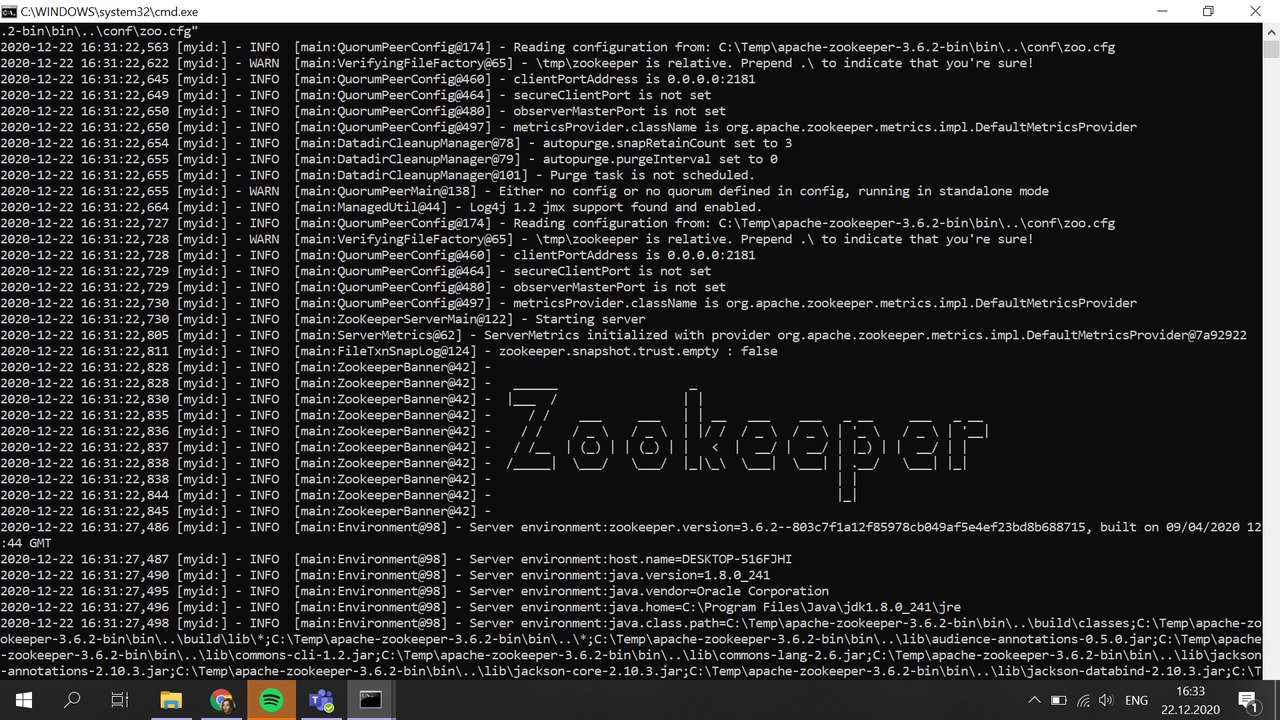


Рисунок 1 –запуск сервера Zookeeper

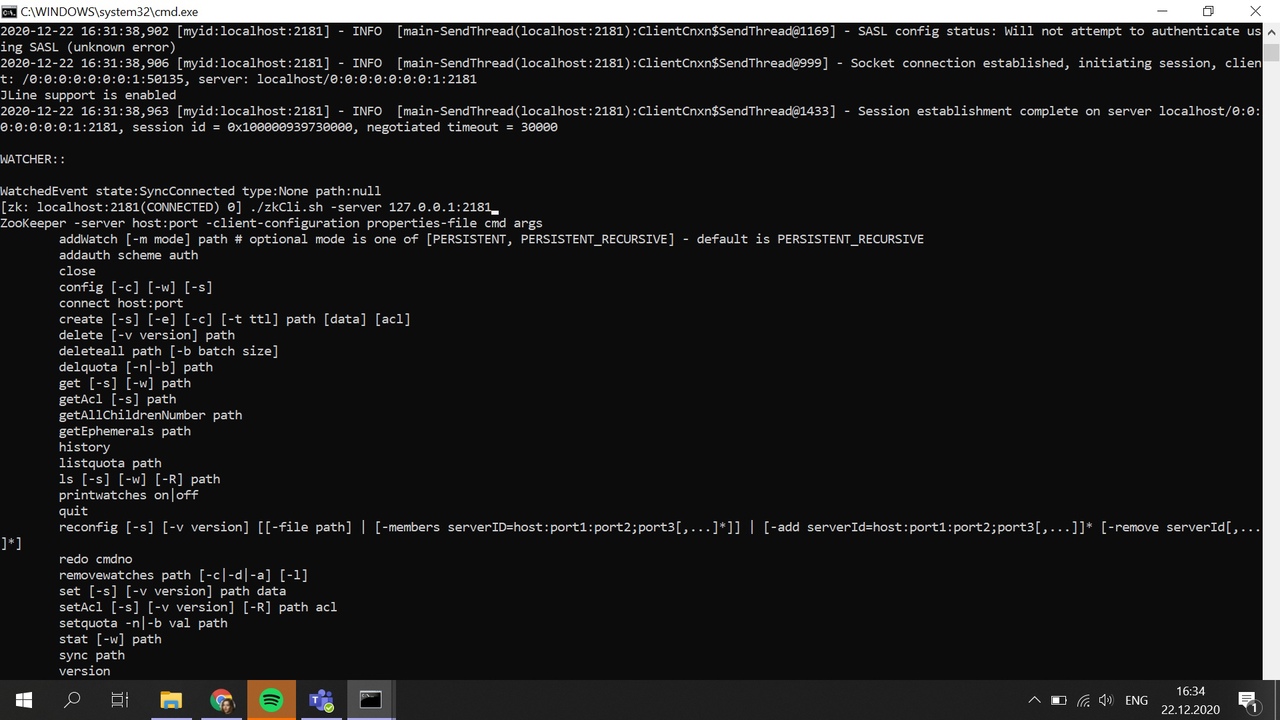


Рисунок 2 – Подключение к сессии

При помощи команды ls получим список узлов. Создадим узел /mynode с данными first\_version и убедимся, что он создан. Затем заменим данные узла на second\_version. Создадим два нумерованных узла в качестве дочерних.

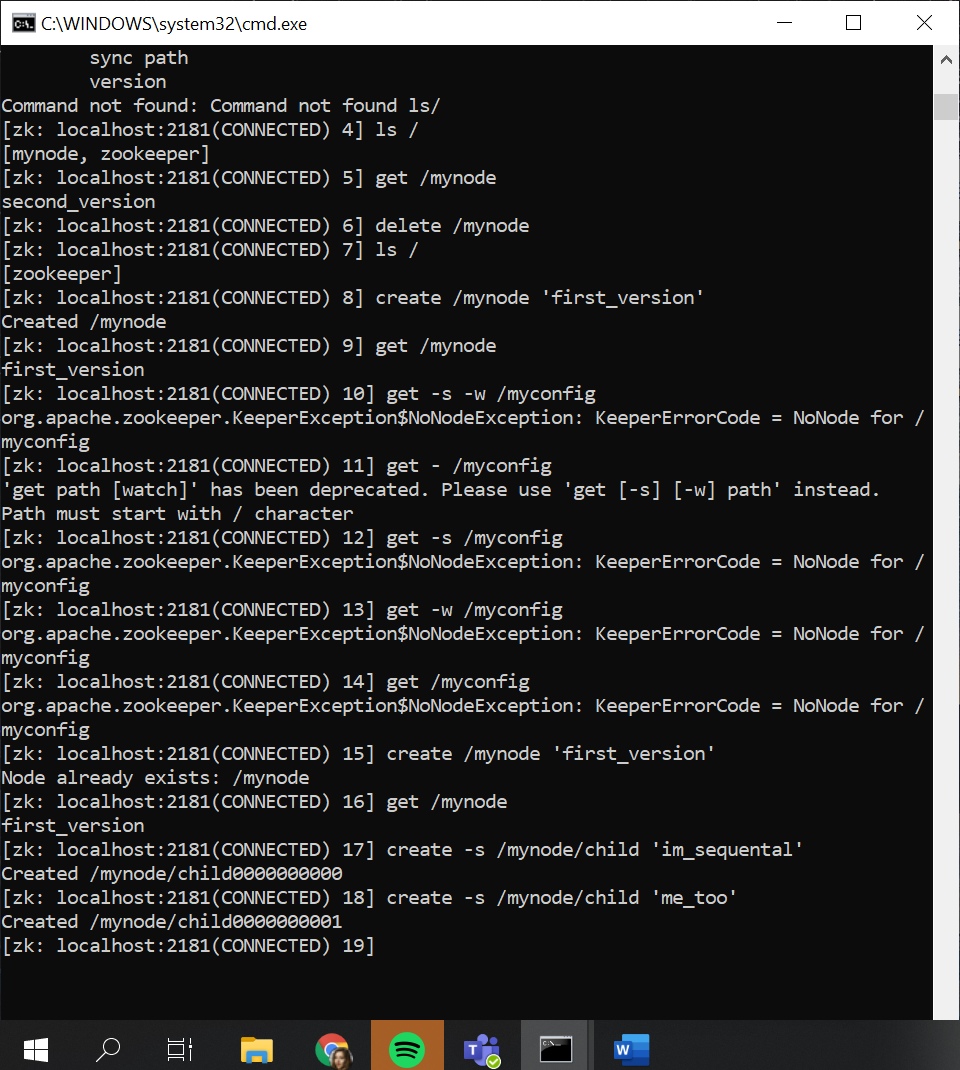


Рисунок 3 – Взаимодействие с ZooKeeper через командный интерфейс

Внутри CLI сессии, создайте узел mygroup. Откроем две новых CLI консоли и в каждой создадим по дочернему узлу в mygroup. Консоль 1 – grue, консоль 2 - bleen. Проверим в исходной консоли, что grue и bleen являются членами группы mygroup. Эмулируйте аварийное отключение и проверяем узел и затем удаляем его

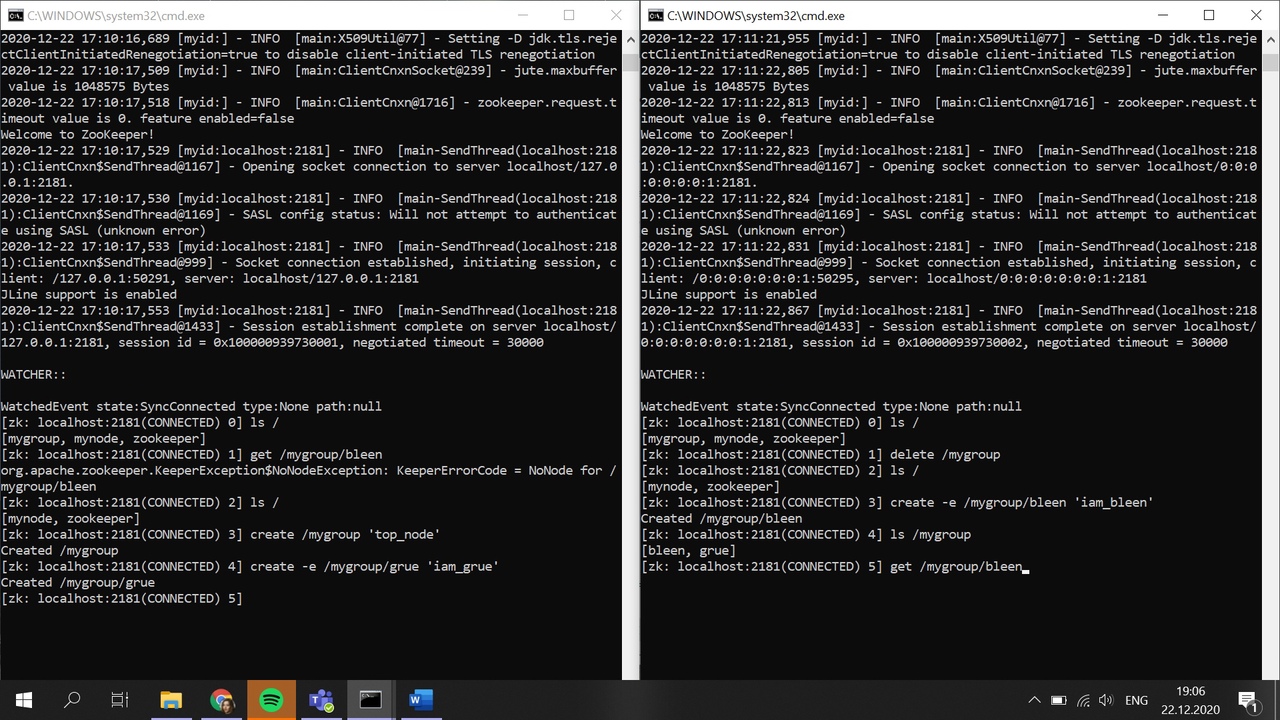


Рисунок 4 – Взаимодействие с двумя консолями

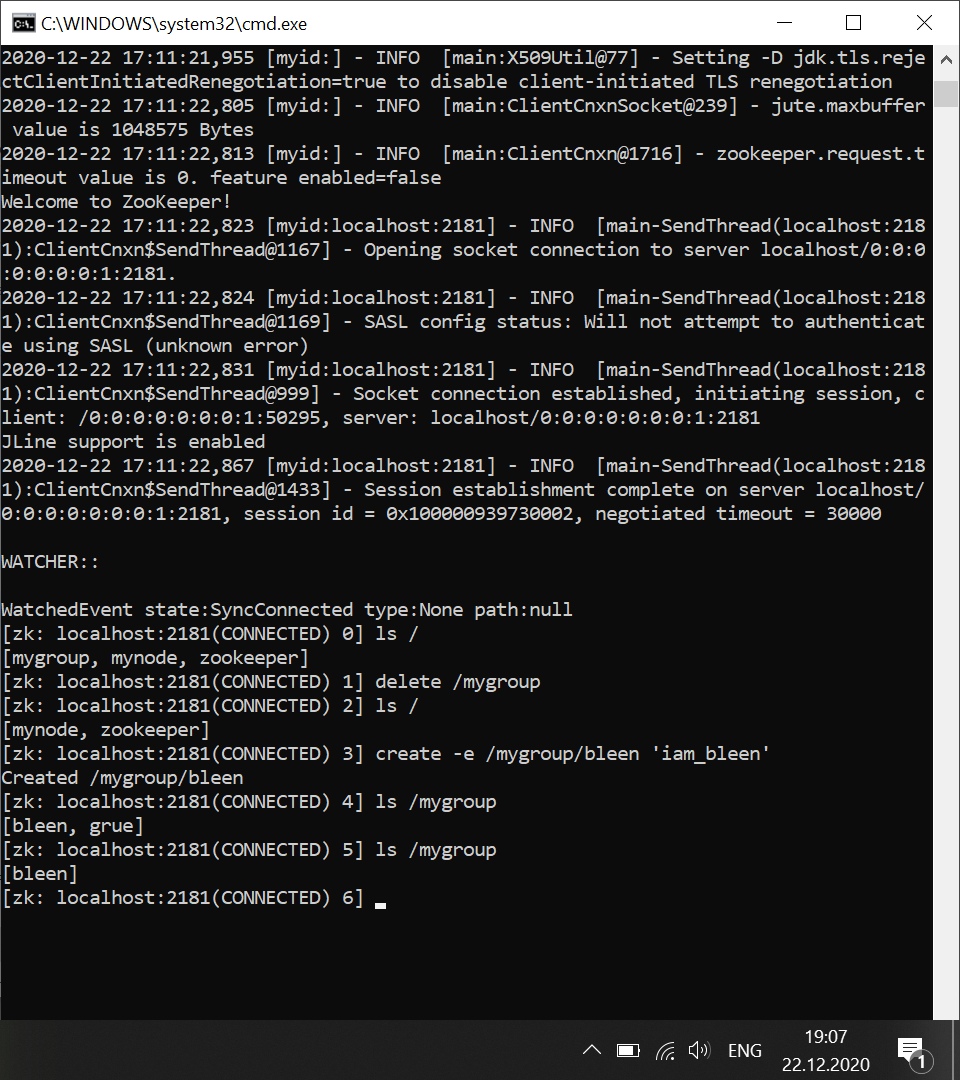


Рисунок 5 – Проверка узлов

Таким образом клиенты могут получать информацию о появлении и отключении других клиентов.

Создадим в корне узел "myconfig" в задачу которого будет входить хранение конфигурации. Откроем новую консоль и подключимся к ZooKeeper. Установим watch-триггер.

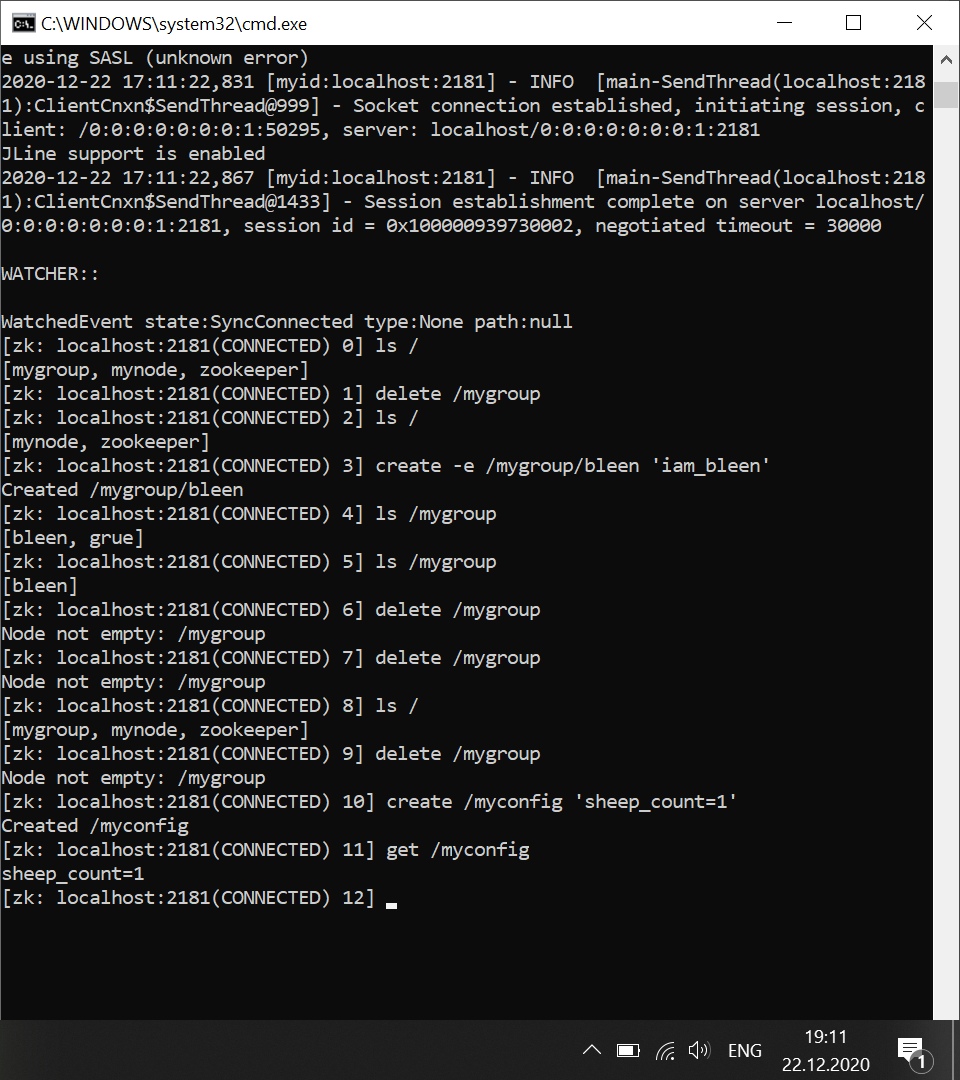


Рисунок 6 – Создание узла myconfig

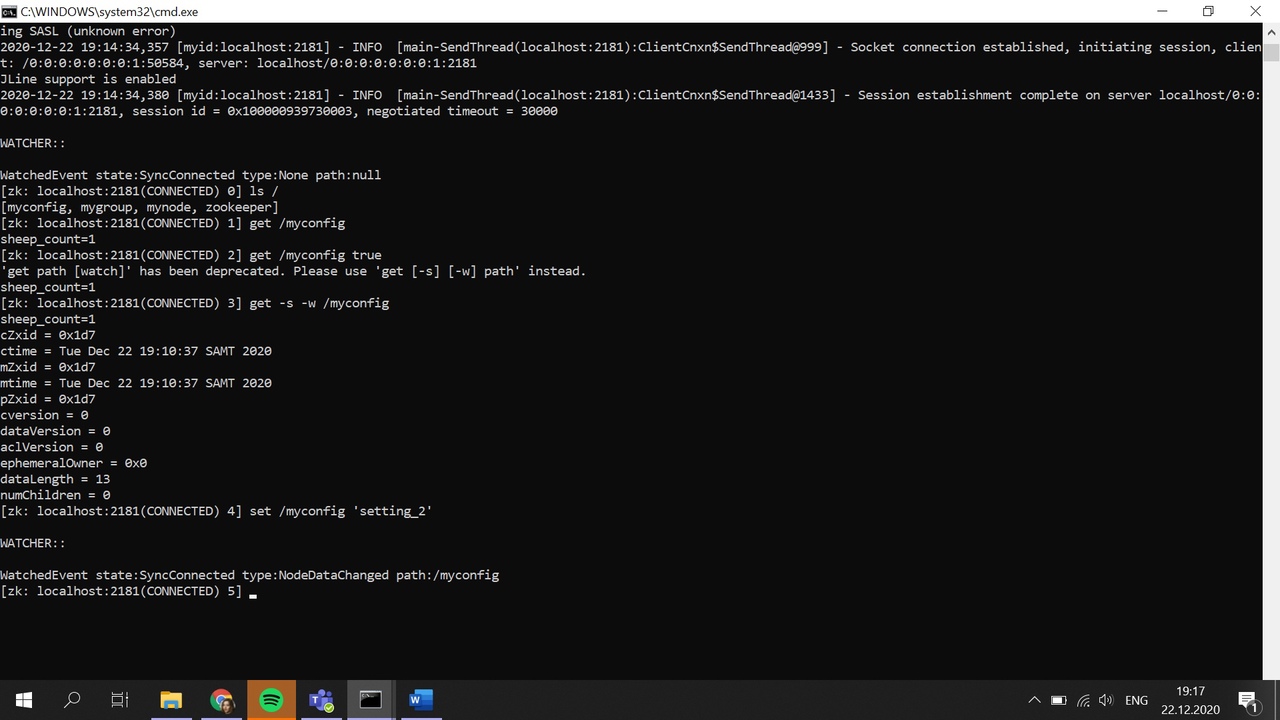


Рисунок 7 – Установка watcher для узла myconfig

Изменим значение myconfig в первом терминале. Во втором терминале появилось оповещение.

Создадим приложение зоопарк, корневым узлом будет являться /zoo/, который создаем заранее. Каждое животное входит в зоопарк, создает дочерний эфемерный узел в зоопарке со своим именем. После запуска первого приложения zebra1 идет ожидание остальных животных. При запуске zebra2 каждое зебра начинает бежать и останавливается через определённый период времени. В конце приложения эфемерные узлы удаляются.

Запустим несколько клиентов распределённого приложения и проверим его работу. Так же проверим случай для одной обезьянки и вечеринки на одного. Работает, даже в случае вечеринки на одного

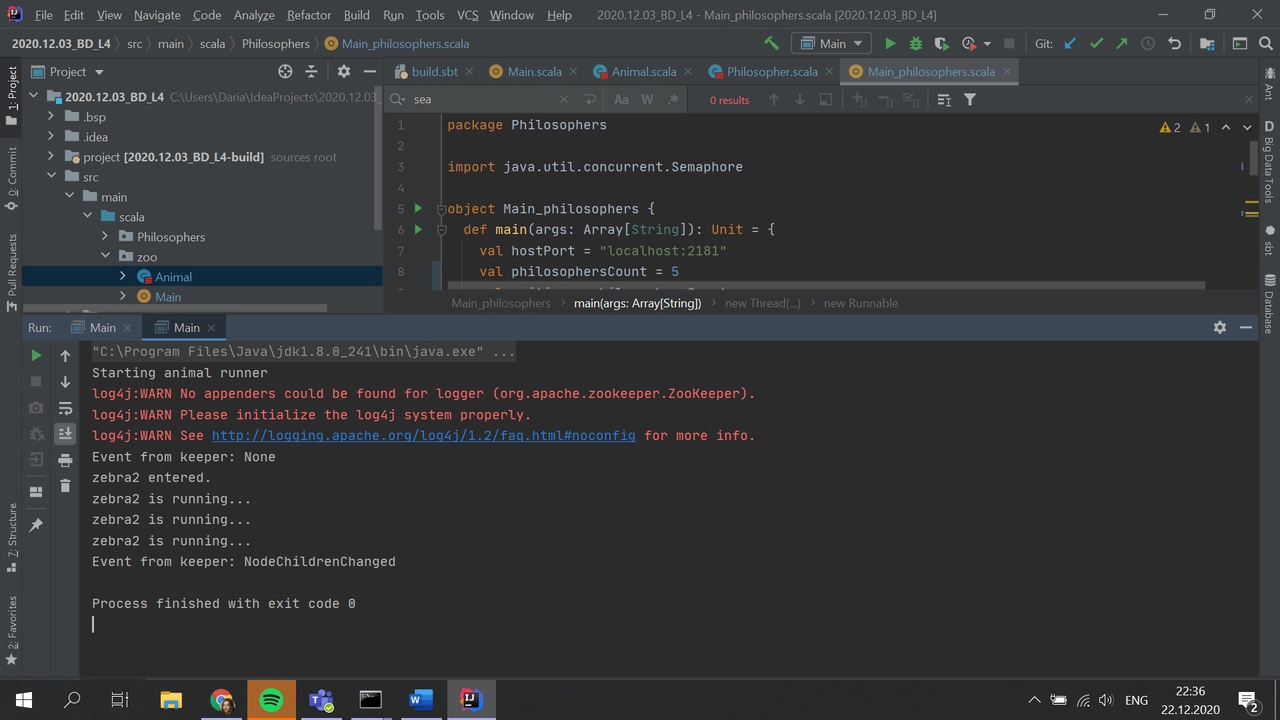
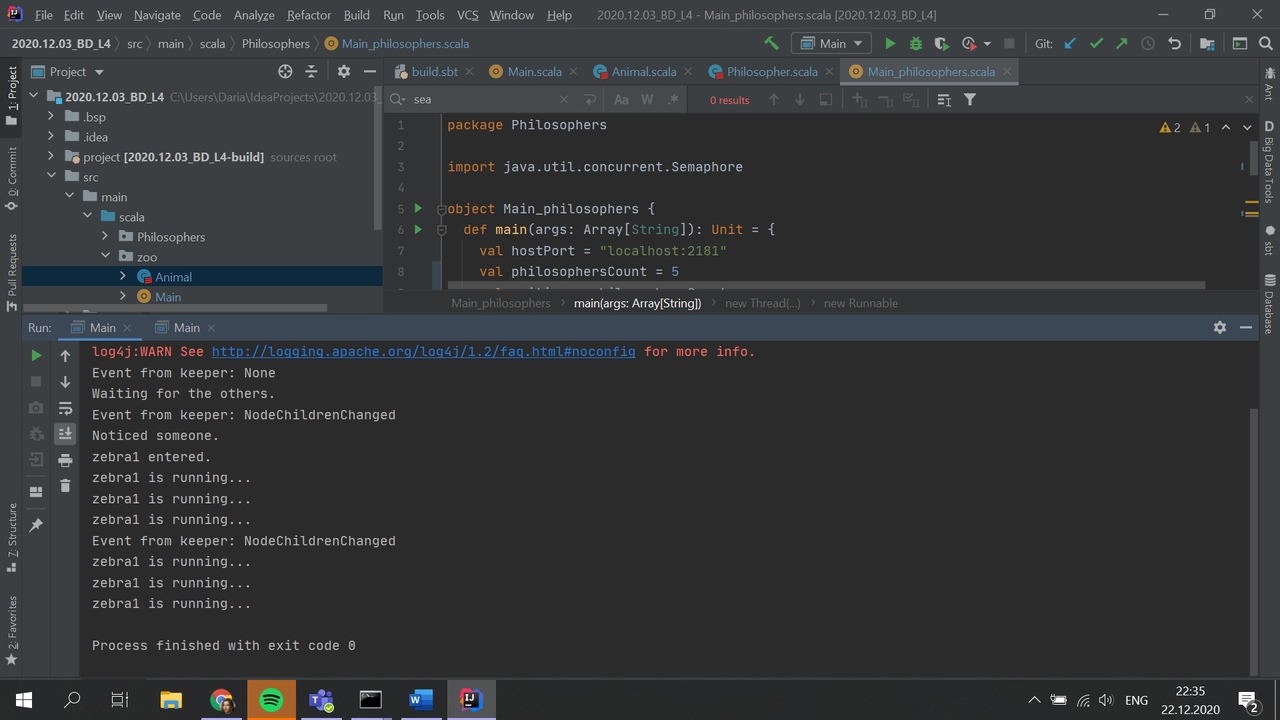


Рисунок 9 – Зоопарк

Пять философов располагаются за круглым столом. Философ может размышлять или кушать. Для приема пищи в центре стола большое блюдо с неограниченным количеством спагетти и тарелки, по одной перед каждым философом. Поесть спагетти можно только с использованием двух вилок. Для этого на столе располагается ровно пять вилок – по одной между тарелками философов.

Решение задачи состоит в следующей. Каждый философ сначала берет левую вилку, затем – правую, кушает, кладет правую вилку, затем – левую и прекращает прием пищи, начинает прокрастинировать. Для исключения ситуации когда все философы взяли по левой вилке и никто не кушает, для 0 философа изменим порядок взятия вилок.. Он берет правую вилку, затем левую, нарушая общий порядок взятия вилок

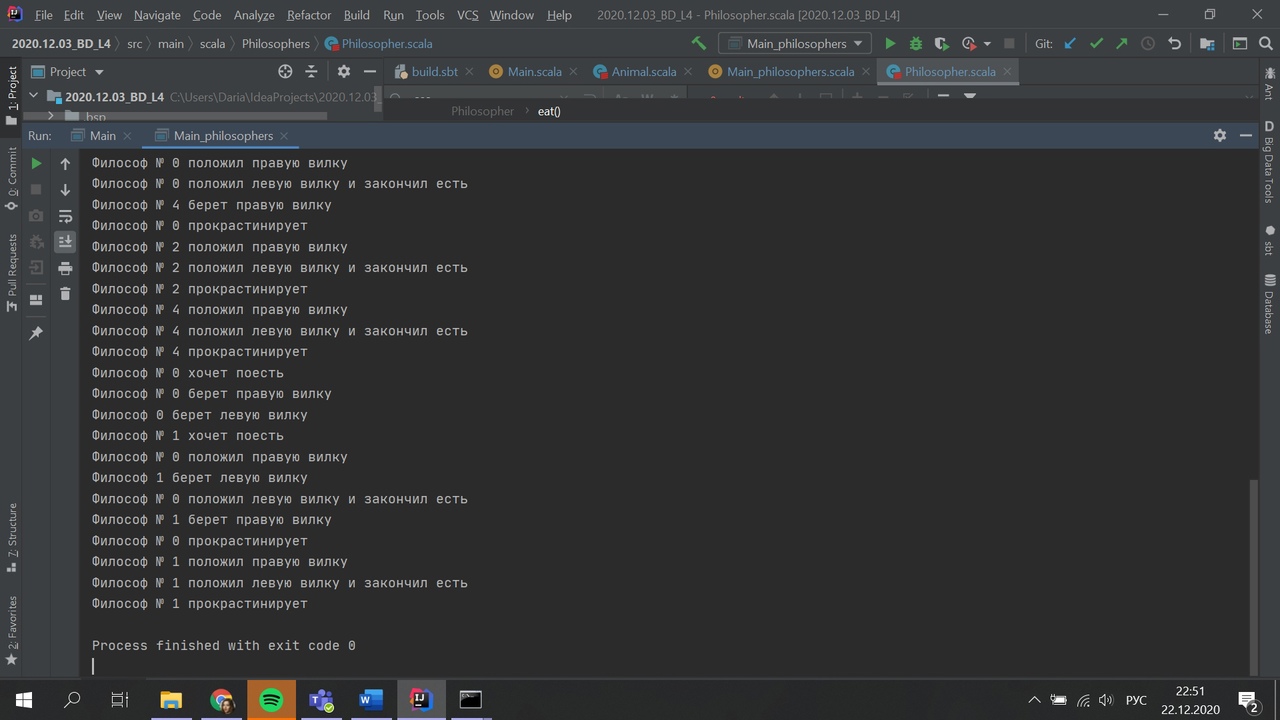


Рисунок 10 - Философы

Вывод программы:

Философ № 2 хочет поесть

Философ № 4 хочет поесть

Философ № 1 хочет поесть

Философ № 3 хочет поесть

Философ № 0 хочет поесть

Философ 3 берет левую вилку

Философ 2 берет левую вилку

Философ 1 берет левую вилку

Философ № 3 берет правую вилку

Философ № 3 положил правую вилку

Философ 4 берет левую вилку

Философ № 3 положил левую вилку и закончил есть

Философ № 4 берет правую вилку

Философ № 2 берет правую вилку

Философ № 3 прокрастинирует

Философ № 2 положил правую вилку

Философ № 2 положил левую вилку и закончил есть

Философ № 1 берет правую вилку

Философ № 2 прокрастинирует

Философ № 3 хочет поесть

Философ 3 берет левую вилку

Философ № 4 положил правую вилку

Философ № 3 берет правую вилку

Философ № 4 положил левую вилку и закончил есть

Философ № 4 прокрастинирует

Философ № 2 хочет поесть

Философ № 1 положил правую вилку

Философ 2 берет левую вилку

Философ № 1 положил левую вилку и закончил есть

Философ № 0 берет правую вилку

Философ № 1 прокрастинирует

Философ 0 берет левую вилку

Философ № 4 хочет поесть

Философ 4 берет левую вилку

Философ № 3 положил правую вилку

Философ № 3 положил левую вилку и закончил есть

Философ № 2 берет правую вилку

Философ № 3 прокрастинирует

Философ № 0 положил правую вилку

Философ № 0 положил левую вилку и закончил есть

Философ № 4 берет правую вилку

Философ № 0 прокрастинирует

Философ № 2 положил правую вилку

Философ № 2 положил левую вилку и закончил есть

Философ № 2 прокрастинирует

Философ № 4 положил правую вилку

Философ № 4 положил левую вилку и закончил есть

Философ № 4 прокрастинирует

Философ № 0 хочет поесть

Философ № 0 берет правую вилку

Философ 0 берет левую вилку

Философ № 1 хочет поесть

Философ № 0 положил правую вилку

Философ 1 берет левую вилку

Философ № 0 положил левую вилку и закончил есть

Философ № 1 берет правую вилку

Философ № 0 прокрастинирует

Философ № 1 положил правую вилку

Философ № 1 положил левую вилку и закончил есть

Философ № 1 прокрастинирует

**ЗАКЛЮЧЕНИЕ**

В результате лабораторной работы была изучена директория и установка ZooKeeper. Освоены команды для работы с клиентом. Разработано приложение с барьерной синхронизацией, основанной на ZooKeeper.